Extraction of Memory Forensic artifacts from Windows 7 RAM Image

Sunu Thomas
Dept. Of Computer Science,
TocH Institute of Science and Technology,
Cochin, India
sunuthomas@live.in

Sherly K.K
Dept. Of Information Technology,
TocH Institute of Science and Technology,
Cochin, India
shrly_shilu@yahoo.com

Dija S
Resource Centre for Cyber Forensics,
Centre for Development of Advanced Computing (CDAC),
Trivandrum, India
dija@cdac.in

Abstract— **Memory Forensics is a novel and fast growing field in computer forensics, providing access to volatile information unavailable from a disk image. The memory forensics commenced when malware writers began reducing their footprints on the victim's hard disk and instead started storing crucial information within the machine's Random Access Memory. Windows 7 claims to be the most secure version of windows yet, thereby causing the forensic investigations a tedious one. Identification of kernel variables, running processes and extraction of process memory from a Windows 7 memory dump is more difficult when compared with previous versions of Windows. This paper discusses various windows kernel data structures and provides a methodology for extracting the list of running processes from 32-bit and 64-bit Windows 7 memory dump. The paper also presents a method for recovering process memory of running processes from a Windows 7 memory dump.**

Keywords—CR3 register; Memory Forensics; Windows 7; KPCR; EPROCESS;

I. INTRODUCTION

Forensics is the process of using scientific knowledge for preserving, collecting, examining and reporting evidence to the court. It deals primarily with the recovery and analysis of latent evidence such as files on hard drive, DNA evidence from blood stains and finger prints on left over window [1]. Digital Forensics [2] is a new discipline in computer science, which was developed due to the rise in unauthorized activities in computer and information systems. Computer attacks and cyber-crimes are continually on surge due to the high dependency of the society and business on information systems and the fact that e-commerce and online business became an essential part of today's business world.

Computer Forensics is one of the branches of digital forensic science pertaining to legal evidence found in computers and digital storage media. The main goal of computer forensics is to examine digital media in a forensically sound manner thereby identifying, preserving, recovering, analysing and presenting facts and opinions about the acquired information. Investigations are performed on static data (i.e. acquired images) rather than "live" systems. Conventional computer forensics dealt mainly with the capture and analysis of data on permanent storage devices, predominantly hard disks [3]. Live Forensics in contrast collects data from running systems instead of permanent storage devices. Memory forensics is a live forensic approach which involves collecting forensically sound evidence from the physical memory (RAM) of a running system. In Memory Forensics, first a bit stream copy of memory is acquired into a file. The acquired dump is analysed to find out crucial evidence from the suspect's system. Memory Forensics is very essential because it has become the location of choice for attackers and other malware writers to store hidden information.

II. BACKGROUND

Memory Analysis work came into existence in the year 2005 for extracting evidence from volatile memory accurately and reliably [4]. Initially physical memory was captured just to retrieve strings like e-mail addresses, passwords and IP addresses. Due to the significant advancement in the area of memory forensics, different memory acquisition and analysis tools are available in the current market.

The Microsoft Windows kernel maintains different data structures in order to keep track of its resources. To enumerate processes, threads, handles and other important artifacts, the maintained list and tables are to be walked through, based on a global variable available in the memory. In Burdach's approach [5], to enumerate the list of running processes, the PsActiveProcessHead is used as a starting point which is a pointer to the start of the kernel's list of Executive Process(EPROCESS) structure. Unfortunately, finding these kinds of variables are very difficult. Burdach finds EPROCESS structure through searching strings such as "smss.exe" and "csrss.exe" in the memory image and hard-code the offset address to get addresses of concerned global variables such as PsActiveProcessHead, PsLoadedModuleList etc. The main disadvantage of the approach of finding kernel global variables is that, the hard-coded address of each variable is a constant only for a particular version of Windows and even a single hotfix can change it and let the tool useless. So finding these addresses is not an easy job for forensic developers. In [7] a reliable method, an approach based on Kernel Processor Control Region (KPCR) for locating the kernel EPROCESS block for listing the running processes is proposed. This approach for identifying KPCR is an efficient

978-1-4673-5758-6/13/$31.00 © 2013 IEEE

Authorized licensed use limited to: Universiti Kebangsaan Malaysia. Downloaded on January 24,2021 at 07:28:22 UTC from IEEE Xplore. Restrictions apply.

method for XP, Vista, 2003 Windows operating system, but not suitable for Windows 7 32-bit and 64-bit operating system. This is because, in Windows 7 the KPCR and Kernel Processor Control Block (KPRCB) are not located in these addresses. Therefore the physical address of KPCR structure cannot be found by searching the location of a specific binary string in the memory image file. An advanced KPCR approach is used for tracking KPCR in Windows 7 memory dump. But in all cases, even after identifying KPCR, valid EPROCESS structure cannot be tracked. A detailed step by step procedure for identifying these structures is not available in the forensic world.

This paper describes a methodology for listing running processes, loaded DLLs and extracting process memory of running processes. Section III describes the method for identifying the KPCR structure in 32-bit and 64-bit Windows 7 memory dump. The method of locating the PsActiveProcessHead and thereby tracking list of running processes is also being described in section III. The details of retrieving information from Process Environment Block (PEB) structure and the procedure for extracting process memory of running processes are described in the following sections.

III. MEMORY FORENSICS IN WINDOWS 7

A. Identification of KPCR

KPCR is an internal data structure in which the windows kernel stores per - processor information. There are as many KPCR in the system as there are CPUs. KPCR contains an embedded data structure called KPRCB, which contains scheduling information such as current, next and idle threads scheduled for execution on the processor, time accounting information, statistics for the processor etc. The details of different memory structures of Windows are available in [6].To obtain KPCR, the method described in [7] and [8] can be adopted. Address translation of 32-bit and 64-bit address are described in [9].

The size of KPCR structure in case of Windows 7 32- bit operating system is 0x120.The Self field of KPCR structure, contains the virtual address of KPCR and CurrentPrcb field contains the virtual address of KPRCB (located at offset 0x120 of KPCR). In order to track KPCR structure in 32-bit Windows 7 memory dump, search for two contiguous four bytes strings which are both greater than 0x80000000 and their difference is 0x120. The virtual address obtained from the contiguous strings is converted to physical address. Check whether the obtained physical address from Self field is equal to the physical address of KPCR(Offset of Self- 0x1C) and also the physical address from CurrentPrcb is equal to the physical address of KPRCB(Offset of CurrentPrcb+0x100). Similarly in case of 64-bit Windows 7 memory dump, the only difference is that, the search string is eight bytes instead of four bytes. For identifying KPCR structure in 64-bit Windows 7 RAM image, search for two contiguous eight bytes strings which are both greater than 0xFFFF000000000000 and their difference is 0x180. The KdVersionBlock field of KPCR structure is null in case of 32-bit and 64-bit Windows 7 memory dump.

In Fig.1 the value of Self field of KPCR structure is 0x807C7000, on converting this virtual address to physical address we get 0x2CA85000. Similarly on converting the virtual address 0x807C7120, available in CurrentPrcb field, we get the physical address of KPRCB.

In Fig.2 the value of Self field of KPCR structure is 0xFFFFF80003041D00, on converting this virtual address to physical address we get 0x3041D00. Similarly on converting the virtual address 0xFFFFF80003041E80, available in CurrentPrcb field, we get the physical address 0x3041E80, the actual physical offset of KPRCB.

Once the KPCR structure is identified the CR3 register value can be obtained by adding 0x1D0 to KPCR in case of 64-bit Windows 7 memory dumps and 0x40C in case of 32-bit Windows 7 memory dumps. CR3 register value plays a crucial role in address translation. The _KTHREAD structure of KPRCB structure will not always point to a valid EPROCESS structure and therefore will not be able to list the running processes.

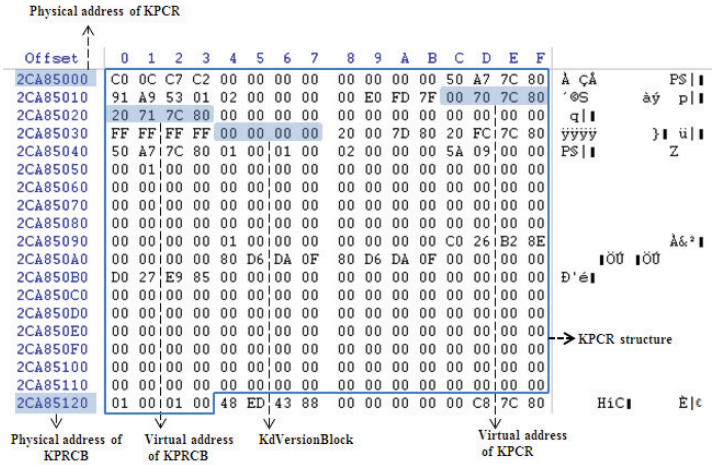

Figure 1. Hex View of KPCR structure in 32- bit Windows 7 memory dump

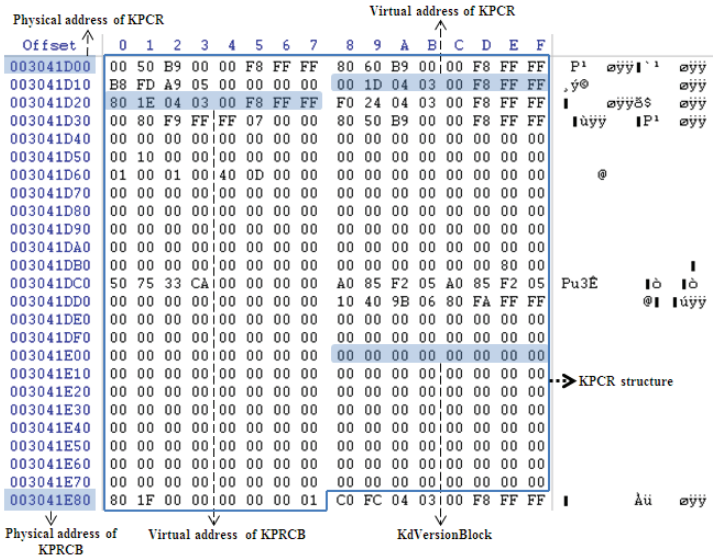

Figure 2. Hex View of KPCR structure in 64- bit Windows 7 memory dump

Authorized licensed use limited to: Universiti Kebangsaan Malaysia. Downloaded on January 24,2021 at 07:28:22 UTC from IEEE Xplore. Restrictions apply.

B. *Extraction of Running Process list*

Memory resident malwares may not leave any footprints in the system. Such information can only be retrieved by analyzing physical memory dump. So the extraction of running process list from memory dump is very essential for finding evidence from against suspect. The ActiveProcessLink field in the EPROCESS structure is used to link all the processes in the system. PsActiveProcessHead is the starting point to this list. In order to identify the PsActiveProcessHead, first search for hex string "4B44424740030000" in Windows 7 memory dump. The hex string is searched mainly to track the _DBGKD_DEBUG_DATA_HEADER64 structure and the _KDDEBUGGER_DATA64 structure in memory dump.

In case of 64-bit Windows 7 memory dump, the virtual address of PsActiveProcessHead is obtained by subtracting 0x1E320 from the virtual address of PsLoadedModuleList. The address translation process varies based on, whether the Physical Address Extension (PAE) is being enabled or not. Information regarding this can be identified from the PAE Enabled field of the _KDDEBUGGER_DATA64 structure as shown in Fig.3 and Fig.4. If the PAE Enabled field is set, it is PAE Enabled otherwise Non-PAE. In case of 32-bit Windows 7 memory dump with PAE Enabled field set, the virtual address of PsActiveProcessHead can be obtained by subtracting 0x7938 from virtual address of PsLoadedModuleList. In 32-bit Windows 7 memory dump with Non-PAE, subtract 0x6F18 from the virtual address of PsLoadedModuleList to get the virtual address of PsActiveProcessHead. Once the virtual address of PsActiveProcessHead is obtained, find the physical address of PsActiveProcessHead and traverse through the list entry to find all the running processes as in Fig.7. While traversing the list entry, one of the major functionalities is the virtual to physical address conversion. A detailed-flowchart describing the steps to be followed for extracting list of running processes is shown in Fig.5 .The hex view of the EPROCESS structure is depicted in Fig. 6.

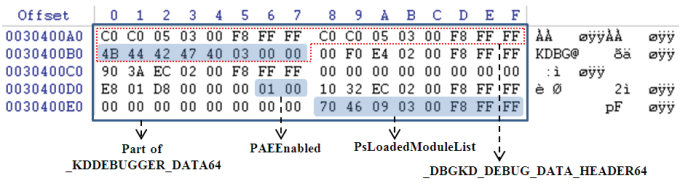

Figure 3. Hex View of _KDDEBUGGER_DATA64 structure in 64-bit Windows 7 memory dump

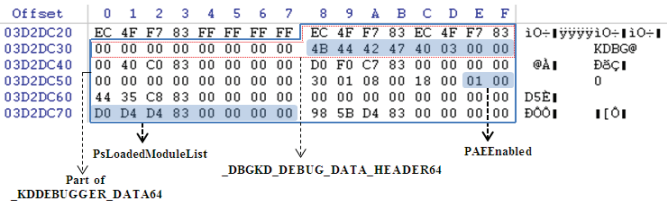

Figure 4. Hex View of _KDDEBUGGER_DATA64 structure in 32-bit Windows 7 memory dump

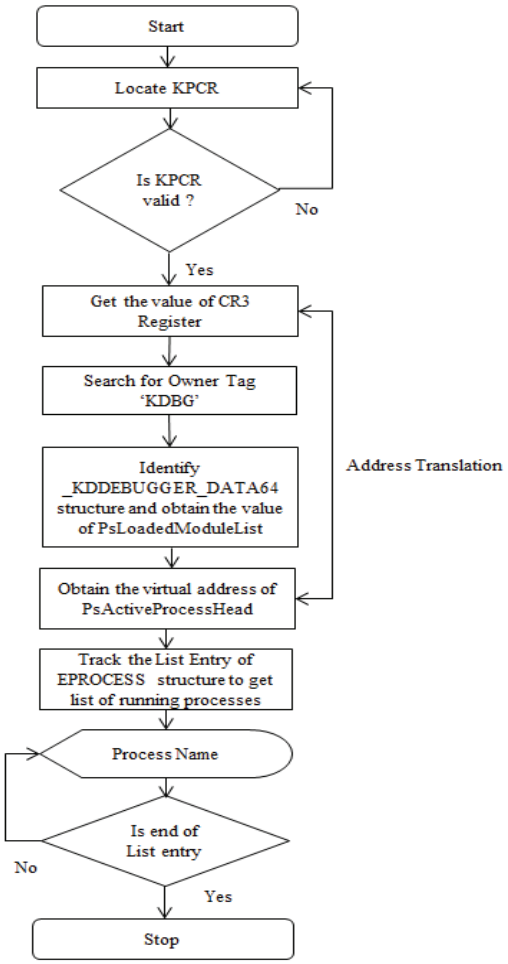

Figure 5. Flowchart depicting steps for extracting running process list

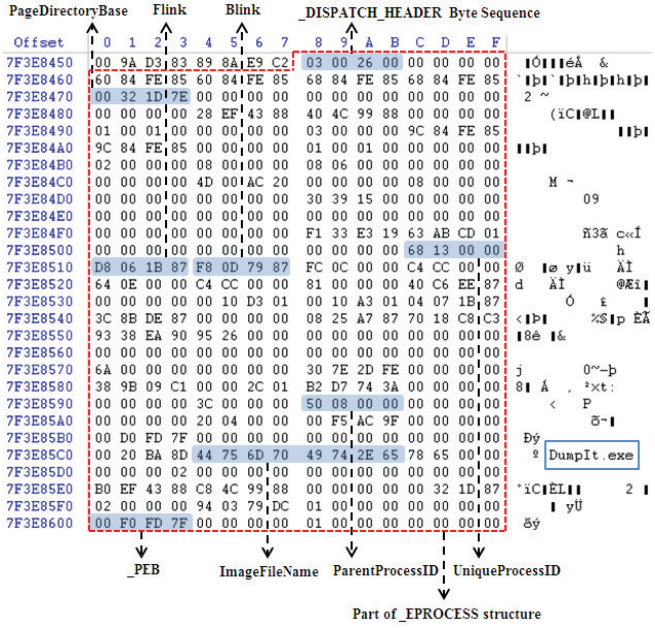

Figure 6.Hex View of EPROCESS structure in 32- bit Windows 7 memory dump

Authorized licensed use limited to: Universiti Kebangsaan Malaysia. Downloaded on January 24,2021 at 07:28:22 UTC from IEEE Xplore. Restrictions apply.

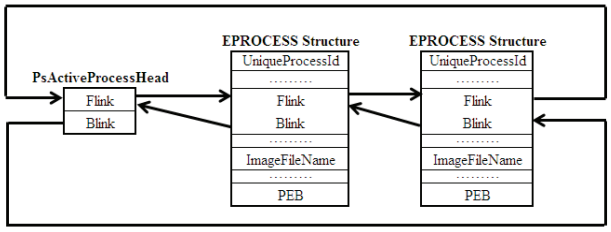

Figure 7. EPROCESS structure pointed from PsActiveProcessHead

C. *Identifying PEB and its related structures*

Process Environment Block (PEB) pointed by the EPROCESS structure is one of the important elements of a process and contains a great deal of information. PEB is a per-process structure. The DirectoryTableBase (DTB) of EPROCESS structure acts as the starting point of address translation of PEB, for converting virtual address of PEB to physical address. In order to get the list of the loaded DLLs of each process, the first step is to locate the PEB structure of EPROCESS.

Once the _PEB structure is identified, information such as Number of Processors, OS Major Version, OS Minor Version, OS Build Number, Session ID etc. can be obtained. The Ldr field at offset 0x00C of PEB, points to a structure called _PEB_LDR_DATA which contains the list entry of loaded DLLs. List Entry of loaded DLL starts from the structure _LDR_DATA_TABLE_ENTRY. From the FullDllName field of _LDR_DATA_TABLE_ENTRY, the length and the location of DLLName can be obtained.

The _RTL_USER_PROCESS_PARAMETERS pointed by the _PEB structure contains information such as the DllPath at offset 0x038, ImagePathName at offset 0x040, Command line at offset 0x040 and Windows Title at offset 0x070. All these are of type _UNICODE_STRING which consist of Length, MaximumLength and Buffer pointer. The location pointed by the Buffer field contains the required data. The relationship between the EPROCESS and other structures is shown in Fig.8. The hex view of _RTL_USER_PROCESS_ PARAMETERS and _ PEB structure are shown in Fig. 9 and Fig. 10.

The information regarding the operating system of the suspects system can be obtained by acquiring the OS Build Number field of _PEB structure. Table 1 shows OS Build Number corresponding to different operating system.

TABLE I. WINDOWS BUILD NUMBER IN DIFFERNET OPERATING SYSTEM

OS Version	OS Build Number
Windows Vista	6000(RTM)
	6001(SP1)
	6002(SP2)
Windows Server 2008	6001(RTM)
	6002(SP2)
Windows 7	7600(RTM)
	7601(SP1)
Windows 8	9200

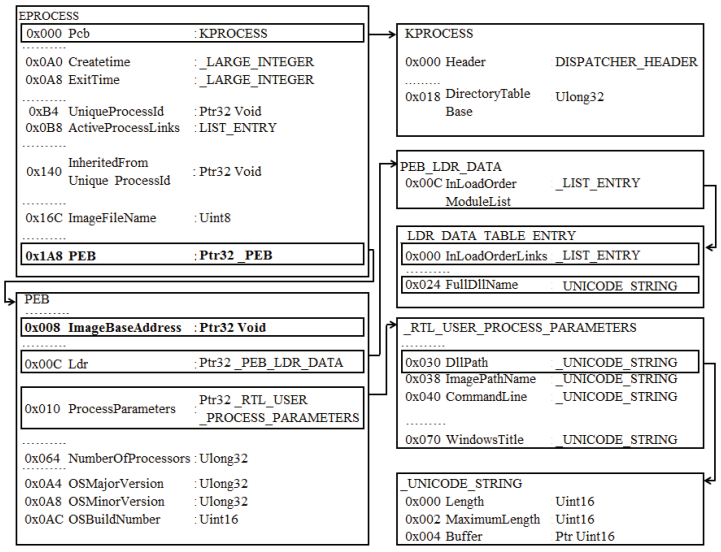

Figure 8.EPROCESS and its related structures in 32-bit Windows 7 memory dump

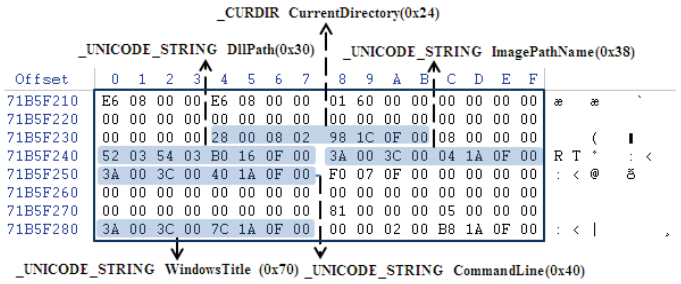

Figure 9. HexView of _RTL_USER_PROCESS_PARAMETERS structure in 32- bit Windows 7 memory dump.

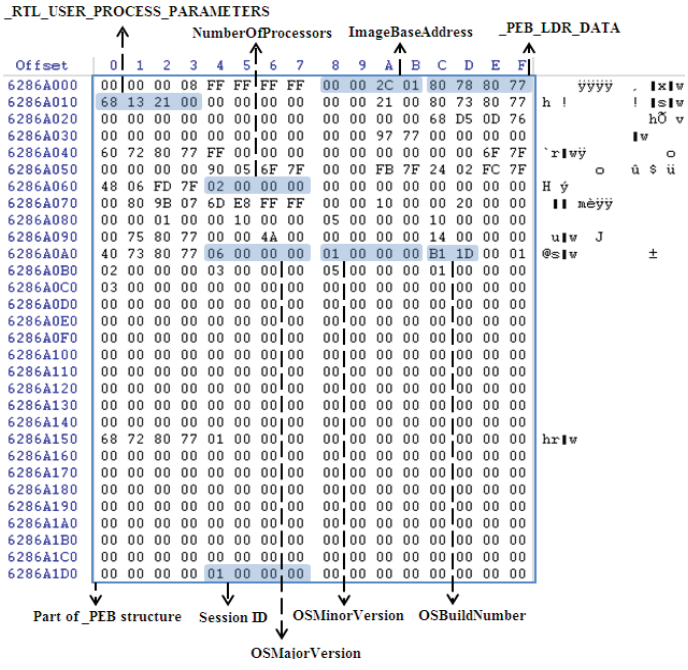

Figure 10. HexView of _PEB structure in 32-bit Windows 7 memory dump

Authorized licensed use limited to: Universiti Kebangsaan Malaysia. Downloaded on January 24,2021 at 07:28:22 UTC from IEEE Xplore. Restrictions apply.

D. *Extracting Process Memory*

Process memory extraction is very important as it helps the investigator to get an idea behind the application, by analyzing the process dump using a reverse engineering tool.Using a disassembler, executable in machine language format can be converted to assembly language equivalent. It generates assembly language corresponding to the executable in portable executable format. From the assembly code generated by the disassembler the analyst can extract the functionality implemented inside the executable.

For acquiring process memory of running processes, first we need to locate the _PEB structure of each process. The _PEB structure is located at offset 0x1A8 of EPROCESS structure. The ImageBaseAddress at offset 0x008 of _PEB structure in 32-bit Windows 7 contains the virtual address, where the executable of the process is being located. Using DTB in EPROCESS structure the virtual address in ImageBaseAddress is converted to physical address.

At beginning of the memory page pointed by the Image Base Address, a signature 'MZ' will be present. MZ stands for Mark Zbikowski which is the magic signature of executable.

A detailed flowchart describing the process of extracting process memory is shown in Fig 11. The hex view of part of extracted process memory page is shown in Fig. 12.

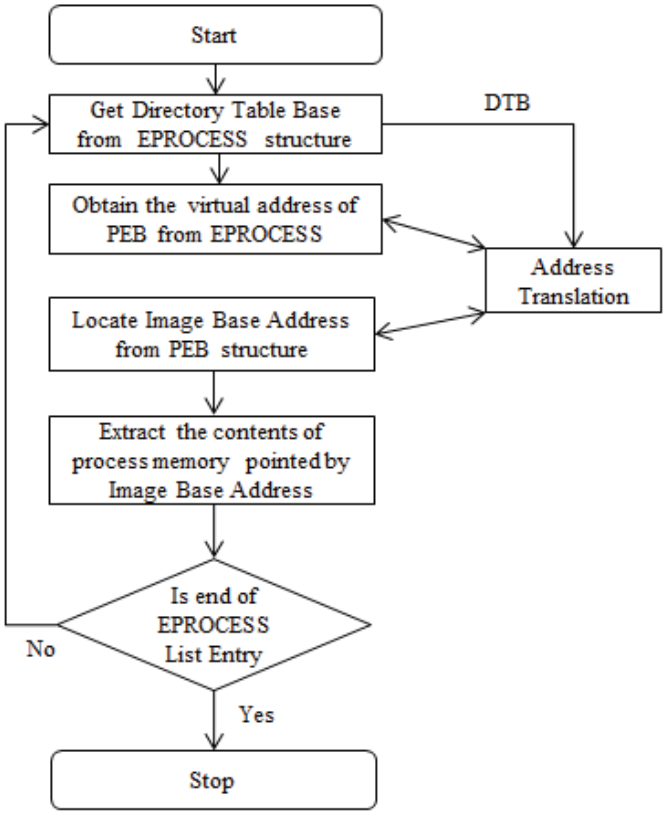

Figure 11. Flowchart depicting steps for extracting process memory

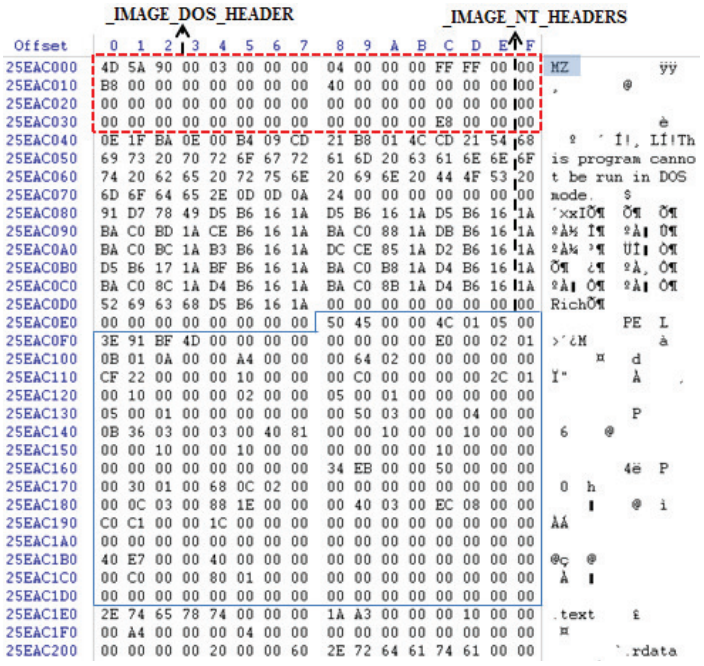

Figure 12. HexView of process memory of a sample process Wiint.exe

IV. EXPERIMENTAL RESULTS

Experiments were conducted on both 32-bit and 64-bit memory dump collected from Windows 7. The GUI results of test done on Windows 7 memory dump are shown in Fig.13, Fig.14 and Fig.15.

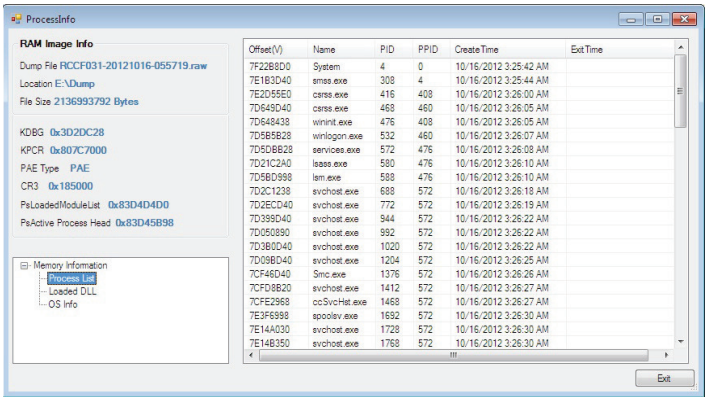

Figure 13.Snap shot of Running Process list from a 32-bit Windows 7 memory dump

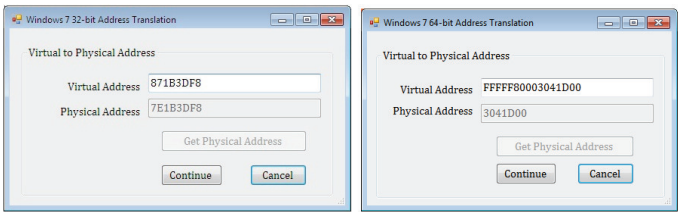

Figure 14.Address Translation in 32 -bit and 64- bit Windows 7 memory dump

Authorized licensed use limited to: Universiti Kebangsaan Malaysia. Downloaded on January 24,2021 at 07:28:22 UTC from IEEE Xplore. Restrictions apply.

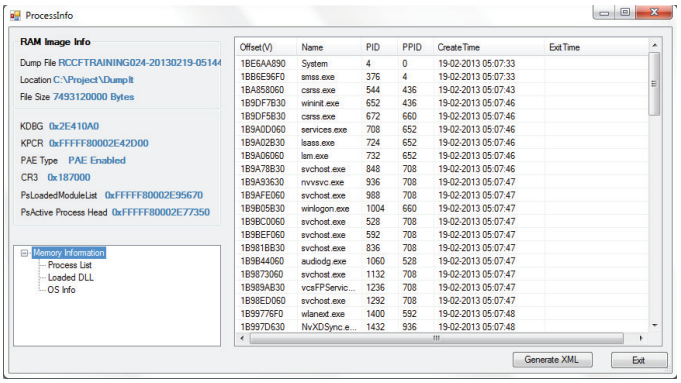

Figure 15. Snap shot of Running Process list from a 64-bit Windows 7 memory dump

V. Conclusions And Future Work

As cybercrimes are becoming more prevalent, the forensic analyzers are very eagerly hunting for victims in the cyber world. The memory analysis works are highly dependent on the operating system. Crucial information regarding the activities going on in a running system can be identified by analysing the physical memory dump collected from the suspect's system. In this paper methods for acquiring important information such as KPCR, CR3 register value, process details, loaded dlls and process memory from Windows 7 memory dump are discussed. The extraction of running process list and address translation are supported by snapshots in Section IV. More detailed work is to be done in the area of Windows 7 64-bit operating system for obtaining more crucial information.

Acknowledgement

This research is supported by Resource Center for Cyber Forensics (RCCF) Department, Centre for Development of Advanced Computing (CDAC), Trivandrum.

References

[1] "Computer Forensic" , US-CERT.[Online].Available: http://www.us-cert.gov/reading_room/forensics.pdf.

[2] Khidir M. Ali "Digital Forensics Best Practices and Managerial Implications". In: Fourth International Conference on Computational Intelligence, Communication Systems and Networks. 2012, pp:196-199.

[3] Wang Lianhai, Zhang Ruichao, Zhang Shuhui. "A Model of Computer Live Forensics based on Physical Memory Analysis". In: Proceedings of 1st International Conference on Information Science J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[4] DFRWS 2005 Forensic Challenge, http: //www.dfrws.org /2005/ index. shtml.

[5] Mariusz Burdach, "An introduction to Windows Memory Forensic". [Online].8(3), pp 25.Available: http://cfile1.uf.tistory.com/attach /1143B9404EFAA38E1010DA.

[6] Moonsols Dveloper Network MSDN, http://msdn.moonsols.com.

[7] Ruichao.Z, Lianhai. W, Shuhui. Z, "Windows Memory Analysis Based on KPCR". In: Fifth International Conference on Information Assurance and Security, vol. 2, pp.677-680,2009.

[8] Shuhui.Z, Lianhai.W, Ruichao.Z, Qiuxiang.G, "Exploratory Study on Memory Analysis of Windows 7 Operating System". In: 3rd International Conference on Advanced Computer Theory and Engineering , Vol. 6, pp. V6373-V6377, 2010.

[9] Mark Russinovich, David A. Solomon, Alex Ionescu: Windows Internals (6th Edition),Part 2, Microsoft Press, 2012.

Authorized licensed use limited to: Universiti Kebangsaan Malaysia. Downloaded on January 24,2021 at 07:28:22 UTC from IEEE Xplore. Restrictions apply.