A Brief Survey of Memory Analysis Tools

Zia Ur Rehman, Aneeq Ahmad and Shahzad Saleem
School of Electrical Engineering & Computer Science, National University of Science & Technology, Islamabad, 44000 Pakistan
Email: {14msiszrehman, 14msisaahmad, shahzad.saleem}@seecs.edu.pk
Received: Accepted: 7

Abstract

This paper covers five major tools used for memory forensics that would be helpful to scientific community and forensics researchers in determining which tools are best according to their requirement. From memory forensic analysis, it is very easy to judge about malware presence and behavior. This paper shows a brief survey of tool's attributes and their supported platforms. We have mainly focused to mention results on the basis of running process, dll's, drivers, registry data, event logs, web activity, services, Malware IOC(Indicators of compromise) analysis, network information, size of the tool, address translation etc. Investigators may choose one of the tools according to their requirements.

Keywords: Memory Forensics, forensic analysis, malware, behavior analysis, Malware IOC

Introduction

Memory forensics is one of the major steps in crime scene investigation. It involves collecting the image of the RAM and then analyzing its contents for various purposes like finding the traces of malware, acquiring the list of running process, network information, files, loaded DLL's, programs, retrieving passwords or to verify digital signatures of processes. It is pivotal for forensics investigators to make the correct choice of the tools for memory image acquisition and analysis because one wrong choice can lead to wrong diagnostics or loss of critical data needed for the forensics investigation, as RAM is volatile and once its contents are lost or overwritten, they can't be recovered. In order to help forensics investigators in this part of crucial investigation and analysis of delicate data, this paper is an effort to ease the forensics investigation process for crime scene investigators. RAM contains artifacts that are mostly hidden for an investigator like rootkits hidden in kernel level processes [1]. Mainly two methods are used to capture volatile memory. Hardware-based acquisition is more reliable and also difficult for an attacker to interrupt or sabotage. But most researchers make use of software based acquisition tools due to ease of availability and cost effectiveness. Hardware based tools relies on the use of hardware to mirror the RAM image. Special hardware is used in order to capture memory in this way. Whereas software based tools are hardware independent and are more popular due to ease of usage and efficiency. In order to collect information about the well-known tools multiple online resources are cited by us to help in our research. After acquisition; proper analysis of the captured data is very important to reach a proper conclusion in forensics investigation.

Literature Review

Various techniques and tools for acquiring volatile data and some commercial tools like Memdump, FATKit, WMFT, Procenum, Idetect, Volatility Framework, VAD Tools, F-Response, Encase and their capabilities are stated by[2]. He also explained in great detail the contents that are found in volatile memory like the list of processes, file activity, network information, passwords and malicious content.

When the drives are encrypted by strong cryptographic algorithm, memory forensics may be the only possibility to recover the cryptographic keys necessary to decrypt the hard drives [3].

Main memory acquisition tools have two major aspects; successful acquisition of main memory image and analysis of the acquired image. Which lead us to draw the conclusion that a success of a tool depends upon how well a tool fulfills those main memory image acquisition and analysis requirement[4]

The readable strings in the memory can provide critical insight about critical information like passwords and cryptographic keys in plaintext which can help us in looking for strings in the memory for suspicious URLs, keys and decrypted passwords[5].

The methodologies of how to extract processes and threads from main memory are explained by [6]. His research effort described how the memory dumps can be used to search for those fragments of memory that will represent system processes and threads. He also eloquently explained the circumstances under which the details of terminated processes can be retrieved from the system.

Our research also focuses on how memory forensics can help the investigator in retrieving useful artifacts from an infected computer. Research paper by [7] is very helpful in this regard. With the help of the malware samples, they demonstrated how a typical malware especially in its metamorphic forms can be analyzed. Online malware databases can also be utilized for the research in the domain of memory forensics.

Memory analysis comprises of two primary components; Kernel memory and userland memory. Kernel memory in windows consists of device driver information, operating system executables etc. The userland memory consists of individual processes and files that are loaded from disks etc. Furthermore he told that disk forensics is as important as memory forensics, and the importance of disk forensics cannot be ignored. They also discussed a methodology to reduce useless and redundant data in the memory, so that it can help in digital forensics process [8].

Mcdown et al has conducted a thorough research on the comparison of memory acquisition tools. They have devised malware attribute metrics based on their execution time, loaded DLLs, registry entries etc [9]. Our effort mainly builds on their work to give an insight on the forensics

DOI: http://dx.doi.org/10.24949njes.v10i2.474

abilities of the memory acquisition tools and how it can help the forensics examiner in getting useful information from a memory dump.

The research paper written by [10], was very helpful for us to attain the deeper understanding of the memory forensics processes. The major technique discussed by this paper was the analysis of call stack to extract sensitive information and application finger prints.

A comparison of image acquisition tools is explained by [11] and they analyzed imageinfo, kdbgscan, pslist, psscan, psxview by comparing the results to the volatility Framework. Our work is also inspired by their efforts.

Artifacts Recovered from Main Memory

Wealth of information can be recovered from main memory, like running processes, services, loaded DLL's , unencrypted information, passwords, MAC and IP address of computer, cryptographic keys, hidden malware processes, rootkits, readable strings, open files, network information, internet history, registry information, driver information etc. Recovering this information or acquiring a main memory image from a computer is like extracting a blueprint of the computer's execution state. Information recovered by memory forensics can be beneficial to an investigator to investigate a crime scene or analyzing an infected computer or recovering a recently deleted document or file or to know about the hidden processes running in the system or to retrieve a recently entered password etc. Some malwares like PowerShell malwares that don't leave their trace on the hard drive and are directly loaded in the main memory can easily be traced by the analysis of memory images and can be useful for the forensics examiner in malware analysis. The details of PowerShell malware can be found at [12]

Processes

Analyzing memory dumps of a computer can help an investigator to find the list of running processes. This information can help in detection of running programs. The running programs can give an insight the hidden processes and thus about a particular crime.

Services

Services are very important for a malware analyst. Many malwares, in order to gain persistence and to make themselves firmly rooted in the victim's machine, hide themselves as a service. In this context they can be auto-executed when the computer is restarted. Obtaining information about services can help an investigator to identify unknown startup programs or autostart locations.

Most malwares can be eradicated from the computer by removing their autorun entry from a computer and also their traces from windows registry. Memory dumps can help an investigator in determining the service by driving information from memory dumps.

Network Information

A lot of information can be provided by memory analysis about the network artifacts like messaging, emails, chatting, login information (usernames and passwords), network adapter information, IP information and internet history [13]. Moreover, we can also find out what processes are making outbound network connections and what information they are extracting from victim and to what IP they are transferring it. It can help us in network and URL analysis and it can also help us to find out the command and control centre of a malicious process which can lead us to determine the capabilities of malware.

DLLs

Microsoft Windows makes extensive use of DLLs (Dynamic Link Libraries) for different purposes like memory management or for the purpose of library importing. Different attacks can be conducted like DLL hijacking or spoofing. Such an attack forces a legitimate process to load a malicious DLL. Moreover, in windows, DLLs cannot be run directly; some programs (executable) are needed to load them. Windows provide the utility of SysWOW64/rundll32.exe (for 64 bit DLLs) and System32/rundll32.exe (32 bit DLLs) to provide support to run some standalone DLLs. Malicious DLLs loaded , cannot be detected easily by just looking at the windows processes or by traditionally detecting them from task manager. However, an analysis of RAM dump can provide an investigator a direct insight of the DLLs loaded in the main memory.

Sensitive Information

Many encrypted documents with strong passwords cannot be broken by traditional methods because of strong algorithms produced after decades of research and development. Finding the key can help in this scenario so memory forensics on the infected computer can help the investigator to reverse the encryption process.

Registry Information

Information retrieved about registry hives can provide important information about installed programs, user profile information, windows settings, driver's information etc. It can be helpful for the investigator in developing the profile of the suspect's digital life.

Injected Code

Code injection is widely used for the purpose of unauthorized access, attacking databases or installing malware, or sometimes for the purpose of denial of service attacks. Various techniques discussed can be used to prevent injected code from executing. Memory forensics techniques can be used to prevent code injection like runtime image hash validation; in which the hash of image loaded into the memory is compared with the expected hash of the software or process.

Hooks

Malware programmers or rootkit writers can employ the use of hooks to intercept function calls in the operating system. In order to reveal those invisible hook based activity, memory forensics can be used.

Unpacked Files

In order to conceal the payload information, hackers compress the payload. In this way the encrypted or compressed data cannot be read by the user except from the decryption routine programmed by the attacker at the user end. Protected programs or files like these can only be inspected by memory forensics and it can give the

investigator a huge forward leap in the investigation about the malware capabilities.

Test Environment

In order to test the memory analysis tools and their capabilities, we have made use of four different virtual machines in VMware.(Windows 7, Windows 8, Windows 8.1, and Windows 10) with following specification.

- Intel(R) Core(TM) i7 CPU L640 @2.13GHz , ~2.1 GHz with 2 GB RAM

Moreover we have tested them on standalone computing hardware as well, rather than testing them only on virtual environment. The details of computers involved in the tests are listed below.

- Intel(R) Core(TM) i5 CPU (M) 430 @2.27 GHz (4 CPUs), ~2.3 GHz with 4 GB RAM
- Intel(R) Core(TM) i7 CPU L640 @2.13GHz (4 CPUs), ~2.1 GHz with 4 GB RAM

The Support of tools for various platforms is stated in table 1. We have acquired the image of the memory using only FireEye Memoryze in .img format as acquisition was not our area of focus, so we have utilized this one .img file of memory for analysis by all the other tools and based upon their output we have listed their attributes. The complete screenshots of the tests are shown at [14].

We have also used a sample malware from an online source to test these tools on the infected system. The malware is "Keylogger.Ardamax". The download link of malware for testing and research is stated at [15]

Basic Analysis of the Selected Malware

We have analyzed the malware via SysInternals Suite, to identify malware IOC's, so that it would be latter helpful for us to verify tools whether they are identifying those same indicators or not.

The malware runs with an executable name DPBJ.exe, we have identified this process executable via Process Explorer of SysInternals Suite as mentioned earlier. The Process Explorer was unable to show the complete properties of the malware, like image verification, VirusTotal detection ratio, path of the file, or autostart location etc. So we turned to process monitor to investigate the malware further. After adding some filters to refine the search, the process monitor or Procmon.exe showed us that malware is taking screenshots and recording keystrokes and stores them in following locations.

a. C:\Windows\SysWOW64\<random no.>\<system date&time>.jpg (for screenshots)

b. C:\Windows\SysWOW64\DPBJ.001.tmp (for storing keystrokes)

c. C:\Windows\SysWOW64\DPBJ.exe (malicious executable)

There were also other files like key.bin and AKV.exe. The size of malicious file DPBJ.exe is 646KB.

Network analysis of the malware shows us that the malware sends the DNS request to the yahoo mail server (smtp.mail.yahoo.com). The malware after some time interval T uploads/emails the stored screenshots and keystrokes to its Command and Control Server (C&C Server) at IP address 98.139.211.125:587 which is the IP of Yahoo Inc. which reveals that the attacker is using Yahoo email to receive data from victim's machine using port 587.

In order to gain persistence in the victim's machine the malware also hides itself in windows registry as an autorun entry DPBJ Agent, in following location: HKLM\SOFTWARE\Wow6432Node\Microsft\Windows\ CurrentVersion\Run. This sample malware is very well-known and has a detection ratio of about 85.71% on VirusTotal [16].

Tools Used for Memory Analysis

Currently there are a huge number of memory acquisition and analysis tools. We have compiled a list of five major tools (FireEye Redline, FireEye Memoryze 3.0, Volatility, Rekall, Magnet Internet Evidence Finder) depending upon ease of usage and availability. We have tested the latest versions of these tools. At the end of this paper we have summarized our findings using a matrix with the tested malware attributes.

Reasons for Selection of Tools and Specific Attributes

Although a range of memory forensics tools are available in the market with a broad spectrum of capabilities, we have selected these we have selected these five tools only because they are well-known to most investigators and are very popular in the research community[1] [17] [18] [13] [11]. According to our findings; these free and open source tools cover broad range of attributes as compared to other tools. The system requirements of running these tools are low. Two of these selected tools are open-source (**Volatility and Rekall**), two of them are free (**Redline and Memoryze**) and one is paid (**Magnet Internet Evidence Finder**). The main reason for selecting redline is that it also shows malware risk index scores, which was not done by any other memory forensics tool (according to the best of our knowledge). Moreover the selected tools can work on multiple platforms.

The attributes of the tools were filtered in according to their importance in crime reconstruction, malware identification and analysis. These attributes are easy to extract and understand like processes details, running DLLs, malware risk index etc.

FireEye Redline 1.14

Redline by FireEye Inc. like many other memory analysis tools provides processes detail, registry data, file system activity and network information. Its unique feature is the analysis with respect to malware's indicator of compromise and rating of running processes according to MRI (Malware Risk Index). FireEye Redline, when integrated with FireEye HX series can also provide the triage of all the infected clients within the network.

In our test environment we have first used the .img file acquired by FireEye Memoryze and then we have tested it with FireEye Redline. FireEye Redline didn't provided us with the details from that memory image, like event information, persistence details, registry information, cookies details, route entries, prefetch information, windows services etc. But when image was acquired by FireEye Redline then it gave complete details.

We have tested this tool in both infected and clean environment and on multiple platforms like windows 7, 8, 8.1. FireEye Redline can also acquire memory image for analysis and it can also accept formats like .img, .dd and .raw acquired by other tools. The screenshot result of FireEye Redline after image acquisition and then opening the memory file is shown in the Figure 1 which is showing the information gathered during the forensics process. The malware process being tested is also shown in the Figure 1 with an arrow to the left.

FireEye Memoryze 3.0

Like FireEye Redline, Memoryze is also a memory forensics tool that can shed light on DLLs, EXEs, running processes, network information, driver information etc. Memoryze can also show list of printable strings in the memory and can verify digital signatures of the drivers, running processes and DLLs etc. Memoryze is a cross platform tool and can function adequately on MAC operating system also. Memoryze can acquire image and it can also analyze the image by viewing it in Audit viewer or MS-excel as show in Figure 2

The malware executable is pointed out with an arrow on the left. As mentioned in the section 2.2, we have used FireEye Memoryze to acquire the .img file.

Volatility Framework 2.5

Volatility is one of the most popular memory forensics frameworks and it contains a range of different functionalities. It is free and open source allowing the researchers to incorporate their plugins according to their requirement.

Volatility supports various memory formats and can help investigators in a variety of ways like to find malicious code, registry info, event logs, kernel memory analysis, executable file extraction, process information, networking information and internet history etc.[19].

As already discussed we are using a .img memory image acquired by FireEye Memoryze. The results of the analysis of the memory image by Volatility Framework are shown in Figure 3. The malicious processes can also be seen in the figure. Along with the network information is shown in the Figure 4.

Rekall 1.5.2 Furka

Rekall is an open source memory analysis tool. It has multiple user interfaces to help the users from basic to complex memory analysis. From basic analysis like finding the services information, process information to complex analysis like address translation from virtual to physical and dumping results in readable format etc.

In Rekall, image acquired is in .aff4 format and it can extract different requested results by using specific commands. The results of REKALL with .img memory image format are shown in the figure 5. The malicious process is also shown with an arrow on the left.

Magnet IEF

Magnet IEF collects and displays information about URLs, page titles, email information and web content like pictures, browser activity, web history etc. A screenshot of the output of the tool is shown in figure 6 from memory image of win8. The reason for the selection of Magnet IEF is that it is the only tool that focuses on retrieving the browser based artifacts from memory

Our paper is focused on the survey of some of the most popular memory forensics tools and we have identified selected attributes available in the tools as mentioned in table 2. The detailed results of each tool in screenshots is compiled by us on this link for further insight and in-depth information [14]. Salient test results of different tools with .img memory image format acquired from FireEye Memoryze are shown in table 3.

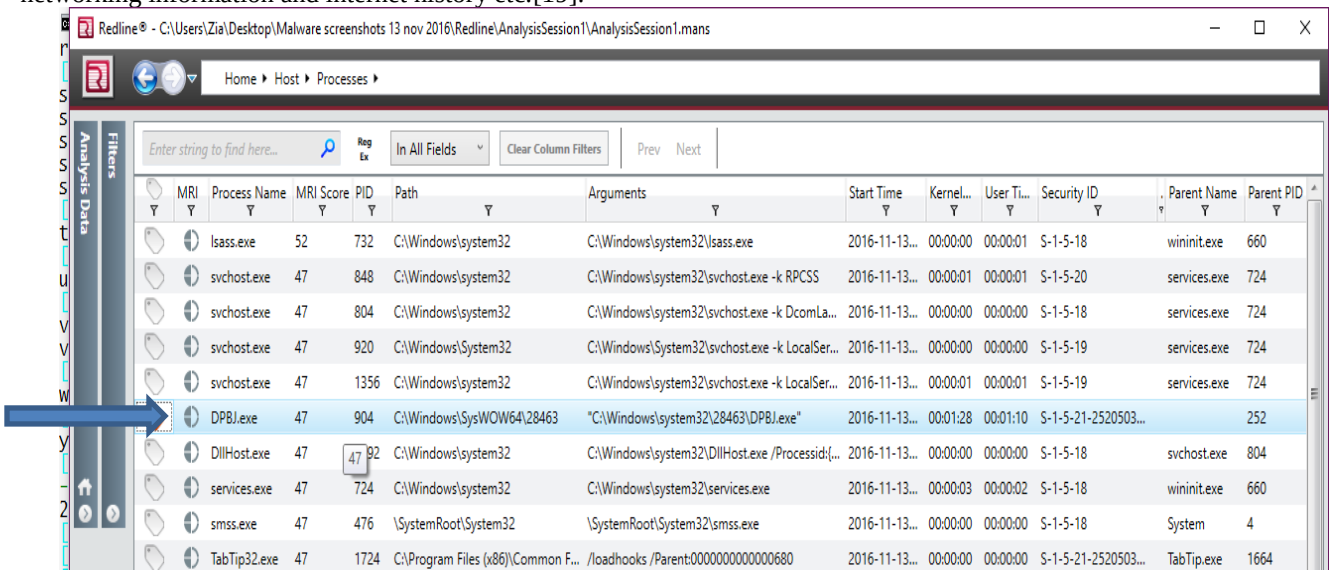
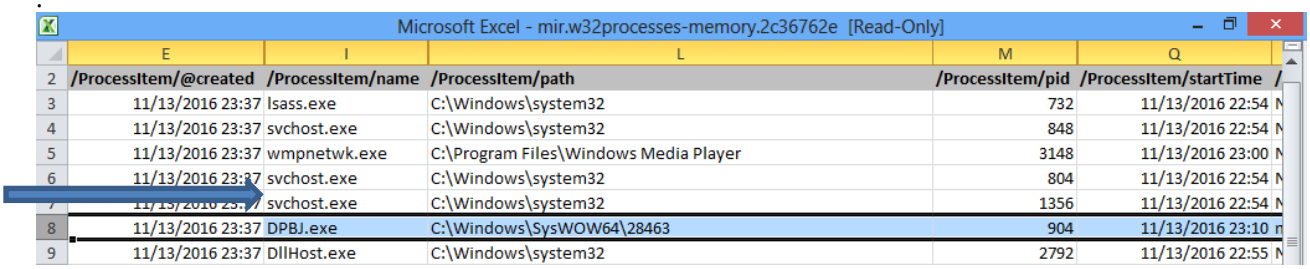
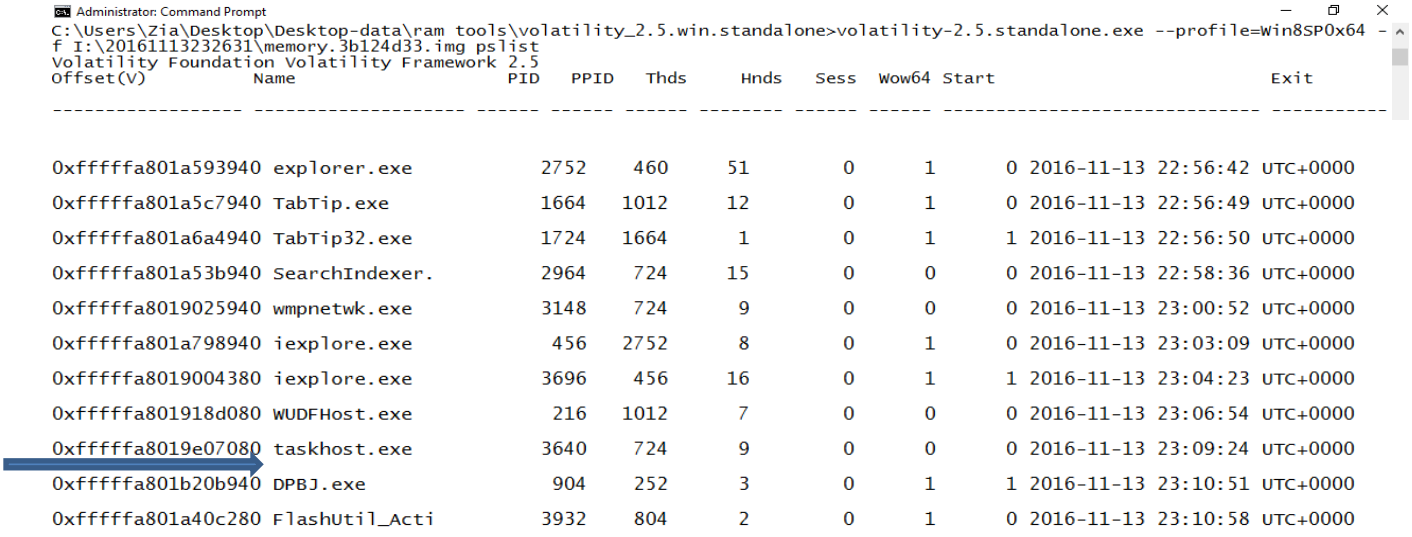


Fig. 1: Redline Process information



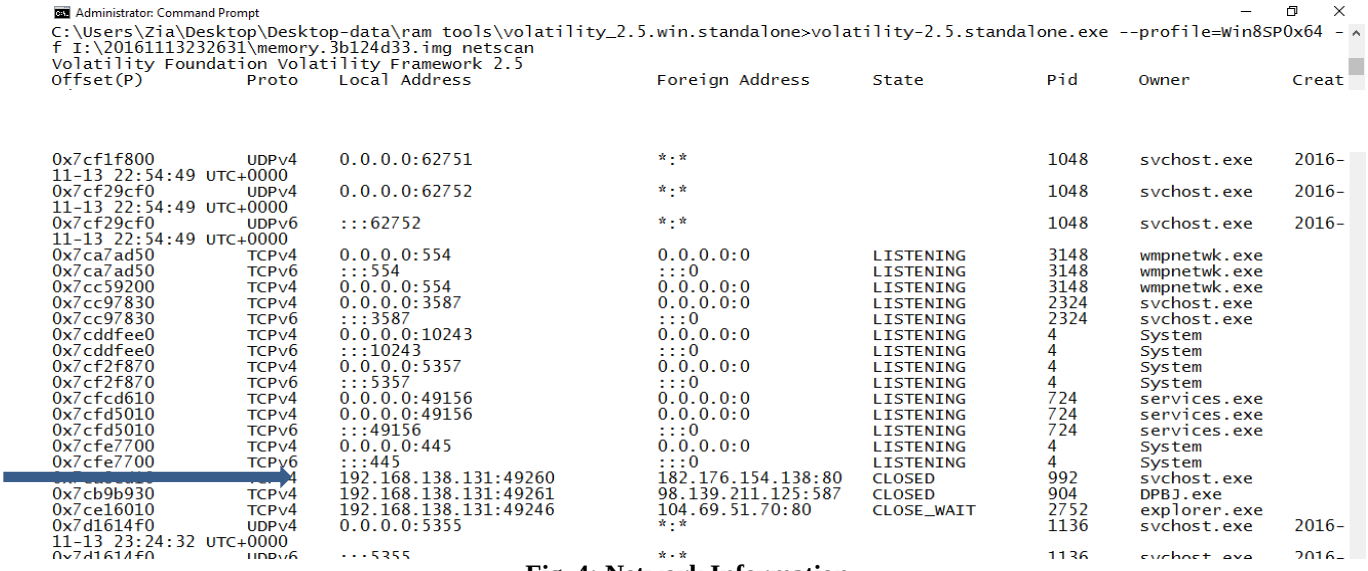
	E	I	L	M	Q
2	/ProcessItem/@created	/ProcessItem/name	/ProcessItem/path	/ProcessItem/pid	/ProcessItem/startTime
3	11/13/2016 23:37	lsass.exe	C:\Windows\system32	732	11/13/2016 22:54
4	11/13/2016 23:37	svchost.exe	C:\Windows\system32	848	11/13/2016 22:54
5	11/13/2016 23:37	wmpnetwk.exe	C:\Program Files\Windows Media Player	3148	11/13/2016 23:00
6	11/13/2016 23:37	svchost.exe	C:\Windows\system32	804	11/13/2016 22:54
7	11/13/2016 23:37	svchost.exe	C:\Windows\system32	1356	11/13/2016 22:54
8	11/13/2016 23:37	DPBJ.exe	C:\Windows\SysWOW64\28463	904	11/13/2016 23:10
9	11/13/2016 23:37	DllHost.exe	C:\Windows\system32	2792	11/13/2016 22:55

Fig. 2: FireEye Memoryze 3.0



Offset(V)	Name	PID	PPID	Thds	Hnds	Sess	Wow64	Start	Exit
0xfffffa801a593940	explorer.exe	2752	460	51	0	1	0	2016-11-13 22:56:42 UTC+0000	
0xfffffa801a5c7940	TabTip.exe	1664	1012	12	0	1	0	2016-11-13 22:56:49 UTC+0000	
0xfffffa801a6a4940	TabTip32.exe	1724	1664	1	0	1	1	2016-11-13 22:56:50 UTC+0000	
0xfffffa801a53b940	SearchIndexer.	2964	724	15	0	0	0	2016-11-13 22:58:36 UTC+0000	
0xfffffa8019025940	wmpnetwk.exe	3148	724	9	0	0	0	2016-11-13 23:00:52 UTC+0000	
0xfffffa801a798940	iexplore.exe	456	2752	8	0	1	0	2016-11-13 23:03:09 UTC+0000	
0xfffffa8019004380	iexplore.exe	3696	456	16	0	1	1	2016-11-13 23:04:23 UTC+0000	
0xfffffa801918d080	WUDFHost.exe	216	1012	7	0	0	0	2016-11-13 23:06:54 UTC+0000	
0xfffffa8019e07080	taskhost.exe	3640	724	9	0	0	0	2016-11-13 23:09:24 UTC+0000	
0xfffffa801b20b940	DPBJ.exe	904	252	3	0	1	1	2016-11-13 23:10:51 UTC+0000	
0xfffffa801a40c280	FlashUtil_Acti	3932	804	2	0	1	0	2016-11-13 23:10:58 UTC+0000	

Fig. 3: Volatility Framework 2.5



Offset(P)	Proto	Local Address	Foreign Address	State	Pid	Owner	Creat
0x7cf1f800	UDPv4	0.0.0.0:62751	*:*		1048	svchost.exe	2016-11-13 22:54:49 UTC+0000
0x7cf29cf0	UDPv4	0.0.0.0:62752	*:*		1048	svchost.exe	2016-11-13 22:54:49 UTC+0000
0x7cf29cf0	UDPv6	:::62752	*:*		1048	svchost.exe	2016-11-13 22:54:49 UTC+0000
0x7ca7ad50	TCPv4	0.0.0.0:554	0.0.0.0:0	LISTENING	3148	wmpnetwk.exe	
0x7ca7ad50	TCPv6	:::554	:::0	LISTENING	3148	wmpnetwk.exe	
0x7cc59200	TCPv4	0.0.0.0:554	0.0.0.0:0	LISTENING	3148	wmpnetwk.exe	
0x7cc97830	TCPv4	0.0.0.0:3587	0.0.0.0:0	LISTENING	2324	svchost.exe	
0x7cc97830	TCPv6	:::3587	:::0	LISTENING	2324	svchost.exe	
0x7cddfee0	TCPv4	0.0.0.0:10243	0.0.0.0:0	LISTENING	4	System	
0x7cddfee0	TCPv6	:::10243	:::0	LISTENING	4	System	
0x7cf2f870	TCPv4	0.0.0.0:5357	0.0.0.0:0	LISTENING	4	System	
0x7cf2f870	TCPv6	:::5357	:::0	LISTENING	4	System	
0x7cfcd610	TCPv4	0.0.0.0:49156	0.0.0.0:0	LISTENING	724	services.exe	
0x7cfd5010	TCPv4	0.0.0.0:49156	0.0.0.0:0	LISTENING	724	services.exe	
0x7cfd5010	TCPv6	:::49156	:::0	LISTENING	724	services.exe	
0x7cfe7700	TCPv4	0.0.0.0:445	0.0.0.0:0	LISTENING	4	System	
0x7cfe7700	TCPv6	:::445	:::0	LISTENING	4	System	
[illegible]	TCPv4	192.168.138.131:49260	182.176.154.138:80	CLOSED	992	svchost.exe	
0x7cb9b930	TCPv4	192.168.138.131:49261	98.139.211.125:587	CLOSED	904	DPBJ.exe	
0x7ce16010	TCPv4	192.168.138.131:49246	104.69.51.70:80	CLOSE_WAIT	2752	explorer.exe	
0x7d1614f0	UDPv4	0.0.0.0:5355	*:*		1136	svchost.exe	2016-11-13 23:24:32 UTC+0000
0x7d1614f0	UDPv6	:::5355	*:*		1136	svchost.exe	2016-

Fig. 4: Network Information

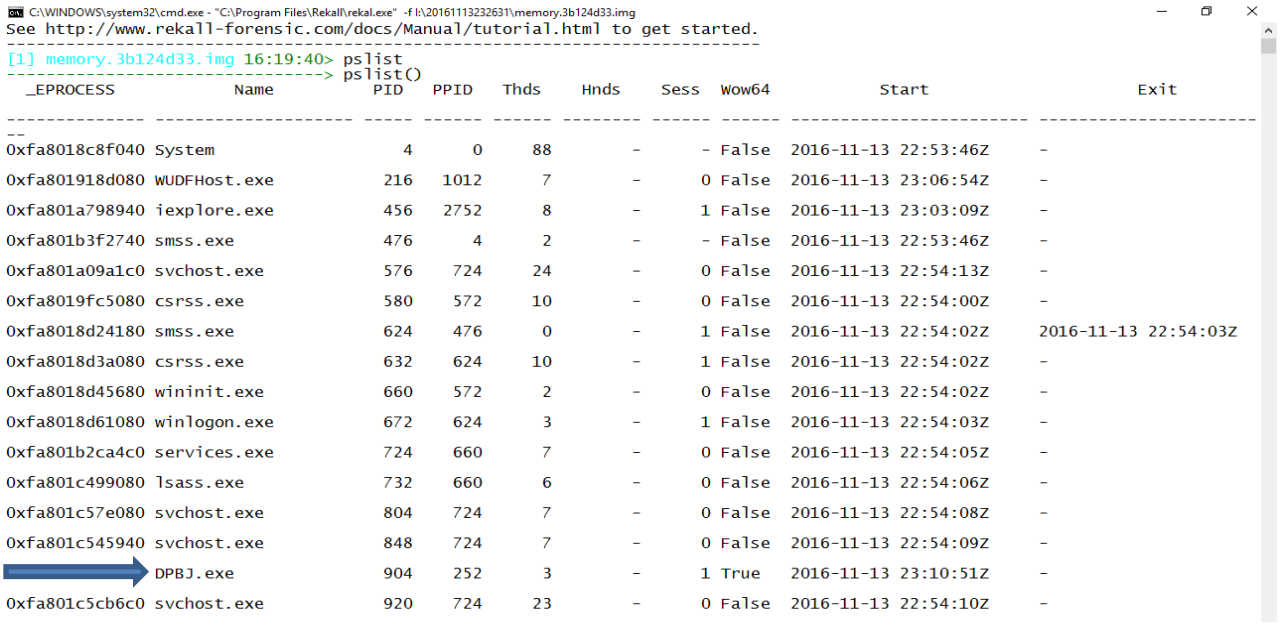


Figure 5: Rekall Framework

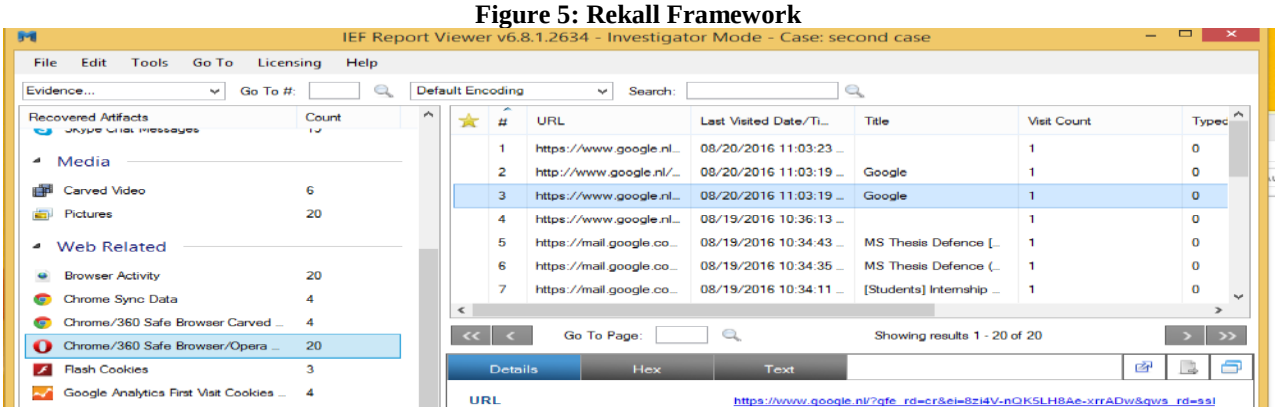


Fig. 6: Magnet IEF

Table 1 Support of tools for various platforms

Supported Operating System	Tools				
	FireEye Redline 1.14	FireEye Memoryze 3.0	Volatility Framework 2.5	REKALL 1.5.2 Furka	Magnet Internet Evidence Finder
Win10			✔ -Some features	✔ -Some features	✔
Win8.1	✔	✔	✔	✔	✔
Win8	✔	✔	✔	✔	✔
Win7	✔	✔	✔	✔	✔
WinXP	✔	✔ 32 bit	✔	✔	✔
Win Vista	✔	✔ 32 bit	✔	✔	✔
Win Server 2012			✔	✔	
Win Server 2008	✔	✔	✔	✔	
Winserver 2003	✔	✔	✔	✔	
Winserver	✔	✔ 32bit	✔	✔	
Linux			✔	✔	
Andriod			✔	✔	
MAC OS X		✔	✔	✔	
iOS					✔

Table 2 Tools attributes

Tool attributes	Tool Name				
	FireEye Redline 1.14	FireEye Memoryze 3.0	Volatility Framework 2.5	Rekall 1.5.2 Furka Framework	Magnet Internet Evidence Finder
Cost	Free	Free	Open Source	Open Source	Paid
Memory Acquisition	✔				
Interactive Web Console				✔	
GUI based Tool	✔		✔		✔
Command Line Supported		✔	✔	✔	
List all running process	✔	✔	✔	✔	
Lists Drivers	✔	✔	✔	✔	
Registry Data	✔	✔	✔	✔	
Event logs	✔		✔	✔	
Internet Artifacts	✔		✔	✔	✔
Services	✔		✔	✔	
List DLL	✔	✔	✔	✔	
Malware IOC's (Indicators of Compromise)Analysis	✔				
Process Information	✔	✔	✔	✔	
Process Tree	✔		✔	✔	
Supported Memory Formats	mans, raw, dd, img	dd, raw, img	dd, raw, vmss, vmsn, EWF, hpak, img	aff4, img, dd, raw	raw, dd, img
File Information	✔	✔	✔	✔	✔
Network Information	✔	✔	✔	✔	✔
Malware Risk Assessment	✔				
Lists Virtual Address Space		✔	✔	✔	✔
Verify Digital Signatures of Drivers,EXE & Dll	✔	✔			
Prints Readable Strings	✔	✔			✔
Address Translation from virtual to Physical				✔	
Login Credentials Extraction			✔	✔	
Size of the tool	70MB	8.21MB	16.6MB	17.2MB	273MB

Table 3 Tests Results

Tool Attributes	Tool Names			
	FireEye Redline 1.14	FireEye Memoryze 3.0	Volatility Framework 2.5	Rekall 1.5.2 Furka Framework
MRI (Malware Risk Index) shown	✔			
Network Information shown			✔	
Process Information shown	✔	✔	✔	✔
PID (Process ID)	✔	✔	✔	✔
Time Stamp Information shown	✔	✔	✔	✔
Parent ID Info.	✔	✔	✔	✔
Parent Process Name				
No of Threads			✔	✔

CONCLUSION AND FUTURE WORK

Memory Forensics is a growing field and a lot of research is being done in this regard. However the choice of memory analysis and acquisition tool still depends upon the investigator according to his choice and requirements of the scenario. To analyze a memory, it is very important to acquire memory properly, so that it should yield correct results.

In our scenario of malware analysis via memory forensics tools, we have seen that volatility gives the best results as compared to the other tools such as FireEye Redline, REKALL, and FireEye

Memoryze were not showing the network activity of the said malware. However, if we acquire the image via FireEye Redline then it gives the best results. Its graphical interface is very easy to understand and all the important artifacts of analysis data are shown with separate sidebars with detailed information. Moreover it also shows the Malware Risk Index which is very beneficial for quick analysis and identification of malware, features like these gives FireEye Redline an edge over Volatility Framework and REKALL Furka. FireEye Redline free premier version does not supports Windows 10, however FireEye HX (paid) can acquire the triage of the infected Windows 10 system with its HX agent and can analyze the results in the same way in FireEye Redline. Apart from Volatility we are unable to recover network based information from .img memory image from other tools.

In future, different methodologies can be devised to use memory analysis for reverse engineering purposes. With the simplest and precisely developed methodology, an investigator can collect filtered information in very less time. Some attributes that we have seen in table 2 are not supporting windows 10 as it is the most recent operating system. So, there is a need to work on open source projects to make more plugins for windows 10 also.

Some malwares employ anti debugging techniques so that they are invisible in memory or are difficult to analyze even from memory analysis. So there is a need to study such malwares and enhance the capabilities of current tools, to thwart malware based attacks.

REFERENCES

[1] S. Logen, H. Höfken, and M. Schuba, "Simplifying RAM forensics: A GUI and extensions for the volatility framework," *Proc. - 2012 7th Int. Conf. Availability, Reliab. Secur. ARES 2012*, pp. 620–624, 2012.

[2] K. Amari, "techniques-tools-recovering-analyzing-data-volatile-memory-33049," 2009.

[3] S. Vomel and F. C. Freiling, "A survey of main memory acquisition and analysis techniques for the windows operating system," *Digit. Investig.*, vol. 8, no. 1, pp. 3–22, 2011.

[4] L. Cai, J. Sha, and W. Qian, "Study on Forensic Analysis of Physical Memory," *Int. Symp. Comput. Commun. Control Autom.*, no. 3ca, pp. 221–224, 2013.

[5] L. Xu, L. Wang, S. Zhang, and H. Li, "A Method to Analyze Memory Images of 64-bit Windows 8," *Int. J. Digit. Content Technol. its Appl.*, vol. 7, no. 7, pp. 304–312, 2013.

[6] A. Schuster, "Searching for processes and threads in Microsoft Windows memory dumps," *Digit. Investig.*, vol. 3, no. SUPPL., pp. 10–16, 2006.

[7] A. Moser, C. Kruegel, and E. Kirda, "Exploring multiple execution paths for malware analysis," *Proc. - IEEE Symp. Secur. Priv.*, pp. 231–245, 2007.

[8] J. Butler and J. Murdock, "Physical Memory Forensics for Files and Cache," *Craigchamberlain.Dreamhosters.Com*, 2011.

[9] R. J. Mcdown, C. Varol, L. Carvajal, and L. Chen, "In-Depth Analysis of Computer Memory Acquisition Software for Forensic Purposes," *J. Forensic Sci.*, vol. 61, no. January, pp. 110–116, 2016.

[10] S. M. Hejazi, C. Talhi, and M. Debbabi, "Extraction of forensically sensitive information from windows physical memory," *Digit. Investig.*, vol. 6, no. SUPPL., 2009.

[11] W. Ahmed and B. Aslam, "A comparison of windows physical memory acquisition tools," 2015.

[12] Mark Baggett, "SANS ISC InfoSec Forums," *Powershell Malware - No Hard drive, Just hard times*, 2016. .

[13] J. Okolica and G. L. Peterson, "Windows operating systems agnostic memory analysis," *Digit. Investig.*, vol. 7, no. SUPPL., pp. S48–S56, 2010.

[14] Z. UrRehman, A. Ahmad, and S. Saleem, "Screenshots reference link of survay of memory analysis paper," 2016.

[15] Sheka, Ytisf, and Visitors, "A repository of LIVE malwares," 2014. .

[16] VirusTotal, "Virustotal," 2016. .

[17] A. Torres, "starring Windows 10," 2016.

[18] J. T. Sylve, V. Marziale, and G. G. Richard, "Pool tag quick scanning for windows memory analysis," *Digit. Investig.*, vol. 16, pp. S25–S32, 2016.

[19] E. G. Singh and M. Kaur, "Forensic Analysis of Data from Random Access Memory," vol. 3, no. 03, pp. 99–103, 2016.