Virtual Machine Memory Forensics

Alvin Huseinović and Samir Ribić, IEEE member

Abstract — **Physical memory can contain various data such as user passwords, encryption keys, web browser activity and other traces interesting for forensic analysis. Virtual machine physical memory is usually presented as a file on a host operating system. In this paper, the obtaining and analyzing of the virtual machine memory dump are presented.**

Keywords — **forensic analysis, memory dump, snapshot, virtualbox, VMware, Volatility framework;**

I. INTRODUCTION

A traditional digital forensic investigation procedure assumes that a computer, if found on the crime scene, is plugged out of electrical power and taken to a laboratory in order to perform offline analysis of data found on the hard drive. This traditional procedure has drawbacks, and it is obvious that data that was in physical memory is lost.

Physical memory can contain some important data such as encryption keys, open network sessions, etc. If the computer is set to off, the data are lost and cannot be retrieved for later analysis. With growth of virtual machine market, it became popular to use virtual machines for different tasks. As a consequence, virtual machine forensic recently took more place in collecting forensic data.

If a suspected person uses a virtual machine for illegal tasks, it may be too hard or almost impossible to identify the virtual machine data by using traditional methods. The reason for this may be that standard tools for computer forensics are not ready and prepared for recognition of specific data structures produced by various virtualization hypervisors.

Virtualization hypervisor is software that enables communication between virtual machines and host operating systems. It provides management of memory page tables and partition scheduling, either by direct virtualization of I/O devices or by delegating requests to special I/O partitions [1]. There are many tools that provide virtualization on the market. In this paper Oracle VirtualBox and VMware workstation memory dumps are collected and analyzed.

In order to proceed with detailed explanation of how to collect non-persistent data from virtual machine, it is important to give an explanation of steps that have to be performed in order to gain appropriate results.

By using VMware workstation it is possible to choose two natively supported options to save a physical memory State and data. The first one is preserving appropriate *vmem* file (by using suspend option) and the second option is to take a virtual machine snapshot. File vmem is virtual machine paging file. The snapshot consists of multiple files: vmdk (the redo log), vmsd (snapshot metadata) and vmsn (snapshot state). [2]

To analyze the data contained in suspend and snapshot files it is necessary to perform data transformation using a third party tool. It is possible to use a volatility framework to create raw dd-style memory dump. DD is software that provides support for creating a bit-by-bit copy of files, but it is also used for copy of hard drives, physical memory and other memory devices.[3] VirtualBox supports similar options, but after reading publicly available official documentation, we realized that it does not save complete physical memory to disk after performing any of these actions. To perform gathering data from VirtualBox memory abstraction, one needs to use a debugger that is implemented in VirtualBox. By using the debugger it is possible to preserve memory data in ELF64 format, that can be processed and converted to dd-raw style image.[4]

The rest of paper shows that standard and built-in options can be used to collect physical memory data from virtual machines that are running in VMware and VirtualBox virtualization environments. It is also shown that data such as list of processes can easily be obtained by using the Volatility framework.

II. PRESERVING AND OBTAINING VOLATILE DATA FROM VMWARE WORKSTATION AND VIRTUALBOX

The data collection process started with creation of VMware and VirtualBox virtual machines with fresh installation of Windows XP system. Every machine has the same set of started applications. Before the procedure of saving the data is started a task manager application is started, we had to obtain a list of processes on both virtual machines. This can be compared to lists obtained from different versions of files produced by natively supported options in virtualization software.

The Fig. 1 depicts the list of processes running on VMware virtual machine. After the screenshot is taken, we also made the snapshot of the virtual machine. The second step performed was sending virtual machine to suspended state.

A. Huseinović is with the Faculty of Electrical Engineering, University of Sarajevo, Zmaja od Bosne bb, 71000 Sarajevo,Bosnia and Herzegovina;(phone: +38761378528, e-mail:ahuseinovic@etf.unsa.ba)

S. Ribić is with the Faculty of Electrical Engineering, University of Sarajevo, Zmaja od Bosne bb, 71000 Sarajevo, Bosnia and Herzegovina;(phone: +38733250700, e-mail:sribic@etf.unsa.ba)

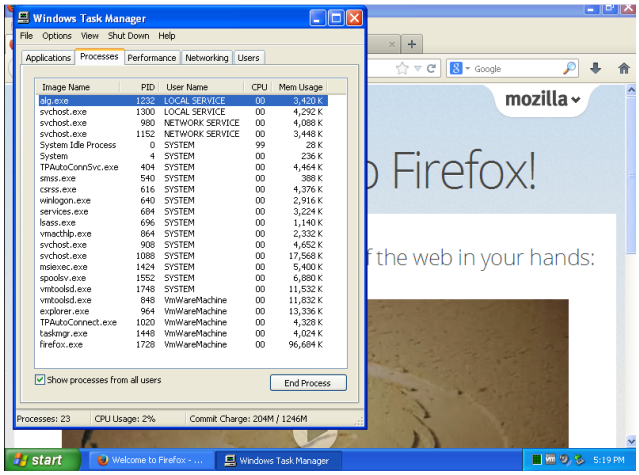

Fig. 1. List of processes running on VMware virtual machine

After the machine is sent to the suspended state, cryptographic MD5 sums of files are taken. They are vmem (memory file), .vmsn (VMware snapshot file) and .vmss (VMware saved state). The first, *.vmem* file, contains data that are saved by performing suspend option. The snapshot option creates the .vmem file which has generic name with *snapshot* prefix and appropriate *.vmss* files and *.vmsn* files are recreated by taking snapshot. It is important to say that snapshot option enables the investigator to change data and run virtual machine without changing the original data in virtual machines. [5] This makes virtual machines easier to manage and handle than traditional computers. Volatility framework enables adequate processing of the data saved by performing these actions. [6]

Processing the first *.vmem* files in Volatility framework gave the list of the processes shown on the Fig 2.

.vmem suspend process list - Notepad

File Edit Format View Help

Name	PID
alg.exe	1232
svchost.exe	908
svchost.exe	980
svchost.exe	1088
svchost.exe	1151
svchost.exe	1300
System	4
TPAutoConnSvc.exe	404
smss.exe	540
csrss.exe	616
winlogon.exe	640
services.exe	684
lsass.exe	696
vmacthlp.exe	864
spoolsv.exe	1552
vmtoolsd.exe	1748
explorer.exe	964
TPAutoConnect.exe	1020
taskmgr.exe	1448
firefox.exe	1728

Fig. 2. List of the processes running on the VMware virtual machine obtained from *.vmem* suspend state file

The same list was obtained from the second *.vmem* file created by taking a snapshot.

On the other hand, VirtualBox virtual machine files do not save memory data in the manner of VMware. In order to perform raw data acquisition it is necessary to start the virtual machine with *-dbg* option. This option enables user to create raw data physical memory image. The option was enabled on the virtual machine and in the case it is disabled, additional tools that runs on host operating system have to run and support finding virtual machine traces in order to collect physical memory data. On the Fig. 3 we can see the list of processes running on VirtualBox guest.

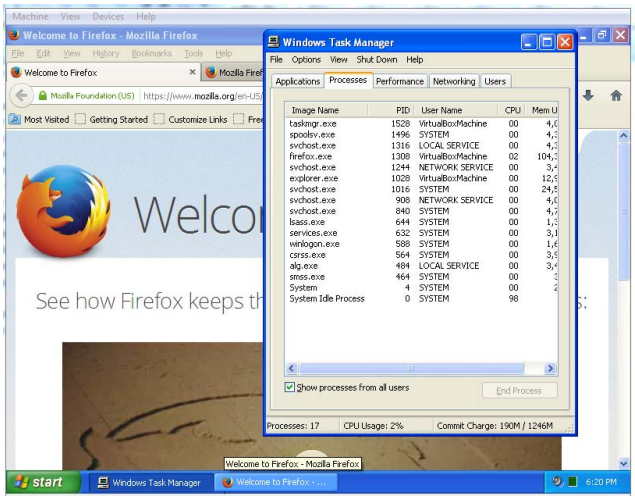

Fig. 3. List of processes running on the VirtualBox virtual machine

Processing the dump file is natively supported by Volatility framework and the list of processes that were running on the system is showed on figure 4.

vbox process list - Notepad

File Edit Format View Help

Name	PID
taskmgr.exe	1528
spoolsv.exe	1496
firefox.exe	1308
svchost.exe	1316
svchost.exe	1244
svchost.exe	1016
svchost.exe	908
svchost.exe	840
explorer.exe	1028
lsass.exe	644
services.exe	632
winlogon.exe	588
csrss.exe	564
alg.exe	484
smss.exe	464
System	0

Fig. 4. List of the processes running on VirtualBox virtual machine obtained from native dump

In absence of VirtualBox *-dbg* option, it is necessary to analyze the host operating system memory to identify virtual machine traces and catch the hypervisor that enables virtual machine hosting. There are some tools that enable hypervisor detection and tracing virtual machines running. One of them is a plug-in developed for volatility framework called *actaeon*.[7]

The use of Actaeon as a tool is described below.

Actaeon is used to obtain the list of processes running on VirtualBox and VMware virtual machines. Obtaining the data from virtual machine memories is done by analyzing memory dump of the virtual machine host.

To obtain the process list from virtual machines running on the host operating system, physical memory dump of the host system is taken. This tool enables the user to find

data structures, and memory offsets of the VMCS.

The steps that are necessary to perform in order to obtain virtual machines memory dump are as follows.

- First we need to take a physical machine memory dump, using methods described earlier-

- With Volatility framework we can detect if any virtualization software is running on it by obtaining the process list from memory dump

- After obtaining the appropriate process ID it is needed to find out if the machine that was running hypervisor has the processor that supports Intel Vt-x instructions

- If supported Intel processor family was used, analysis of the virtual machine address space is possible.

The test system was one with supported processors listed, so it was easy to get process lists of the two virtual machines running on the different hypervisors (VMware and VirtualBox). The process list was the same as one retrieved by using earlier described options.

The main disadvantage of taking physical machine memory dump is the fact, that user changes computer state and can easily do steps that may lead to irretrievable corruption of evidences.

III. FUTURE WORKS

The future work may be related to finding appropriate data structures in host memory that may be suspected of using hardware assisted virtualization so if user creates custom hypervisor that can be encrypted to at least determine the memory address space that belongs to hypervisor and hosted virtual machines. For better overview of possibilities for obtaining data from other virtualization hypervisors should be considered.

The presence of current popular hypervisors is more or less easy to detect in host systems if appropriate tools and knowledge is used. The question of encrypting virtual machines hypervisors in a memory is not considered at all. If criminals create hypervisor that is custom and that cannot be detected by regular tools, it is almost impossible to determine such data structure in memory.

IV. CONCLUSION

Forensic analysis of virtual machines is not an easy task. In this paper, simple cases of virtual machine memory dump analysis are shown.

The process list from VMware virtual machine was obtained in three different ways using the same tool. The first was obtaining the list from .vmem file created by the use of suspend option. The other way was to process .vmem and VMware snapshot files in Volatility framework. The third one was obtaining the list of processes from memory dump of the host operating system.

The process lists from VirtualBox virtual machine were obtained by running virtual machine with -dbg switch. The other was obtaining the list of processes from memory dump of the host operating system.

REFERENCES

[1] E. Hensbergen, “The Effect of Virtualization on OS Interference”, The Fourteenth International Conference on Parallel Architectures and Compilation Techniques: PACT05 , Sept,. 2005

[2] D. Barret, G. Kipper, "Virtualization and Forensics: A Digital Forensic Investigator’s Guide to Virtual Environments", Elsevier 2010, pp 146-147

[3] E.Siever, S. Figgins, R. Love, A. Robbins, “Linux in a Nutshell, A Desktop Quick Reference”, 6th edition, O’ Reilly, 2009, pp 105-106

[4] Oracle Corporation, “Oracle VM VirtualBox User Manual”, 2013, pp 200-201

[5] M. Hirwani, “Forensic Analysis of VMware Hard Disks”, master thesis, Rochester Institute of Technology, 2011, pp.8

[6] S. Mrdovic, A., Huseinovic, E., Zajko, Combining static and live digital forensic analysis in virtual environment. Information, Communication and Automation Technologies, 2009. ICAT 2009. XXII International Symposium on, vol., no., pp.1-6, 29-31 Oct.2009.

[7] M. Graziano, A. Lanzi, D. Balzarotti, Acteaon framework, available at: http://s3.eurecom.fr/tools/actaeon