Object Signature Search for Capturing Processes Memory Image of Windows System

Khairul Akram Zainol Ariffin, Ahmad Kamil Mahmood, Jafreezal Jaafar, and Solahuddin Shamsuddin

***Abstract*—Over the past few years, memory analysis has been an issue that has been discussed in digital forensics. With the introduction of cloud computing, the analysis on memory has become critical as the hard disk is no longer the primary choice to store information and data on the computer system. The online storages with password protected such as ADrive, Dropbox and Google Cloud Storage are already available to all users. Hence, with the progress of development in this technology, the traditional approach (analysis on hard drive) has become obsolete in obtaining information from those applications. The aim of this paper is to present an algorithm that can be used to trace the processes of the memory image. The algorithm uses the signature search to find the possible process that is stored in the memory dump. Then, by the information in Parent ProcessID (PPID) and ProcessID (PID) the Process Block Tree is constructed. Further, the benchmarking test between Process Enumeration technique and this new algorithm is presented in this paper.**

***Index Terms*—Algorithms, Information Retrieval, Memory Analysis, Signature Search.**

I. Introduction

For over the years, computer has becomes common in our daily life. Many activities in our daily life, starting from writing the document to observing the stock market has involved the use of computer system. Therefore, it can be concluded that the introduction of computer and the technology within it has ease our daily routine. However, it also brings a negative effect such as cybercrime to our world.

The Council of Europe's Cybercrime Treaty defines the cybercrimes as a range of crime that is committed using the computer, network and hardware devices [1]. Further, Symantec divides the cybercrimes into two main perspectives such as the single event that facilitated by the crime-ware and the range of activity, starting from cyber stalking to stock market manipulation [2]. In RSA 2012 Cybercrime Trends Report, it is reported that the cybercrime has shown no signs of slowing down. The report also concluded that in every minute, 232 computers have been infected with the malware [3].

Due to the increasing in cybercrimes, a field known as Digital Forensic has been established. This field involve with collecting, preserving, analyzing, documenting and presenting the contents of computer as evidence of cybercrime [4]. For over a decade, the investigation in Digital Forensics has been focused on the analysis of the non-volatile devices. Until recently, and with the merge of online storage, the volatile device such as computer memory has become critical in the investigation of Digital Forensic.

The research on the volatile memory is started in 2005 where Digital Forensic Research Workshop (DFRWS) has organized a Windows Memory Challenge. During the event, two analysis tools have been developed which known as Memparser [5] and KntList [6]. Apart from that, the important of volatile memory analysis has also been listed out in Jesee, K [7]. Nevertheless, the sensitive information such as password, encryption keys and username only available in volatile memory where it is used to stored temporary processes and data before transmitted to the Central Processing Unit (CPU).

Manuscript received June 18, 2013; revised August 20, 2013.

K. A. Z. Ariffin is with the Digital Forensic Department, CyberSecurity Malaysia, Mines, Selangor, Malaysia (e-mail: akram@cybersecurity.my).

A. K. Mahmood and J. Jaafar are with the Computer Information Sciences Department, Universiti Teknologi Petronas, Perak, Malaysia (e-mail: kamilmh@petronas.com.my, jafreez@petronas.com.my).

S. Shamsuddin is with the Research Department, CyberSecurity Malaysia, Mines, Selangor, Malaysia (e-mail: solahuddin@cybersecurity.my).

II. Theory

A. Internal Structures and Address Translation in Computer Memory

An explanation on theories of internal structures and address translation has been outlined in Dhamdhere, M [8] and Russinovich and Solomon [9]. Both explain fully on the function of the internal structure and address translation in the computer memory with Russinovich and Solomon focus directly towards Windows Operating System. Apart of that, the books also explain on the procedure of executive object creation and information storage with regard to the object.

Nevertheless, the procedure of storing data by kernel is also demonstrated by Amari, K [10]. In the demonstration, it illustrated that the kernel has set on pool to store the objects.

Most of the data in the computer memory are stored in the paged pool. Carrier, B [11] deduct that most of the data is stored in the paged pool as it allows the data to be transferred into hard disk if the computer memory is running low in space. Meanwhile, the important objects such as process and thread blocks are stored in the non-paged pools as the kernel need to access them frequently. Since the running process still remains intact all the time if the system still on power, therefore they will be available during the acquisition of the physical memory [12].

Apart from that, Virtual Address Description of a process block has a purpose of tracking the status of the process's address space. This internal structure is maintained by the memory manager and it stores the information on the attributes of the object such as range of the address, inheritance of child and object's security. Due to the information that is stored in this structure, a tool known as VADtool has been developed with purpose to track the

DOI: 10.7763/IJCCE.2013.V2.278

memory mapped files from the memory dump [13].

B. Persistence of Data in the Computer Memory

A study has been carried out by Garfinkel, Chows and Rosenblum in 2004 that concluded 86% of the data contents still remain in the computer memory. From the study it showed that there were 7Kb of data was still remaining in the memory for up to 28 days. Further, a complete data will remain in the memory if the machine only undergoes the soft rebooting. [14]

C. Analysis Technique for Memory Dump

XORSearch[15] is a tool that is designed based on string search technique. It takes a keyword as an input and then performs the search throughout the memory dump. Further, the tool can also help to find the keyword that has obfuscated by using either Exclusive OR (XOR) or Rotate Left (ROL) function that comes with it.

Windows Operating System represents each process in volatile memory as process block. It contains pointers to both next and previous process blocks. Further,the blockstores the information on attributes of the process together with pointer to other data structures that is related to it. Process Environment Block (PEB) which is one of the internal structure of process block, is responsible to store the location of executable files and the DLL's path. Due to this fact, AccessData [16] group has designed a tool that is known as Forensic Toolkit (FTK). This tool will parse the process block to enumerate all the contents within memory. It also applies Directory Table Base (DTB) information in the address translation algorithms in order to identify the process in memory. Further, Windows Memory Forensic Toolkit (WMFT) has applied this technique by tracking the PsActiveProcessHead link to capture all the active processes in the memory dump [17]. Apart from that, Ruichao Zhang, Lianhai Wang, Shuhui Zhang [18] had demonstrated the data extraction from memory dump by using Kernel Processor Control Region (KPCR). In the demonstration, the information such as running processes, current network connection, file content and other data can be extracted from the image.

S. M. Hejazi, C. Talhi, M. Debbabi [19] had outlined the use of aplication or protocol fingerprint to trace the active application in the memory. The test was conducted to track online application such as email and messenger where from the result, it showed that each of the application had used fingerprint representation.

PTFinder [20] is a tool that applies the file carving where the technique is done linearly to recover only the contiguous file. This technique is applicable because most operating systems will convert the file to be contiguous file instead of fragment files.

III. Methodology

The algorithm is designed to capture the process by using process signature search. The approach will make use of the process signature to track the process block that is available in the computer memory. Then, once all the process blocks have been retrieved, the information about the process blocks are read and captured.

Apart from that, a process block tree will be constructed to link the processes within the computer memory due to their status (parent or child process). The overall working of the algorithm is based on the rules that are discussed below:

Rule 1: Searching for process signature and detect the true process block

In theory, all process blocks in the computer memory except idle process are defined by a unique signature value. This signature is known as proã which has a constant hexadecimal value of 50726fe3 (H). This signature is located outside and before the location of the process block. The location of the signature is different among the Windows operating system. Table I lists out the location of the process signature for all Windows version until Windows Vista.

TABLE I: Offset for Process Signature

Windows Operating System	Offset of the signature from starting Process Block (in Hex)
2000	0x01c
2003	0x0c0
XP	0x01c
Vista	0x024

After all possible process blocks have been located, they are then captured, dumped and stored in the database. The size of the possible process that is to be dumped and stored is based on the equation below:

$$\text{size} = (\text{start process offset} - \text{proãoffset}) + \text{size of process block} \quad (1)$$

Once all the possible process blocks have been dumped, a process of selecting the true entity is run to remove the false process blocks. In general, it is not guaranteed that all the captured possible process blocks represent the true process block. This is due to the nature of computer memory which is random and volatile, the processes or data from different period of time will still remain in it until they are overwritten by the new data. Due to this scenario, additional information is needed to remove the data or entity from different period of time. The easier way to accomplish this is by using the information that is stored in the ImageFileName. This information is stored in "uchar" format and only the process block with recognizable character (in word with .exe) is chosen as a true entity. When all the true entities have been detected, the false process will be removed from the database.

Rule 2: Construct the Process Block Tree

In theory, the process block in the computer system has a unique process ID (pid). This value together with Parent process ID (ppid) can be used to determine the child and parent for the process. The Parent process ID is stored in InheritProcess ID offset in the process block. There are two cases in determining the parent and child process in the memory dump:

- If the value in the ppid of the process is the same asthe value of the pid for other process (still remains in the memory dump), then this process is a child of other.
- If the ppid value of the process block does not reflect on any process, then it is a standalone process. This normally happens if the parent process is no longer in

the computer system

IV. Experiment

The test is conducted on the existing memory dump that is available in Digital Forensic Research Conference (DFRWS) website [21], [22]. It allows the author to directly compare the result with the work that has been done in the past as many researchers have make use of this memory dump.

The memory dump is run using Hex Workshop software, and the string/ hexadecimal value search technique is applied to retrieve the possible process block in the memory dump. Once the possible process block has been allocated, it will be stored in the database for further process. In the database, the process block is then being sorted out using the Parent ProcessID (PPID) and ProcessID (PID) to build a process block tree.

TABLE II: Success Rate for Tested Memory Images

Source of memory Image	No of possible block	No of successful block	Succes rate (%)	Source of memory Image
DFRWS 2005	52	46	88	DFRWS 2005
Boomer Win 2000	41	31	76	Boomer Win 2000
Boomer Win 2003	38	30	79	Boomer Win 2003
Boomer XP	78	64	82	Boomer XP

V. Result and Discussion

From the test, the algorithm captures about 52 possible Process Blocks. Then, out of 52 possible results, only 46 blocks that have been chosen to represent the Process Block. The selection of the process block is due to the fact that the information that is stored in ImageFileName offset of process block is represented in" uchar". Hence, only the result with recognizable character for the ImageFileName is chosen as the true Process Block. Once the true Process Block has been selected, the value that is stored in ProcessID (PID) and Parent ProcessID (PPID) will be retrieved. By using these two values, the Process Block Tree is then constructed. The Parent ProcessID (PPID) is defined as the possible child process to parent Process Block in the memory dump.

An additional work has been conducted to obtain the success rate of retrieving the true process block. As discussed in previous section, the true process is defined as the entity that resides in the computer image at the same timeframe. Hence, all type of processes will be included as the true process block, except for the 'old' process. The success rate is computed by using the following formula:

$$Successrate(\%) = \frac{noof trueprocess}{noof possibleprocess} \times 100 \quad (2)$$

No of true process: include active, exile, hidden and duplicate process.

The computational success rate for all the tested memory images are shown in Table VI. From the table, the rate of success is between 75% to 100% whereas the total average for five tested memory images gives a value of 83.2%. The table also shows that the success rate is not proportionate to the number of possible process blocks.

The other important object that has been retrieved with the algorithm is the duplicate process block. This type of process block normally occurs twice at different location in the computer memory. In summary, table IV lists the total number of duplicate process block that have been retrieved from the tested memoryimages.From table IV, there are three duplicate process blocks that have been retrieved in DFRWS 2005 memory image. These processes are known as winlogon.exe, dfrws2005.exe and HKServ.exe.

One of the advantages of this algorithm is the ability to retrieve the exile process from the computer memory. Table III summarizes the number of exile process blocks that are still resided in the tested memory images.

TABLE III: Total Number of Exile Process

Source of Memory Image	No of exile process
DFRWS 2005	7
Bommer Win 2000	3
Boomer 2003	5
Boomer XP	3

TABLE IV: Information on Duplicate Process in Dfrws 2005 Memory Image

Process	No of block	Starting offset	CR3 Register	PID	PPID
HKServ.exe	2	0x2bf86e0 0x2f806e0	0x2ce7000	972	820
DFRWS.exe	2	0x0e1fb60 0x2f806e0	0x6c98000	784	668
Winlogon.exe	2	0x01048140 0x01045d78	0x04fe4000	176	156

TABLE V: The Benchmarking Test

Factor	New algorithm	Previous technique (Extended version, Volatility)
Technique	Process Signature Search	Process Block Enumeration
Requirement knowledge	Operating System	Operating System and System Architecture
Involvement address translation algorithm	No	Yes
Ability to retrieve hidden and exile process	Yes	No
Number of Process Retrieved (for DFRWS 2005 memory image)	46	36
Ability to track other information from other internal object (i.e Threads, PEB, etc)	No	Yes
Advantages/Disadvantages	+ less knowledge needed + easy for new officer + fast + able to trace hidden/exile process _ not able to retrieve other internal objects _ information only at surface level	+ can retrieve many information with regard to the process + able to trace other internal object _ require more knowledge and complicated _ not able to retrieve hidden and exile process

For benchmarking purposes, this algorithm has been tested together with the current available tool for memory analysis such as Volatility. From the result, it shows that the new algorithm is able to retrieve more processes compared to the current available tool. However, since there is no address translation algorithm that has been applied in the new algorithm, its ability to retrieve other object is also limited. The summary of the benchmarking test is listed in Table V.

VI. Conclusion

From the result of the experiment, it shows that the Process Block can be retrieved by using the process signature search. Once the Process Block has been identified, the Process Block Tree can be constructed by a combination of ProcessID (PID) and Parent ProcessID (PPID). Since the mechanism only relies on the signature of the process, it allows the algorithm to capture all type of the Process Block. Hence, it also improves on retrieving any hidden and exited Process Block that still remains in the memory dump.

For further work, other object signature will be traced and applied on the algorithm to improve the chance of retrieving other important blocks.

Acknowledgment

The author would like to present his gratitude towards CyberSecurity Malaysia and Universiti Teknologi Petronas for the help and support that had been accorded during the research period.

References

[1] T. Krone, *High Tech Crime Brief*, Australian Institute of Criminology, Australia: Canberra, ISSN 1832-3413. 2005.
[2] US-Cert, "government organization, Computer Forensic," published report, USA, 2008
[3] RSA, "The current state of cybercrime and what to expect in 2012," published report, USA, 2012.
[4] C. E. Hill, "What is the Definition of Digital Forensics," in *eHow, How to do just about everything*, 2012.
[5] DFRWS, *Memparser Analysis Tool by Chris Betz*, 2013.
[6] DFRWS, *Kntlist Analysis Tool by George M. Garner Jr*.
[7] K. Jesee, "Using every part of the buffalo in Windows memory analysis," *Journal Digital Investigation*, vol. 4, pp. 24-29, 2007.
[8] D. M. Dhamdhere, *Operating Systems: A Concept based* Approach, 1 ed., McGrawHill, 2009.
[9] M. E. Russinovich, D. A. Solomon, and A. Ionescu, Windows®*Internals Covering Windows Server® 2008 and Windows Vista®*, J. Pierce, Editor., Microsoft Press, 2009.
[10] K. Amari, "Techniques and Tools for Recovering and Analyzing Data from Volatile Memory," SANS Institute, 2009.
[11] B. G. J. Carrier, "A Hardware Based Memory Acquisition Procedure for Digital Investigations," *Journal of Digital Investigation*, pp. 13-18, 2004, March.
[12] A. Schuster, "Searching for processes and threads in Microsoft Windows memory dump," *Journal Digital Investigation*, vol. 3, pp. 10-16, 2006.
[13] B. D. Gavitt, "The VAD tree: A process eye view of physical memory," *Journal Digital Investigation*, pp. s62-s64, 2007
[14] T. Garfinkel, B. Pfaff, J. Chow, and M. Rosenblum, "lifetime is a systems problem (Published Conference Proceedings style)," in *Proc the ACM SIGOPS European Workshop*, pp. 2-9, ACM, 2004
[15] D. Stevens, *XORSearch*, January 30, 2007.
[16] AccessData Corporation, *Importance of memory Search and Analysis*, Published White Paper, Lindon, UT, 2006.
[17] M. Burdach, *An Introduction to Windows memory forensic*.
[18] L. W. R. Zhang and S. Zhang, "Windows Memory Analysis Based on KPC," presented at the 2009 Fifth International Conference on Information Assurance and Security, IEEE, Xi'An China.
[19] S. M. Hejazi and C. T. M. Debbabi, "Extraction of forensically sensitive information from windows physical memory," *Journal digital investigation*, vol. 6, pp. S121 – S131, 2009
[20] A. Schuster, *PTFinder*, 2006.
[21] DFWRS. Windows Memory Challenge. [Online]. Available: http:// www. Workshop/Conference.com
[22] K. Jesse, "Computer Forensics Reference Datasets," *CFReDsProject, ManTech*, vol. 2, 2011.

Khairul Akram earned his bachelor and master degree with first class honours in system engineering with computer engineering from University of Warwick, United Kingdom in 2008 and 2009 respectively.

He later joined Universiti Teknologi Petronas (UTP) in 2010 to pursue his journey towards academic research and teaching courses to earn his PhD in Information System. During his time in UTP, a number of journal articles and conference papers have been produced and published internationally. Currently, he is appointed as Researcher in Digital Forensic Department, CyberSecurity Malaysia and has been entrusted with the research on embedded system forensics. His passion in research is towards algorithms, embedded system, image processing and audio authentication. He is a member of IET professional group.

Ahmad Kamil Mahmood earned his bachelor and master degree in actuarial science and statistics from the University of Iowa, Iowa City, USA in 1986 and 1988 respectively. He later joined UUM, Bank Negara Malaysia, Public Service Department and PETRONAS. After 10 years in the industry, he continued his journey in the academia serving the Universiti Teknologi PETRONAS in 1998 teaching courses for the Bachelor Degree in ICT and BIS.

He earned his PhD in Information Systems from the University of Salford, UK in 2005. With his research team, a number of journal articles and conference papers have been produced and published internationally. Currently, his industrial collaboration research projects keep him occupied while supervising 7 post graduate students and assuming the Dean of Faculty of Science and IT.

Jafrezal Jaafar obtain B. Sc in computer science from Universiti Teknologi Malaysia in 1998, MAppSc. (IT) from RMIT University (Australia) in 2002 and PhD from University of Edinburgh (Scotland, UK) in 2009. Previously he works as System Engineer for several years.

He is currently the Head of Department for the Computer & Information Sciences Department, Universiti Teknologi PETRONAS, Malaysia. His research interests are in the area of Soft Computing and HCI. He is actively involved in a number of research works and secured research grants in these areas. He has also produced numerous journal, conference and workshop papers.

Solahuddin Shamsuddin received his PhD from University of Bradford, United Kingdom in Network Security in 2008. He received a post-graduate Diploma in Systems Anaysis from UiTM in 1991. He started his career with the Malaysian Armed Forces after completing his first degree in Electrical Engineering from Wichita State University, USA in 1986. He served in the Royal Signal Regiment of the Malaysian Army for 10 years holding various posts as communications engineer and IT manager before joining the industry after the completion of his stint with the Malaysian Armed Forces.

In 1997 he joined Softlabs Technologies Sdn Bhd as the General Manager. He was entrusted to manage and lead system development projects with various industries such as oil and gas, defence, telecommunications and local governments. In 2002 he joined National ICT Security & Emergency Response Centre (NISER) now known as CyberSecurity Malaysia as the Expert Service Manager. Later on, he was entrusted to be the manager for Malaysia Emergency Response Team (MyCERT). He has earned 4 professional certifications namely CWNA, CISSP, CEH and BS7799 Lead Auditor.

With his knowledge and skills in various security domains, he is now entrusted to be the Chief Technology Officer at CyberSecurity Malaysia. He is also the research coordinator for CyberSecurity Malaysia.