ISSN No:-2456-2165

Forensic Investigation Tool for Volatility Framework

Dilushinie Narmada Fernando
Dept. Information Technology
SLIIT Academy
Colombo, Sri Lanka

Dr. Lakmal Rupasinghe
Dept. of Information Systems Engineering
SLIIT
Colombo, Sri Lanka

Abstract:- According to many research findings, the volatile memory has become a more vital space used by attackers and malicious users to store data that needs to be covert from others and avoid reverse-engineering. Since most incident response teams seldom study the volatile memory and lack the knowledge and equipment needed to extract information from it, there is plenty of data to back this up. Furthermore, the recent development of malicious codes can remain in the memory without affecting the physical disk. Therefore security analysts must prioritize and investigate the volatile memory as an important component rather than being following traditional logic thinking that the malicious users will only look into hard disk storage. The Volatility Framework is an open-source and free set of tools to analyze computer memory. This framework provides many options for data analysis in different aspects as a command-line interface. This makes complications for forensic analysts to memorize and use the tools and plugins. This research offers a GUI and extensions for the Volatility Framework, which simplifies the usage and provides a time-saving approach as the investigators do not want to memorize long command sequences.

Keywords:- *Volatility Framework; Forensic Investigation.*

I. INTRODUCTION

The world is relying increasingly on computing day by day. Firewalls, encryption, and signature or heuristic scanning are some of the digital defences used by businesses. Furthermore, planned attacks on power grids, infiltration of military data centres, and theft of trade secrets from both public and private enterprises have been detected worldwide. As a result, information security experts must be able to detect, respond to, and report on various types of computer system intrusions.

Evidence of compromise has a higher probability of being written to the memory than to the hard drive. Because malicious code must be loaded in memory to function, memory has a significant potential for containing malicious code from infection, in whole or in part, even if it is never written to disk.

The volatility tool is used for forensic investigation of memory dumps from compromised computers. The tool is command-line driven and expects the user to provide a large amount of information to guide the search. The GUI based tool that we have created allows the user to conduct investigations on the volatile memory using the volatility framework in a more user-friendly manner. It also allows reporting to be done in a more human-readable manner. Additionally, it allows the detection and listing of all the malware related information from the memory dump.

II. LITERATURE REVIEW

Many relatively new techniques have been developed to recover and deconstruct the information that can be retrieved from volatile memory, however, many forensic analysts are unaware of or do not make use of these assets because this is a relatively new and rapidly growing discipline. Many amounts of evidence essential to a forensic inquiry can be found in volatile memory, including passwords, cryptographic keys, and other data. It's critical to have the expertise and tools necessary to retrieve that data, and this capability is growing more important as hard drive encryption and other security features make traditional hard disk forensics more difficult.

A. PC Architecture

The underlying hardware in a digital environment determines which instructions may be performed and which resources can be accessed. Investigators who are able to recognize a system's unique hardware components and the influence those components can have on analysis are in the ideal position to perform a successful investigation. Operating systems are responsible for dealing with the low-level specifics of the processor, peripherals, and memory hardware present in a system.

During an investigation, you search for objects that suspected software or users may have brought into the digital environment, and you try to figure out how the digital environment altered as a result of those objects.

ISSN No:-2456-2165

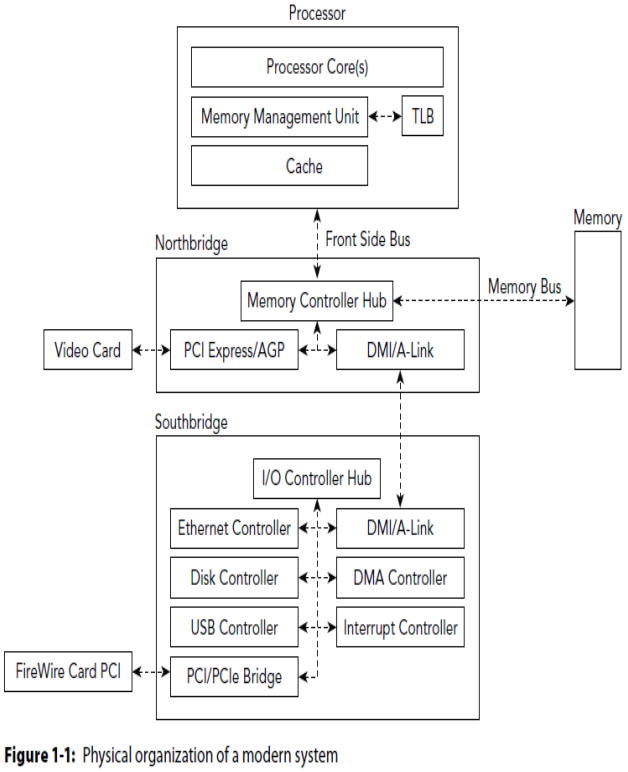

Figure 1-1: Physical organization of a modern system

Fig. 1. Physical Organization of a Modern System

Reading from the main memory takes a long time compared to reading from the CPU's internal memory. As a result, current systems make use of many layers of fast memory, referred to as caches, to assist compensate for the difference. For the Memory Management Unit (MMU) translation table, the processor employs a specific cache known as the Translation Look-aside Buffer (TLB).

The MMU is a hardware unit that converts the address requested by the processor into its main memory equivalent. On older PCs, the front-side bus was used to connect the CPU to the Northbridge (memory controller hub), while the memory bus was used to connect the Northbridge to the main memory. Devices (such as network cards and disk controllers) were connected to the Northbridge via the Southbridge, or input/output controller hub, which shared a single memory and CPU link. In today's computers, most memory controller hub features are now incorporated into the processor, which increases performance and lowers costs.

B. Volatility Framework

Volatility is a memory forensics platform that allows you to recover digital artefacts from volatile memory (RAM) dumps for incident response and malware analysis. You may use Volatility to get details about running processes, open network sockets and network connections, dynamic link libraries (DLLs) loaded for each process, cached registry hives, process IDs, and more. The extraction procedures are performed independently of the system under investigation, but they provide full transparency into the system's runtime state.

The Volatility framework includes commands for listing open network connections, printing a list of open DLL files, printing the memory map associated with the memory dump being analyzed, printing a list of open files associated with a process, and much more. Its capacity to reconstruct and write down an executable sample from its associated process is one of its most fascinating aspects (especially for malware investigators), [3-5].

C. How Volatile Memory Works

It will be required to have a rudimentary understanding of how volatile memory works in an attempt to comprehend some of the information in the next sections, [1-2]. This section will aim to lay out the key principles needed to comprehend how memory works in general on both Windows and Linux.

To begin, anything that the kernel, or central operating system, uses and requires in order to run will be represented as an object in both Linux and Windows. Every object utilized by the kernel on Windows has an OBJECT_HEADER, which is a structure that contains information about the object. When objects are saved in memory on Windows, the kernel can store them in one of two ways. The kernel has two sets of memory that are arranged like heaps and are commonly referred to as pools. There is a paged pool, which will hold the majority of the data, and a non-paged pool, which will hold only the most important objects that the kernel will need to access regularly. If the machine's actual physical memory is running low, any data in the paged pool can be transferred to a file on the hard disk, [3]. Because the process and thread objects are so critical and often accessible, they are stored in the non-paged pool, which means that the analyst will have access to all processes that are active at the moment physical memory is recorded, [4].

III. METHODOLOGY

This GUI tool is built on Java as an executable. The volatility framework will be wrapped around Java. We have chosen Java for the GUI because of its cross platform support and investigators have a choice to use any system. The analysis part of the module uses the volatility framework to extract information acquired from memory dumps.

A. Investigate Memory Dump

TABLE I. INVESTIGATE MEMORY DUMP

Use Case 1:	**Investigate Memory Dump**
Description:	Investigator will issue required commands to gather evidance
Actors:	1. User 2. GUI client tool
Preconditions:	Memory dump loaded in to the GUI tool
Outcomes (Post conditions):	Outputs related to the issued commands in a GUI format
Common Normal Course:	Proper user-friendly outputs
Alternate Courses:	Error in analysing memory dump

Use Case 1:	Investigate Memory Dump
Functional Requirements	Select proper commands from the list and a working system.
Business Rules	Investigator selects proper commands
Assumptions	Investigator selects proper commands

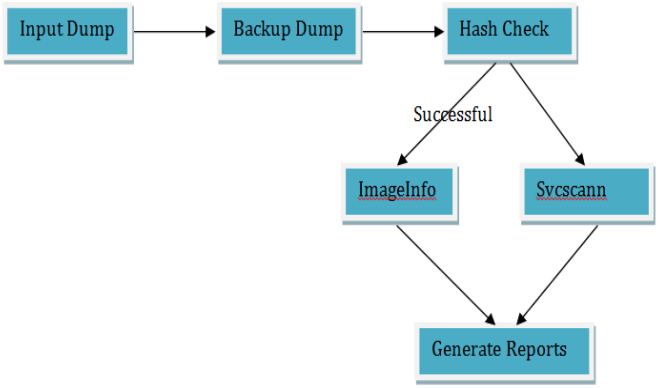

Fig. 2. System Diagram

B. Implemetation of the System

When the user inserts a memory dump into the system, it will take a backup dump before investigating the security issues. Then the system will calculate the hash value to check the integrity. Using "Image Info" button it will display the information or the properties of the image. The following "Fig. 3" screenshot shows the main view of the volatility GUI tool created. If the dump image is not selected, when we click on the 'Integrity Check' button, the following error message will be displayed as "Fig. 4".

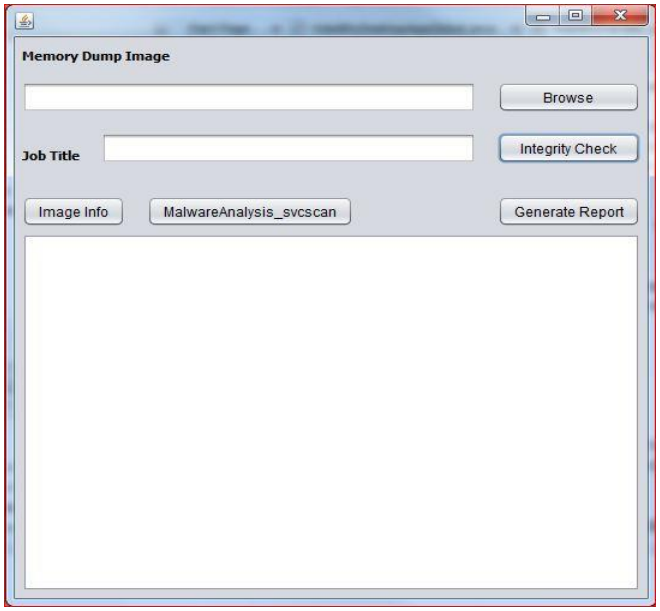

Fig. 3. Main Interface

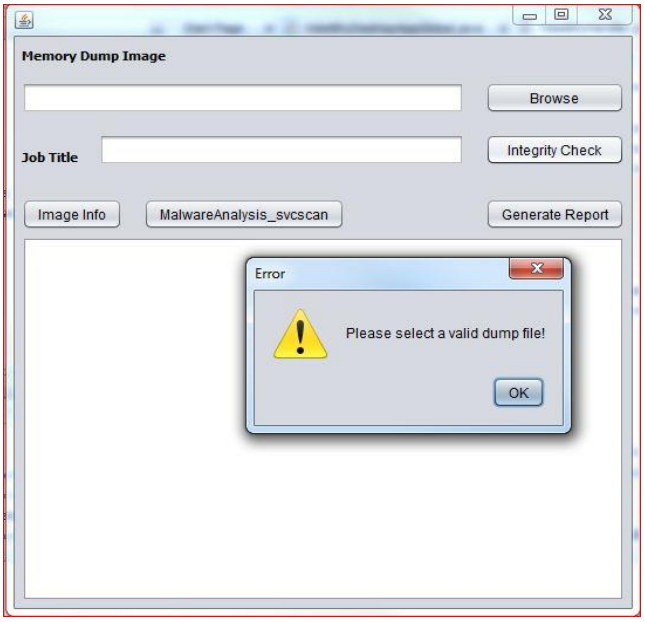

Fig. 4. Validate the Browse Text field

We can click on the 'Browse' button to select the memory dump that we need to analyse.

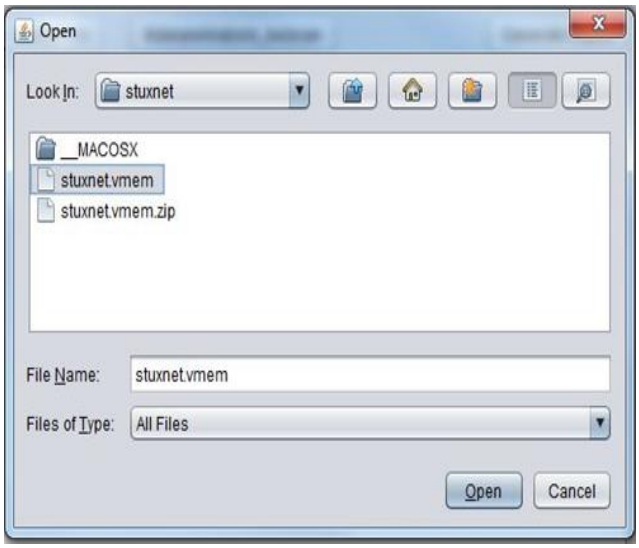

Fig. 5. Browse for Dump file

If the 'Job Title' is not entered, when we click on the 'Integrity Check' button the following error message will inform the user about that fact.

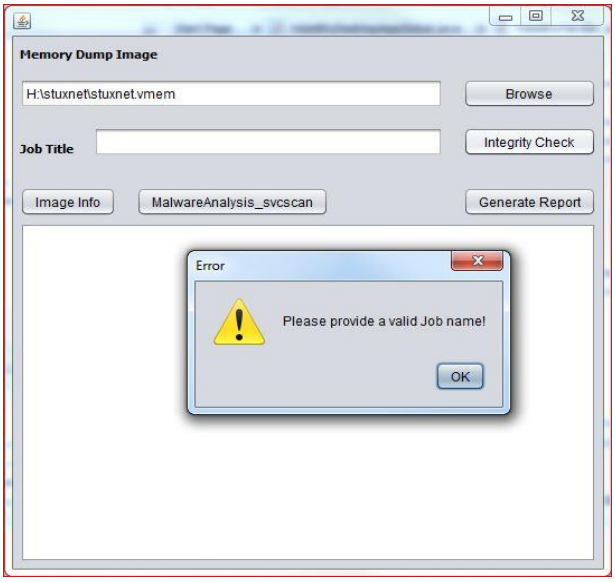

Fig. 6. Validate Job Title Text field

Fig. 7.

The 'Integrity Check' button gives the following output, when the memory dump and the Job title are successfully entered. When the memory image is uploaded to the tool, it takes a copy of the dump and saves it in another location. The software then generates MD5 hash value for both copies of the dump and compares them to check integrity. The following screenshot shows the output we get when the file is copied and the hash is generated.

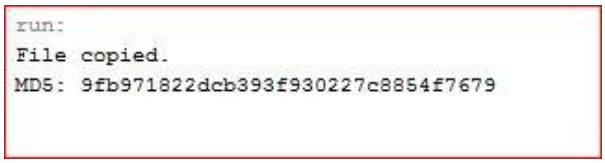

Fig. 8. MD5 Hash Value

If the hash values match successfully the following message will be displayed to the user.

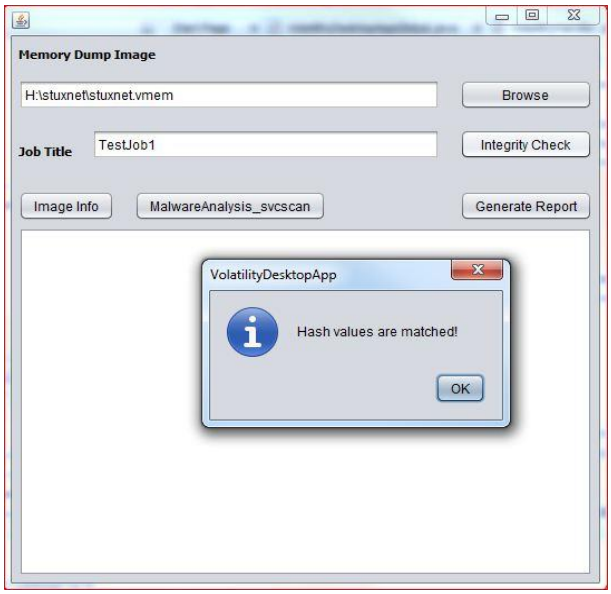

Fig. 9. MD5 Hash Values matched successfully

When we click on the 'Image Info' button the imgeinfo plugin of the volatility framework will be called. Imageinfo provides a high-level overview of the memory sample we're looking at. This command is most widely used to describe the operating system, service pack, and hardware architecture, but it also includes other essential information like the Delay Time-Based (DTB) address and the time the sample was taken.

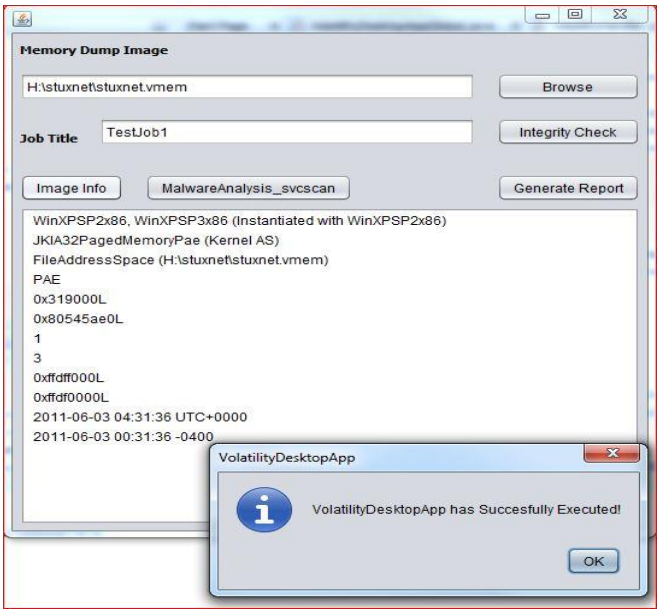

Fig. 10. Output of ImageInfo Command Under Image Identification

When we click on the 'MalwareAnalysis_svcscan' button the svcscan plugin of the volatility framework will be called. Volatility is the only memory forensics framework that can list services on a live machine without having to use the Windows API. To see which services are registered on your memory image, run the svcscan command. Each service's process ID, name, display name, type, and current status are all displayed in the output. It also displays the registered service's binary path, which is an EXE for usermode services and for kernel mode services will be a driver name.

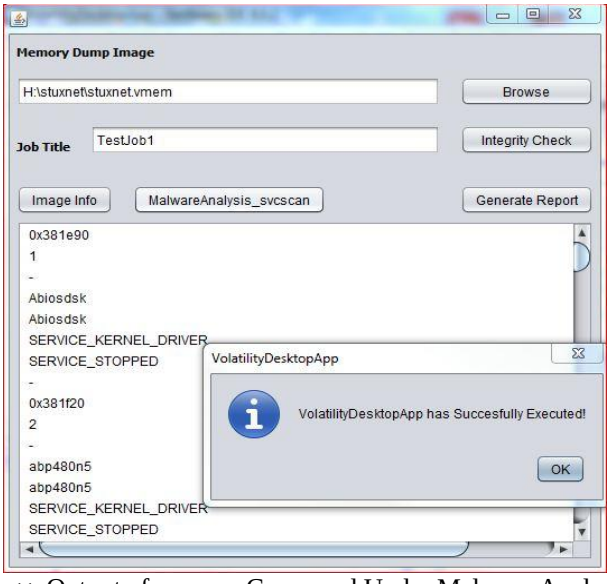

Fig. 11. Output of svcscan Command Under Malware Analysis

IV. CONCLUSION AND FUTURE WORK

The digital investigative procedure requires volatile memory analysis. While there are numerous tools for capturing and analyzing live memory, it is still a relatively recent endeavour in the field of digital forensics. Volatility is one of the greatest free open source applications for examining RAM in 32-bit and 64-bit systems. It can analyze Linux, Windows, Mac OS X, and Android systems. It can examine raw dumps, crash dumps, VMware dumps (.vmem), virtual box dumps, and a variety of other types of dumps.

This Volatility Framework's GUI provides a user-friendly interface for digital forensic investigators. A simple and easy-to-install GUI replaces the CLI interface, which might be a barrier to investigators using Volatility. Aside from improving usability, the GUI expands the Volatility Framework's functionality in several ways. Complex command sequences can be condensed to a single button click.

Overall, the new GUI makes memory dump analysis easier and more accurate. The tool is future-proof in terms of new plugins or even new toolkits because it can be readily extended. A few existing Volatility plugins and commands need to be included, while some new functionalities are still being developed. Furthermore, during this project, fresh concepts were generated that may be worth considering. Batch processing of many images is one example. Automated virus scans of the extracted processes would be one use for such a feature. Another example is to provide data correlation in a graphical manner, which can help people comprehend the relationships between processes, network connections, and other things.

REFERENCES

[1]. Russinovich, M. E., & Solomon, D. A. (2005). Windows Internals (4th ed.). Redmond, Washington: Microsoft Press.
[2]. Burdach, M. (2005, July 9). An Introduction to Windows memory forensic. Retrieved October 25, 2008, from http://forensic.seccure.net/
[3]. Carrier, B. (2005). File System Forensic Analysis. New York, New York: AddisonWesley.
[4]. Schuster, A. (2006). Searching for processes and threads in Microsoft Windows memory dumps. The Proceedings of the 6th Annual Digital Forensic Research Workshop. 3, pp. 10-16. Digital Investigation.
[5]. Logen, S., Höfken, H. & Schuba, M. (2012). A GUI and Extensions for the Volatility Framework. *Simplifying RAM Forensics*,.
[6]. Volatile Systems, Llc (2006-2011), *Memory Forensics | Malware Analysis | Volatile Systems*. Available from: https://www.volatilesystems.com/
[7]. Volatility Wiki, "Volatilitycommand reference", Available from: https:// code.google.com /p/volatility /wiki/CommandReference