On The Analysis Of Information Found On Windows Application Memory

Article *in* International Journal of Intelligent Computing Research · June 2013

DOI: 10.20533/ijicr.2042.4655.2013.0042

4 authors:

Funminiyi Olajide
Nottingham Trent University
31 PUBLICATIONS **122** CITATIONS
SEE PROFILE

Nick Savage
University of Portsmouth
31 PUBLICATIONS **308** CITATIONS
SEE PROFILE

Galyna Akmayeva
15 PUBLICATIONS **53** CITATIONS
SEE PROFILE

Charles A. Shoniregun
Infonomics Society
117 PUBLICATIONS **349** CITATIONS
SEE PROFILE

On the Analysis of Information Found on Windows Application Memory

Funminiyi Olajide,
School of Engineering,
University of Portsmouth,
United Kingdom

Nick Savage,
School of Engineering,
University of Portsmouth,
United Kingdom

Galyna Akmayeva,
Infonomic Society
Republic of Ireland

Charles Shoniregun,
Infonomic Society
United Kingdom

Abstract

Digital forensic community feels the urge for the development of tools and techniques in volatile memory analysis. The extraction of user input from physical memory of Windows applications may reveal useful information that could be used as evidence in crime cases; the information that may not be found on traditional hard disk forensic investigations. However, there have been few digital investigations into the amount of user input recovered from Windows application memory. This paper presents on the analysis of user input stored on an application, and the forensically relevant information recovered from the memory of some commonly used Windows applications. Quantitative results of the experiments carried out on these applications will be presented.

1. Introduction

Of recent, there has been demand for more tools and techniques to be developed for capturing memory images and analyzing their content [1]. This is because the development of digital forensic investigation techniques has focused mostly on evidence contained within the hard disks. The user input stored on the memory allocated to an application may be recovered and analysed. The information could be used to determine the activity of the user input stored on Windows application memory. This information may not be visible when using traditional hard disk forensic investigation tools and techniques such as EnCase [2] and Forensic Toolkits[3].

However, there have been few investigations regarding the analysis of the information that may be acquired from commonly used Windows applications. The investigations that have previously been conducted have focused mostly on user data persistence, age of information, whereby the ages of deallocated pages have no bearing on the time of subsequent reallocation. On the research experiment carried out by [4], there is no visible dependency of the age of free pages, but it does demand for memory until the resources on were severely depleted. The majority of pages persisted for less than 5 minutes with single pages only lasting longer. The research limitation shows that it is not possible to determine the age of text segment pages. A research of [5] indicate a framework for automatic evidence discovery and correlation from a variety of forensic targets. This research only improves the automatic correlation of forensic data from the file system. In the previous research, they have not identified the quantity of the user input stored on an application and the information that may be recovered from the volatile memory of Windows applications.

In this paper, the quantity of information recovered and user input stored on an application is determined by the mean value of user input found on repetitions of the application memory, the percentage of user input recovered from the memory and the average length of user input found in continuous block of the application memory. The user input was dispersed in the application memory, but the quantity of user input was calculated based on the extracted application level information.

Application level information is the extracted user input information from the commonly used Windows applications [6]. Application level information is defined as information which indicates how the user is (or in the case of terminated process, was) using an application. It is expected that application level information will include spreadsheet data, word document text, browser text, email text and any other user stored data related to the application. By extracting and identifying this application level information, a clear picture of the actions carried out by the user on the Windows systems can be built and analysed. There is possibility that user actions on the applications can be re-created or reconstructed.

There is then the possibility that this reconstruction process may illuminate further digital information, which can be used to support digital evidence in the court of law. The application level information is relevant to the original user input on applications, when the application is closed or running on Windows systems. Paging is used as memory management allocation in Windows operating system. The paging was performed when a program tries to access pages that are not currently mapped to the physical memory and the Random Access Memory (RAM) [7]. This could be referred to as page fault and hence, the operating systems take control of the page fault in a manner invisible to the program.

The operating system determine the location of the data in the auxiliary storage, obtain an empty page frame in RAM as a container and then load the available page frame data [8]. This is updated to show new data and then return control to the program that causes the page fault. In this paper, investigations into the quantity of user input from the extracted memory dump processes of Windows applications has been carried out.

In the experiment, the quantity of information is determined while images were captured when the applications have been closed, and Windows system is running, the user is still interacting with other applications. This investigation is contained within the memory of Windows applications. The user input recovered may become an essential tool for information assurance, as well as solving computer related crimes. This information may also be used for forensic digital electronic disclosure and discovery of user input stored on the applications.

2. Related work

Memory analysis of forensically relevant evidence requires skills and understanding of some specific issues relating to volatile data. This is a data stored in the memory (RAM) and that will be lost when the computer loses power or is powered off. This type of data resides in registers, cache and Random Access Memory (RAM). The paper of [9], issued a memory analysis challenge to encourage research and tool development in this direction. This tool is used to reconstruct the Windows command history from a Windows XP memory capture, but it is not capable of carving a wide variety of such objects from memory. Therefore, there is need for a research process of extracting user-entered data from memory captures. The research of application level evidence [10], from physical memory, identified the important aspects of memory analysis and proposed an approach for application level evidence. A paper of [11], identified the dispersal of time aspect of information stored on physical memory. This approach provides prospective information on how evidence is dispersed in the memory of some commonly used application during memory analysis.

A research paper [12] was reviewed, and focused on digital investigation analysis technique of the computer history model. This research was carried out to ascertain how forensic framework is sufficient for all type of digital investigation. A research of “Volatools” [13], is used as an integrated volatile memory tool for some digital investigation processes. This area of research is reviewed as limited to the encrypted file systems.

A paper of [14] discusses tools and techniques on acquiring volatile operating system data and therefore, presented volatile data that may be vital in determining criminal activity on digital devices or computer related crimes. It was presented that much work has been done with memory analysis to recover processes using the EPROCESS data structure. However, little work has been done on data carving process of the application metadata. Based on this notion, the research gap was identified and this paper is focused on the extraction of user input from the memory allocated to an application. This also include the digital forensic investigations of the quantity of user input stored on these applications, as well as the information recovered from some commonly used Windows applications.

3. Forensic memory capture

In order to analyse the memory allocated to an applications, it must first be captured and extracted. This process can be achieved by capturing the contents of memory in Windows computer system when the system is still running. In this experiment, user interacted with the sample applications, and then the application was closed. However, the Windows system is still active and images were captured at every 30minutes while user is interacting with other applications. This approach is to ascertain the amount of forensically relevant user input that can be recovered from Windows systems. This information is stored on the memory allocated to the application. In order to achieve this, find the list of processes related to the sample applications, by means of the Volatility framework. Once the process number is known, the memory allocated can be extracted for further digital investigation. Process is associated with the applications that is running and or closed on Windows systems. This process is attributed with a

name of the application that the user had interacted with on computer systems. However, the process of capturing the memory of a Windows system and extracting the memory allocated to a particular process is typically done by separate software. In order to capture the memory of a Windows system, there are varieties of tools that can be used. Some of these tools required specialist hardware to be added to the Windows computer system, so as to extract the memory image. An example of this type of tool has been developed by Garcia, [15]. Garcia's tool is a hardware-based memory acquisition tool that uses a PCI extension card to dump the memory content to an external device.

Hardware based acquisition tools have the advantage that they do not require any additional software to be executed on the computer system when acquiring evidence. However, they are limited in that they do need to have the tool installed on every machine prior to that investigation. For this reason there has been much focus on software-based tools. DD is widely tool for dumping memory but has a file size limitation and it does includes into images part of the disk that are not parts of the file system [16]. Msuiche has developed a command line tool that can be used on Windows to extract the volatile memory (RAM) of the computer system, [17]. It also reconstructs the virtual address space of system process and other processes. However, this tool is limited and can be accessible only while the command Windows is still open. Win32dd is the only tool that could be used to collect the contents of volatile memory (RAM) for Windows. It will dump memory in a raw format but does not contain processor state. Nigilant32 overcomes this limitation. This is another tool that the investigator uses to preview a memory image and take a snapshot of it.

Nigilant32 has a small footprint, using less than 1 MB in memory when loaded on Windows computer systems and with a minimal impact during acquisition [18]. In this paper, Nigilant32 tool was used to image the physical memory of computer systems. Other tools that may be used to extract the memory allocated to an application are all software based tools. For example, the Volatility Framework [19] is more extensive and is capable of analysing a variety of memory image formats such as DD format, crash dump and Hibernate Dumps. There are few tools that will perform Windows memory capturing and extracting memory allocated to the applications. An example of one of these tools is FatKit [20], FatKit enables the automation of the extraction of objects from memory. However the limitations of FatKit are that it is used for a static memory dump file analysis and in a real mode it has a 4GB segments limit and this is why it has not been used in this research. In order to make the results of the experiments as applicable as possible, a normal working environment was replicated while capturing memory images. In achieving this, the computer would be turned on at the start of the business day and then turned off at the end of the day. When the computer is first turned on, the application will be opened. User input was made once on these applications. After this, the application was then closed and not re-opened. However, the user will be interacting with the systems as the system is still running. In this research, there is typical experiment of user input on internet web pages, powerpoint and outlook email documents. These applications were closed and the volatile image was captured as set interval while the system is still running. Examples of 30 minute block of user input on these applications are as shown in Table 1.

Table 1. User interacting with the application

Sample Application	At 30 minutes User Action was made Once
PowerPoint 2007	Write slides of text, with commas, semi-colon, full-stop. Click on slide show, click from beginning or from current slide, add animations, save or do not save, or do nothing.
Outlook Email 2007	Write paragraph of text, with commas, semi-colon, full stop send and receive email data from and to recipient, or do nothing, save document or do not save document.
Internet Explorer 7.0	Open or click on breaking news or stories. Click back or front, save or do not save. Highlights texts, search for texts or do nothing.

The original user input on this application was made once and this contain powerpoint slides, outlook email contents, opening of internet web pages, browsing, click on news or stories, click back or front, save pages or do not save. The application was then closed but system was still running and images were captured at set interval of 30minutes. The result of the investigation reveals that retention of user input on this application is also significant for information assurance or digital fraud. In this paper,

we are presenting the results of PowerPoint 2007, Outlook Email 2007 and Internet Explorer 7.0.

Table 2. Quantitative result

Metric	PowerPoint 2007	Outlook Email 2007	Internet Explorer (IE7.0)
Mean Evidence Repetition	52.45	170.51	448.53
Mean % of Evidence found	25	53	71
Average Length of Evidence found in Continuous block	44.08	29.09	61.70

Outlook 2007 is used because it is mostly used in business organisation and because it does not necessarily need to determine which server to send requests to, it connections is 1GB long but terminated when this limit is reached. Internet explorer is one of the most commonly used applications in business organisation. It support commonly used web browsers, and its productivity is at its best. PowerPoint 2007 is used in this experiment because it is also considered to be mostly used in business organisation for data presentation. It is a presentation software program, and being part of the Microsoft Office package, the applications uses a graphical approach in the form of slide shows and accompany the oral delivery of the topic to audience. This application is widely used in businesses and classrooms, an effective tool for training purposes.

Nigilant32 was used to capture the memory of the machine and Volatility was used to extract the memory allocated to the process. Series of tests were run for days until 100 images were captured on each application. The physical memory of computer system was 2 Gigabytes (GB) and this resulted in 200 GB of images being captured. After the collection of data, we made copies of images captured on each application for data storage and for preservation purposes. In this experiment, the objective was to identify what quantity amount of user input stored on the application memory. This is the information recovered from only the RAM when the application is closed and the user is still interacting with the systems, as assumed, user may be doing something else on Windows computer system.

4. Result: Quantitative assessment

In this section, the quantitative assessment of forensically relevant data was extracted from the volatile memory content of PowerPoint 2007, Outlook Email 2007 and Internet Explorer 7.0. User input was made once on the applications and while application has been closed, user is still interacting with other applications. As shown in Table 2, this investigation focused on the identification of user input and the amount of data recovered from the volatile memory (RAM).

In Outlook Email, the application level information was found stored over time and dispersed in continuous block of evidence. The amount of evidence recorded shows that this application can retain information as on the memory for some time even when the application was closed and the systems is still running, and when user is still interacting with the system. User input recovered from the Internet Explorer, revealed that the application level information was stored for as much longer period on the memory of the application. Again, the amount of relevant data found on this application shows that information can be retained longer in the memory when the application has been closed and when user is still interacting with the system. PowerPoint application revealed the amount of user input stored on the memory. As recovered from the allocated memory, little information can be retained for short period of time when the application has been closed and user is still interacting with the system.

Three approaches to the amount of quantity information recovered from the memory of these applications were designed for the purpose of this research.

Firstly, *the Mean Evidence in Repetition* is the average number of repeated evidence extracted and found in the physical memory of an application. This process can be used to match the original character of word information entered by the user and with the extracted memory dump strings processes. Then, it searches to count the identical characters that appears repeatedly in the memory allocated to that application.

Secondly, *the mean percentage of evidence found* is calculated based on the number of character of the original user input and the extracted memory dump string data. This character was counted as it was stored in the memory and found relevant to the evidence searched for. Where evidence is not found, it resulted in zeros. The percentage of evidence found is the total numbers of evidence found without zeros.

Thirdly, *the mean evidence in a continuous block* is the length of evidence found in a continuous block in the physical memory of an application. The length of the character is 10 in a row. This character length was determined by the process of matching the original user input information with the extracted memory dump strings data. This process counts the lengths of identical character numbers of evidence found in the memory, where evidence was allocated and then, matched the evidence found with the original user inputs information. It was discovered that the percentage amount of evidence found in PowerPoint is 25% being relevant evidence that was found on repetitions in the volatile memory. This information is stored in continuous block. The original user input contains slides of texts with commas, semi-colon, and full stop. Outlook Email 2007 recorded 53% of relevant data and as repeated in the memory. This information is stored in continuous block of the allocated memory with a paragraph of text with commas, semi-colon, full stop and special character. Moreover, we have large amount of evidence stored over time in Internet Explorer 7.0 applications with a remarkable amount of 71% of evidence recorded as dispersed in the memory. Memdump strings of user input were extracted from Internet Explorer 7.0 application. This information can be termed as application level information. The forensically relevant data was assessed to quantify what user is typing on the application, what user has been doing and what user is using the application for.

5. Future Work

In the future, we will investigate the quality of user input that can be recovered from Windows application memory using some commonly used English words.

6. Conclusion

In this research, we have presented the extracted user input found in the physical memory of Windows applications. This approach was based on how much data can be recovered when images were captured at 30 minutes, when the application was closed, but user was interacting with the system. We assumed user may be doing something else. Specifically, we have laid emphasis on the percentage amount of relevant evidence found, the repeated evidence and user input found in continuous block of evidence. This approach describes the process of securing digital evidence that is stored in the physical memory. This experiment involves memory dumping, data conversion of evidence into strings processes, and extraction of relevant user input. This information may be used as evidence in the court of law.

7. References

[1] DFRWS.(2007) 'Digital Forensic Research Workshop', http://www.dfrws.org/2007/challenge/index.shtml. (26 March 2010).

[2] EnCase. (2011) 'EnCase Guidance Forensic Software'. http://www.guidancesoftware.com/forensic.htm. (11 April 2010)

[3] FTK®. (2011) 'AccessData Digital Forensic Toolkit Software', http://accessdata.com/products/computer-forensics. (07 February 2010)

[4] Huebner Ewa, Bem Derek, Szezynska Magdalena, and Solomon Jason, (2007) 'User Data Persistence in Physical Memory', *Journal of Digital Investigation*, vol. 4, no. 2, pp. 68-72.

[5] Andrew Case. Andrew Cristina. Lodavico Marziale. Golden G. Richard. Vassil Roussev, (2008) 'FACE: Automated digital evidence discovery and correlation', *Journal of digital investigation,* , vol. 5, no. 8, pp. 65-78.

[6] F. Olajide and N. Savage, (2011) 'Forensic extraction of user information in continuous block of evidence', *International Conference of Infornomic Society (i-Society),* London, 2011, pp. 476-481.

[7] Solomon DA. Russinovich ME, (2009) '*Microsoft Windows internal Covering Windows Server 2008 and Windows Vista'*, 5th ed. Washington, USA: Microsoft Press.

[8] Brian Carrier., (2005) '*File System Forensic Analysis'*, 1st Ed. Donnelley Mark M. Pollit, Ed. Crawfordsville, United States: Addison Wesley Professional.

[9] Carvey H. Kleiman D., (2007) 'Windows Forensic Analysis Incident Response and Cybercrime Investigation Secrets', *International Journal of Digital Investigation*, vol. II, no. 2, pp. 23-78.

[10] Olajide F. Savage N., (2009) 'Application Level Evidence From Volatile Memory', *Journal of Computing in Systems and Engineering*, vol. II, no. 2, pp. 70-78.

[11] Olajide F. Savage N., (2011) 'Dispersal Of Time Aspect Of Information Stored On Physical Memory', *International Conference on Cybercrime Security and Digital Forensics*, Glassgow.

[12] Spafford Eugene and Carrier Brian, (2006) 'Categories of Digital Investigation Analysis Techniques based on the Computer History Model', *Digital Forensic Research Workshop (DFRWS)*, vol. 1, no. 2, pp. 1-28.

[13] Petron Nick and Walters AAron, (2007) 'Volatools: Integrating Volatile Memory into Digital Investigation Process', *Journal of Digital Investigation*, vol. II, no. 2, pp. 26-35.

[14] Sutherland Iain, Evans Jon, Tryfonas Theodore, and Blyth Andrew, (2008) 'Acquiring Volatile Operating System Data Tools and Techniques', *ACM SIGOPS Operating Systems Review - The ACM Digital Library published by Association for Computing Machinery*, vol. 42, no. 3, pp. 65-73.

[15] Garcia G.L., (2007) 'Forensic Physical Memory Analysis: An Overview of Tools and Techniques', *TKK T-110.5290 Seminar on Network Security*, Helsinki, Finland, pp. 305-320.

[16] ManTech. (2008) 'ManTech Memory DD', http://www.mantech.com/msma/MDD.asp (8 March 2010)

[17] Msuiche. (2008) 'Msuiche.net Capture memory under win2k3/vista/Windows7 with win32dd/win64dd' win32dd.msuiche.net/ (22 November 2010)

[18] Nigilant32. (2006) 'Agile Risk Management, Nigilant32 - Windows Incident Response Tool'. http://www.agilerm.net/publications_4_.html (16 April 2010).

[19] Volatile Systems. (2006) 'The Volatility framework: Volatlile Memory Artifact Extraction Utility Framework', https://www.volatilesystems.com/ (12 April 2009).

[20] Walters AAron, Fraser Timothy, Petroni Nick, and Arbaugh William, (2007) 'FATKit: A Framework for the Extraction and Analysis of Digital Forensic Data from Volatile System Memory', *Journal of Digital Investigation*, vol. 3, no. 4, pp. 197-210.