Analysis of Privacy of Private Browsing Mode through Memory Forensics

Ahmad Ghafarian Department of CSIS Mike Cottrell College of Business University of North Georgia Dahlonega, GA 30005, USA

ABSTRACT

Most popular web browsers support private browsing mode. It is claimed that private browsing mode protects privacy by leaving no trace of surfing activities behind. Yet it poses a great challenge to the computer forensics investigators who try to reconstruct the past browsing history, in case of any computer incidence. The aim of this research is to use volatile memory forensics methodologies and tools to examine the artifacts left in main memory after a private browsing session. To achieve this goal, it first presents a memory forensics framework that will help the investigators to effectively capture and analyze memory associated with private browsing with respect to incidence response. It then uses the framework to experimentally capture and analyze the memory, for its evidential potential related to private browsing using Firefox, Google Chrome, IE and Safari. We also report the degree of privacy offered by the browsers under study.

General Terms

Computer forensics, user privacy, private browsing mode

Keywords

Browser; residual data; RAM forensics; tools; volatile memory; forensics artifacts; framework.

1. INTRODUCTION

When people surf the web, browsers save information about the surfing activities. In an attempt to maintain privacy of web browsing, recently all major web browsers have added private browsing mode (PBM) feature to their user interface. According to Aggarwal, et al. [1] there is two ways that a web browser saves information about surfing activities namely, local machine and web server. The local machine saves processing data in both static media such as hard drive and random access memory (RAM) which is also referred to as volatile memory. The major difference between the data sources in relation to a computer forensic investigation is that volatile memory is a less tangible source of evidence and is harder for an investigator.

Conventionally, computer forensic investigators focus on static media for data retrieval and acquisition. For example, Oh [11] and Ohana [12] show that private mode browsing in all major web browsers does leave some kind of recoverable data but it is difficult to establish a link between the user and a web browsing session. The same researchers also used RAM forensics methodology to investigate traces of artifacts left in main memory with regard to private browsing for several web browsers. They discovered that the private browsing mode in their tested browsers did not deliver privacy as they claimed they would. Other research results in the use of RAM forensics with respect to the privacy of PBM are also promising. For example, Mahendrakar, et al. [9] have Syed Amin Hosseini Seno Department of Computer Engineering Faculty of Engineering Ferdowsi University of Mashhad Mashhad, Iran

developed a memory parser tool and used it to parse the physical memory after a private mode browsing session. Their results show that memory forensics retrieves artifacts of private mode browsing which has some information about the suspect. Hejazi, et al. [7] used searching and other methods to retrieve forensically valuable data from physical memory. The authors demonstrated that their memory forensics methodology retrieve sensitive private mode browsing data from memory.

Memory forensics involves two steps, memory capture and analysis of the captured memory. RAM capture is the process of making an image of the physical memory and saving it as a file on a storage media. Memory analysis involves parsing the data structure tree of the captured memory file, looking for processes that were running when the memory was taken as well as other browsing data such as passwords, downloaded files, SSL Certificates, URLs, etc. To facilitate memory forensics, several open-source and proprietary RAM forensics tools have been developed. Some of the popular examples include Volatility [17], Mandiant Redline [10] and Belksoft evidence center [3]. Although technically all memory analysis tools parse the Virtual Address Descriptor tree but there are many issues that computer forensics investigators need to know before selecting a tool (see section 3). In addition, some tools do not analyze data about terminated processes, closed programs and files. A forensics specialist needs to explore other options or tools such as using WinHex [18].

The focus of this research is the examination of the residual traces left in main memory when PBM is used. First, it proposes a memory forensics framework that helps the investigators to effectively capture and analyze memory associated with private browsing mode with respect to incidence response. Then, it uses the framework to experimentally analyze the live captured memory, for its evidential potential related to private browsing mode using Firefox, Google Chrome, IE and Safari. The live memory image is taken in two different scenarios, i.e. with the browsers being left open after a session and the browsers being closed. The retrieved artifacts can be used as evidence admissible in the court of law.

The remainder of this paper is organized as follows: Section 2 gives literature review, section 3 provides memory forensics framework, section 4 covers research methodology, results are discussed in section 5, section 6 discusses conclusion and future research are explained in section 7.

2. LITERATURE REVIEW

Most of the previous research on private browsing mode leakage concentrates on static media with some reference to live memory forensics. For example, Oh, et al. [11] have used web browser's log file to collect information from static sources that is relevant to a private web browsing session. They concluded that it is possible to determine the objective, methods, and criminal activities of a suspect through analysis of the log file. Aggarwal, et al. [1] presents a comprehensive study of problems and issues with the privacy of PBM. But they acknowledge that they ignored privacy leakage through physical memory forensics.

A report of PBM weaknesses for popular regular and portable browsers can be found in [12]. In addition to the conventional forensics methodology they also performed limited RAM forensics. The researchers reported that they were able to retrieve some private browsing mode related activities but they acknowledge that no link between the suspect and the evidence was established and more memory forensics is needed.

Mahendrakar, et al. [9] examined various popular web browsers in private mode to determine traces of browsing activities that remains in physical memory. They created a website which contained individual pages that required the browser to interact with various types of data including SSL certificates, form passwords, form text entries, HTML files, JPEG files, and cookies. Since they used their own memory parser tool, which is not publicly available, and their experiment was performed in a controlled research setting environment, their result cannot be replicated.

Said, et al. [14] examined the content of the volatile memory after a private browsing session and found artifacts left in memory about user activities. They did not disclose the tools and their methodology in their paper. Many aspects of private mode browsing activities including memory forensics have also been reported by Satvat, et al. [15]. In their experiment, after navigating a few websites in the private mode and closing the session, they inspected the content in RAM and discovered traces of private navigation. These researchers also did not disclose the details of RAM forensics tools and methodologies in their paper and thus their findings cannot be proved by replication.

In a study of physical memory forensics, Hejazi, et al. [7] proposed a new technique for extracting sensitive information from physical memory. Their technique is based on analyzing the Call Stack and the security sensitive Application Program Interfaces (API). They have implemented this technique as part of memory analysis plug-in, which takes a memory image file and analyze the file. Although their result is important; but it does not suggest any practical application guidelines for computer forensics investigators.

A theoretical discussion of RAM forensics tools, techniques and guidelines can be found in [4], [16] and [2]. The authors provide a comprehensive discussion of the way physical memory works in Windows and Linux operating systems as well as the types of data that can be extracted from physical memory. In the light of these past researches on RAM forensics, we present our memory forensics in the next section.

3. RAM FORENSICS FRAMEWORK

Memory forensics is the acquisition and analysis of volatile memory [13]. Since the 2008 DFRWS challenge [6], many tools and techniques have been developed for the acquisition and analysis of physical memory. Selection of the appropriate tools and the process of RAM forensics are more challenging than conventional forensics for several reasons: The volatility nature of memory makes it difficult to collect data from live memory. Since the memory does not use a set structure, it makes it difficult to analyze the captured memory. In an effort to help forensics investigators in this process, we propose a RAM forensics framework. Our framework is shown in Figure 1 below.

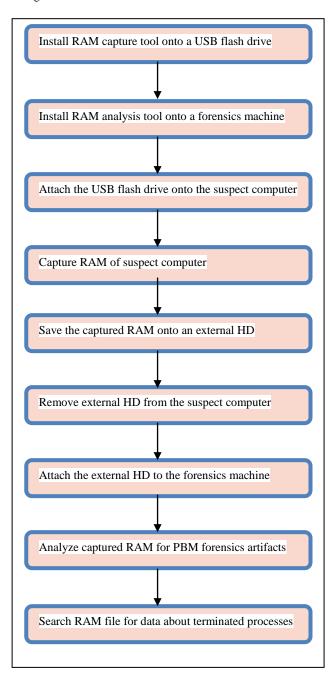


Figure 1: RAM Forensics framework

The framework shown in Figure 1 consists of three-stages:

- 1. Criteria for selecting memory acquisition tools
- 2. Criteria for selecting memory analysis tools
- 3. Steps for carrying out memory forensics

Table 1 summarizes the details of these stages.

RAM capture tool selection criteria.	RAM analysis tool selection criteria.	RAM forensics steps
The tool should:	The tool should:	
Support forensics workstation and	Support both forensics workstation and	Install selected RAM capture tool onto a
suspect machine OSs	suspect machine OSs	wiped flash drive, e.g. flash 1
Not require installation on the suspect	Be able to analyze different types of file	Install selected RAM analysis tool onto the
machine	formats e.gmem, .mans, etc.	forensics workstation
Run on kernel mode not the user mode	Have good review by other users of the	Attach flash 1 onto a suspect machine
	tool	
Be able to save the captured RAM on	Have GUI as opposed to command line	Capture RAM of the suspect machine and
a removable media		save it onto a wiped USB flash drive, e.g.
		flash 2
Capture the memory image in a	Have well written and easily accessible	Turn off the suspect machine and remove
reasonably small file	user manual	flash 2 from the suspect PC
Save the captured image in a file type	Be relatively easy to use by investigators	Import captured ram file from flash 2 onto
that is readable by the selected RAM		the forensics machine and start analyzing
analysis tool		the captured RAM
Be open-source/proprietary	Be open-source/proprietary	Search RAM file with another tool for
		residual data

Table 1. Memory forensics framework

3.1 Interpretation of Table 1 Entries

As Table 1 suggests, the size of the captured RAM and the amount of time it takes to acquire memory image are important criteria for the tool selection. This will help the efficiency of the memory forensics process. Additionally, the selected tool should not require installation on the suspect machine. This is because software installation on the suspect machine may change the content of memory which is not forensically acceptable. Also, the selected tool should run on the Kernel mode because the Kernel mode is secure and there would be no chance of suspect machine contamination. With regard to saving the captured RAM, researchers suggest the best practice is to save the memory image to an external device in order to minimize the impact the capture process has on the system being investigated. To address questions such as: should the tool be launched from an external drive or it should be installed on the hard drive. Again, the researchers suggest that it should not be installed on the target machine. This would also minimize the effect that the installation process may have on the machine being analyzed. Finally, RAM analysis should be able to analyze different file formats. This will give the RAM investigator flexibility of using different tools for RAM imaging and RAM analysis if needed. We should note that memory forensics tools can retrieve information about running processes and programs, open files, registry handles, network information, event logs, cookies, etc. [2]. However, once a process is terminated or a file is closed, the data structure that defines the process will no longer be a member of the data structure that the operating system maintains to keep track of what is currently running. Because RAM forensics tools parse the data structure tree, they cannot retrieve data about terminated processes and closed files. In addition, since RAM data may be available in hibernation files, in swap files, and in RAM that has not been reused, an investigator should use other tools and other methodologies to retrieve forensics artifacts. In this research we utilized a Hexadecimal editor such as WinHex [18].

4. RESEARCH METHODOLOGY

The tools used during forensics memory capture and analysis are listed in the next subsection.

4.1 Technology Used

- Five 64-bit laptops all running Windows 7, SP 1. Four laptops were used as suspect machines and the fifth one was used as forensics workstation.
- SATA to USB adaptor
- Tableau USB Write Blocker -IDE/SATA
- VMware workstation 10
- A forensically wiped USB flash drive
- WinHex
- Firefox 31.0, Chrome 43, IE 8 and Safari 5.1.7
- A WD Passport external hard drive
- Mandiant Redline.

We chose Redline an outstanding RAM forensic tool for the reconstruction of private web browsing activities for the following reasons:

- Graphical User Interface
- Selection option which allows you to choose only browsing related processes and disable all the other processes and files. This action shortens RAM analysis
- Allows you to import the memory analysis results to a MS Work file for offline processing
- Easy to user and has a comprehensive user manual

The process of RAM forensics is listed below:

- Use Redline to create a Redline Collector and save it onto an external media such as a USB flash drive. The collector is used to forensically capture the memory of the suspect machines.
 - 2. Save the Collector onto a portable storage device.
 - 3. Run the Collector from the portable storage device on the suspect computer to generate an audit i.e., collect data and metadata and save it to a file.

- 4. Save the audit from the target host back onto the portable storage device.
- 5. Import the audit into Redline to create an analysis session.
- 6. Review the data in the analysis session to begin the investigation.
- 7. Use additional tools and techniques to retrieve possible existing data in memory about terminated processes and closed files.

4.2 Experiment Details

We applied the framework described in the previous section and evaluated several existing memory forensics tools. We decided to choose Redline [10] which meets most of the criteria listed in the framework. The Redline RAM capture is called Collector which can be customized based on the type of the investigation. For example, in our case we were only interested on the processes that the operating system created for browsers. The memory analysis feature of Redline is called Memoryze which has a GUI interface and is embedded in Redline. In addition, we used WinHex to view residual information which was not retrievable by Redline collector.

Using the framework shown on Figure 1, a formal forensics environment was established, and all the experiments were carried out in forensically sound manner such that it is acceptable in court of law. We used five 64-bit laptops all running Windows 7, SP 1. Four laptops were used as suspect machines and the fifth one served as forensics workstation. We installed memory analysis tool of the Redline, i.e. Memoryze software on the forensics workstation. To simplify analysis, we disabled physical address extension mode on Redline. We ran Redline, created the RAM capture software called Collector and saved it on a wiped flash drive. Then we followed the below steps:

- We created a baseline virtual machine, i.e. VMware 10 workstations (VM) on all four suspect machines. The virtual machines were also running Windows 7, SP 1. The reason for using VM was to have an identical environment for all browsers used in this experiment.
- 2. To make data extracting less cumbersome, we uninstalled all currently installed web browsers from the suspect machines, cleared all cookies, cache, history, bookmarks, etc.
- 3. On each suspect machine' VM, we installed one specific Internet browser. The web browsers installed were Firefox, Microsoft Internet Explorer, Google Chrome Incognito (the term used by Google for the private browsing mode), and Safari. Then, we configured the browsers as the default browser with extensions and plug-ins disabled. This is because previous research shows that browser extensions and plug-ins interfere with private browsing [1]. Firefox, IE and Chrome Incognito were configured in private mode and since Safari does not support private mode configuration, we selected private mode manually.
- 4. For this experiment we define a browsing session as: images search, document search, video search on hacking, email login, attempted logon to a secure site such as a bank and attempted online purchase.

- 5. On each suspect's machine, we performed a browsing session as described in step 4 above. Next we:
 - Closed the browser.
 - Attached the flash drive that had RAM capture software, i.e. Redline Collector to the suspect machine.
 - Captured RAM and saved the file onto a sterile external hard drive to avoid contamination
 - Removed the external hard drive from the suspect machine for RAM analysis.
- 6. Step 5 was repeated for all the other suspect machines.
- 7. For comparison purposes, we repeated steps 5 and 6 above but this time we left the web browsers open after a browsing session ended.
- 8. For the memory analysis part, we attached the external hard drive that had Redline Memoryze installed on it onto the forensics workstation. We configured Redline to retrieve only browsing related information and processes. This action reduced the amount of data analysis and consequently shortens analysis time. We imported the memory parsed data to a MS Word for offline analysis. We should note that Redline only provide information about running processes and programs that were running before memory was captured. In order to evaluate the data about terminated processes, we also used WinHex. This process was very time consuming and requires knowledge of memory addressing.
- 9. Step 8 was repeated for the other three suspect machines' captured RAM files.

Over all we had four RAM captured files for the cases when browsers were closed after each browsing session and four memory captured files for the cases when the browsers were left open. The total captured memory files were eight. Considering each RAM capture on average taking one hour, eight hours was spent to capture the memory of the suspect machines. The process of memory capture and analysis were performed according to the forensics investigations rules and regulations. The results are discussed in the next section.

5. RESULTS

Retrievable computer forensics artifacts after a private browsing session through memory forensics are summarized in Table 2.

For Mozilla Firefox analysis of the memory dumped file showed considerable browsers related entries in memory indicating web browser activity. We were able to detect email communication details (see Figure 1), browsing and URL history, search history and downloaded files (documents, images, and videos) even after the browser was closed. However, when the browsers were closed, some information such as email password and Firefox process could not be retrieved. For Internet Explorer analysis of the RAM showed that browser closure had little effect and we were able to identify HTML data containing various types of information including the Certificate for accessing a secure website, URL, file downloaded and more. Before we captured RAM we deleted all cookies. After the browser was closed memory forensics showed deleted cookies. Also, all the event files with time stamp were retrieved from the memory (see Figure 2).

Analysis of physical memory when Google Chrome was used revealed forensically valuable artifacts such as Certificate, HTML text file, URL history, Cookies, files downloaded, etc. Like IE, Google Chrome explicitly saved considerable browsing information. For example, Figure 3 shows registry details that was captured during RAM analysis

For Safari, the amount of web activity data after private browsing is somewhere between Firefox and IE. We also compared the browser activities before the closure of the Safari. We found Safari also zeroed parts of the memory upon closure of the browser. Table 2 shows details of retrievable forensics artifacts with RAM forensics when various browsers were used in both cases of closing the browsers after a browsing session and leaving them open.

Data Item	Firefox 31	IE 8	Chrome 43	Safari 5.1.7	Firefox 31	IE 8	Chrome 43	Safari 5.1.7
	Closed	Closed	Closed	Closed	Open	Open	Open	Open
browser process	-	-	-	-	\checkmark	\checkmark	\checkmark	
URL History	V	\checkmark	V	V	V	V	V	
Cookies		V	\checkmark	\checkmark				\checkmark
File downloads	V		\checkmark	\checkmark	\checkmark	\checkmark	\checkmark	
Timelines	V	\checkmark	V	V	V	V	V	
Browser history	V		\checkmark	\checkmark	\checkmark	\checkmark	\checkmark	
Email password	-	\checkmark	-	\checkmark	\checkmark	\checkmark	\checkmark	
Email ID	V		\checkmark	\checkmark	\checkmark	\checkmark	\checkmark	
Videos	V		\checkmark	\checkmark	\checkmark	\checkmark	\checkmark	
Images	\checkmark		\checkmark	\checkmark	\checkmark			
Search history	\checkmark	\checkmark	\checkmark	\checkmark	\checkmark	\checkmark	\checkmark	

Table 2 Retrievable	nrivate browsing	mode artifacts v	with different browsers
1 abic 2. Kell levable	private browsing	mout ar matter v	with uniterent browsers

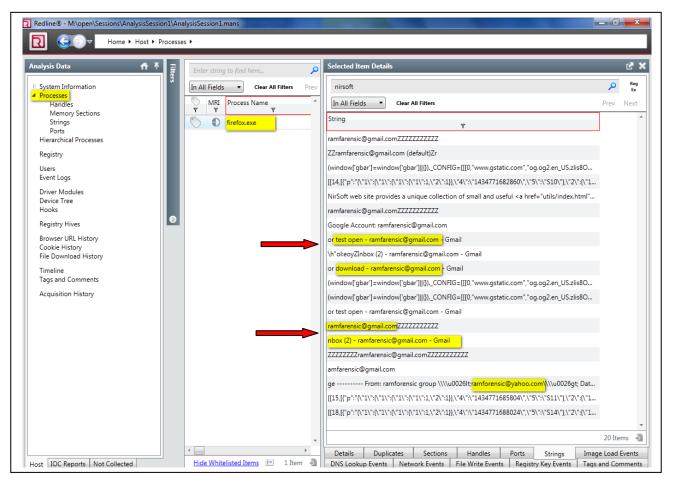


Figure 1. Memory analysis of Firefox showing email communication details

Analysis Data 🤺 🖡	-							
ulaiyaia Data 11 T	Enter string to fin	d here P Reg Ex In All Fields * Clear A	Il Filters Prev Next					
System Information Processes	Туре Т	Message Y	Source T	Generated ४	Written Y	Rese V	Username Y	Ma
Handles Memory Sections	Information	100:Get shader support	Microsoft-Windows-WindowsSystemAssessm	2015-06-07 14:22:30Z	2015-06-07 14:22:30Z	0	WIN-2A0GQFV3OSG\YA ALI	WI
Strings	Information	Recieved user logoff notification on session 1.	Microsoft-Windows-User Profiles Service	2015-06-13 08:10:23Z	2015-06-13 08:10:23Z	0	WIN-2A0GQFV3OSG\YA ALI	WI
Ports Hierarchical Processes	Information	100	Microsoft-Windows-WindowsSystemAssessm	2015-06-07 14:22:30Z	2015-06-07 14:22:30Z	0	WIN-2A0GQFV3OSG\YA ALI	Wit
Registry Windows Services Persistence	Information	Finished processing user logoff notification on sessio.	Microsoft-Windows-User Profiles Service	2015-06-13 08:10:24Z	2015-06-13 08:10:24Z	0	WIN-2A0GQFV3OSG\YA ALI	WI
	Information	110:Get WDDM support	Microsoft-Windows-WindowsSystemAssessm	2015-06-07 14:22:30Z	2015-06-07 14:22:30Z	0	WIN-2A0GQFV3OSG\YA ALI	WI
Users	Information	Recieved user logon notification on session 1.	Microsoft-Windows-User Profiles Service	2015-06-13 08:11:02Z	2015-06-13 08:11:02Z	0	WIN-2A0GQFV3OSG\YA ALI	WI
Event Logs	Information	110	Microsoft-Windows-WindowsSystemAssessm	2015-06-07 14:22:30Z	2015-06-07 14:22:30Z	0	WIN-2A0GQFV3OSG\YA ALI	WI
Tasks Ports	Information	50	Microsoft-Windows-WindowsSystemAssessm	2015-06-07 14:22:30Z	2015-06-07 14:22:30Z	0	WIN-2A0GQFV3OSG\YA ALI	WI
Driver Modules	Information	30	Microsoft-Windows-WindowsSystemAssessm	2015-06-07 14:22:30Z	2015-06-07 14:22:30Z	0	WIN-2A0GQFV3OSG\YA ALI	WI
Device Tree Hooks	Information	Starting session 0 - 2015-06-05T08:38:11Z.	Microsoft-Windows-RestartManager	2015-06-05 08:38:11Z	2015-06-05 08:38:11Z	0	WIN-2A0GQFV3OSG\YA ALI	WI
ARP Entries	Information	Finished processing user logon notification on sessio.	Microsoft-Windows-User Profiles Service	2015-06-13 08:11:03Z	2015-06- <u>13 08:11:03Z</u>	0	WIN-2A0GQFV3OSG\YA ALI	WI
Route Entries	Information	10	Microsoft-Windows-WindowsSystemAssessm	2015-06-07 14:22:30Z	2015-06-07 14:22:30Z	0	WIN-2A0GQFV3OSG\YA ALI	WI
Registry Hives	Information	Recieved user logoff notification on session 1.	Microsoft-Windows-User Profiles Service	2015-06-13 12:55:377	2015-06-13 12:55:372	0	WIN-2A0GQFV3OSG\YA ALI	WI

Figure 2. Content of event files retrieved during RAM analysis of IE

nalysis Data 🕺 👬 🖡	Filters 🔻	chrome	P Reg	In All Fields	Clear All Filters	Prev Next 198 ma	tches found		
System Information Processes		Path	Ÿ		Туре Ф	Text Value	Username T	Modified Ϋ	R 7
Hierarchical Processes		HKEY_USERS\S-1-5	-21-1732157850-41	15415769-1845848233-	REG_KEY		Raheleh\LaleH	2015-06-20 07:38:18Z	
Registry Windows Services	▷ □ SOFTWARE	HKEY_USERS\S-1-5	-21-1732157850-41	15415769-1845848233	REG_SZ	43.0.2357.124	Raheleh\LaleH	2015-06-20 07:38:18Z	2
	SYSTEM HKEY USERS	HKEY USERS\S-1-5	-21-1732157850-41	15415769-1845848233-	REG DWORD	1	Raheleh\LaleH	2015-06-20 07:38:18Z	4
Users Event Logs		-		15415769-1845848233-		0	Raheleh\LaleH	2015-06-20 07:38:18Z	4
Tasks	C_Users_Defa	-		15415769-1845848233-	-	,	Raheleh\LaleH	2015-06-19 12:36:08Z	
Driver Modules	C_Users_Defa	-				10070100050055500			
Device Tree Hooks	▷ □ S-1-5-18 ▷ □ S-1-5-19	ten.		15415769-1845848233-		13079190968856500	Raheleh\LaleH	2015-06-19 12:36:08Z	
Registry Hives	▷ □ S-1-5-20	HKEY_LOCAL_MAC	HINE\SOFTWARE\W	ow6432Node\Microsoft	REG_KEY		BUILTIN\Administrators	2015-06-19 12:33:18Z	
Sec. 1	▶ 🗌 S-1-5-21-173;	HKEY_LOCAL_MAC	HINE\SOFTWARE\W	ow6432Node\Microsoft	REG_SZ	Google <mark>Chrome</mark>	BUILTIN\Administrators	2015-06-19 12:33:18Z	2
Browser URL History Cookie History	▷ 🗌 S-1-5-21-173;	HKEY_LOCAL_MAC	HINE\SOFTWARE\W	ow6432Node\Microsoft	REG_SZ	"C:\Program Files (x86)\G	BUILTIN\Administrators	2015-06-19 12:33:18Z	2
Form History		HKEY_LOCAL_MAC	HINE\SOFTWARE\W	ow6432Node\Microsoft	REG_SZ	C:\Program Files (x86)\G	BUILTIN\Administrators	2015-06-19 12:33:18Z	9
File Download History		HKEY_LOCAL_MAC	HINE\SOFTWARE\W	ow6432Node\Microsoft	REG_SZ	C:\Program Files (x86)\G	BUILTIN\Administrators	2015-06-19 12:33:18Z	1
Timeline			HINE\SOFTWARE\W	ow6432Node\Microsoft	REG DWORD	1	BUILTIN\Administrators	2015-06-19 12:33:18Z	4
Tags and Comments				ow6432Node\Microsoft		1	BUILTIN\Administrators	2015-06-19 12:33:18Z	
Acquisition History					Charles and Charles and Charles	1			
		0		ow6432Node\Microsoft	0.011584900 -2 00569	Google Inc.	BUILTIN\Administrators	2015-06-19 12:33:18Z	
		HKEY_LOCAL_MAC	HINE\SOFTWARE\W	ow6432Node\Microsoft	REG_SZ	43.0.2357.124	BUILTIN\Administrators	2015-06-19 12:33:18Z	2
		HKEY_LOCAL_MAC	HINE\SOFTWARE\W	ow6432Node\Microsoft	REG_SZ	43.0.2357.124	BUILTIN\Administrators	2015-06-19 12:33:18Z	2
		HKEY_LOCAL_MAC	HINE\SOFTWARE\W	ow6432Node\Microsoft	REG_SZ	20150430	BUILTIN\Administrators	2015-06-19 12:33:18Z	1
		HKEY_LOCAL_MAC	HINE\SOFTWARE\W	ow6432Node\Microsoft	REG_DWORD	2357	BUILTIN\Administrators	2015-06-19 12:33:18Z	4
		HKEY_LOCAL_MAC	HINE\SOFTWARE\W	ow6432Node\Microsoft	REG_DWORD	124	BUILTIN\Administrators	2015-06-19 12:33:18Z	4
				15415769-1845848233-			Raheleh\LaleH	2015-06-04 18:34:23Z	

Figure 3. Details of Registry with time stamp with RAM analysis using Google Chrome

5.1 Analysis of the results

Interpretation of the data captured from memory indicate that private browsing mode does leave browsing evidence even after the browsers were closed in all four web browsers under this experiment. The type and the amount of data varied slightly among the browsers. For example, we created two email accounts namely ramforensics@gmail.com and ramforensics@yahoo.com and used them across all browsers to send/receive emails. For all browsers we were able to see the email ID and details of email communication. For Chrome, IE and Safari we could also retrieve email passwords but not for Firefox. Figure 2 shows retrieved email Id as indicated above and the password as 123456 when we used Google Chrome browser. This is because Firefox overwrites parts of the memory with zero after the browser process is terminated. This indicates that Firefox supports private browsing better than the other three browsers we worked with. Another important forensics artifact is downloaded files during a private browsing mode session. With RAM analysis we were able to retrieve the details of downloaded files such

as file name; timeline, size and type. Examination of the RAM analysis show that the searched items can be found after "q=" in memory dumped files for all four browsers. Also, all the sites that the suspect has visited are retrievable. Figure 5 shows the searched items and the site visited by the suspect during a private browsing session. They are shown after =" symbol in memory. The result of this experiemtn show that in the case of browser being left open, almost everything is retrievable through RAM forensics. When the browsers were terminated after a browsing session ends, Redline's Memorize did not report the existence of any private browsing processes for none of the browsers under consideration. However, we could see passwords for IE and Safari. We believe the data left in memory for all browsers are valuable forensics artifacts for an investigator.

With regard to searching the Internet, for the browsers we used every search made such as image search, document search, video search together with accessed email accounts were all recovered.

Offset	0	1	2	з	4	5	6	7	8	9	A	в	C	D	E	F	~
21E3EA00	01	20	28	39	2F	20	27	32	20	47	31	20	39	45	44	03	(9/ '2 G1 9ED
21E3EA10	12	83	FS	ЗE	04	00	27	01	20	27	33	2A	41	27	2F	47	∎ø> ' '3*Å'∕G
21E3EA20	20	34	48	2F	20	03	32	83	F8	зD	04	00	67	01	2E	6D	4H∕ 2∎ø= g.m.
21E3EA30	65	6D	31	27	20	28	27	20	72	61	6D	20	61	6E	61	6C	em1' (' ram anal
21E3EA40	79	73	69	73	20	74	6F	6F	6C	20	20	28	46	27	45	20	ysis tool (F'E
21E3EA50	72	65	64	6C	69	6E	65	22	46	27	44	03	13	83	F8	3C	redline"F'D ∎ø<
21E3EA60	04	00	29	01	20	28	44	27	41	27	35	44	47	20	48	20) (D'A'5DG H
21E3EA70	41	27	03	20	83	F8	зB	04	00	43	01	20	20	36	2D	28	A'∎ø; C 6-(
21E3EA80	44	27	41	27	35	44	47	20	27	32	20	31	45	20	63	61	D'A'5DG '2 1E ca
21E3EA90	70	74	75	72	65	20	03	16	83	F8	3A	04	00	2F	01	20	pture 🛛 ø: 🖊
21E3EAA0	28	27	34	2F	20	35	2D	28	33	2A	46	20	45	31	48	31	('4/ 5-(3*F E1H1
21E3EAB0	03	18	83	F8	39	04	00	33	01	20	27	37	44	27	39	27	∎ø9 3 '7D'9'
21E3EAC0	2A	20	28	2F	33	2A	20	22	45	2F	47	20	03	10	83	F8	* (∕3* "E∕G ∎ø
21E3EAD0	38	04	00	23	01	20	27	32	20	73	73	6C	20	28	27	46	8 # '2 ssl ('F
21E3EAE0	03	81	02	83	F8	37	05	00	82	05	01	20	2F	27	34	2A	∎ø7 ∎ ⁄'4 ×
21E3EAF0	47	20	34	2F	47	28	47	4D	41	49	4C	20	ЗA	20	55	73	G 4/G(GMAIL : Us
21E3EB00	65	72		61		65		ЗA	72	61		66	61	72			ername :ramfaren
21E3EB10	73	69	63	40	67	6D	61	69	6C	2E	63	6F	6D	20	50	61	sic@gmail.com Pa
21E3EB20	73	73	20	3A	20	6D	2E	31	32	33	34	35	36	20	5A	6 F	m.123456 Zo
21E3EB30	6F	6D	69	74	2E	69	72	20	ЗA	20	55	73	65	6E	61	6D	omit.ir : Usenam
21E3EB40	65				61		66	61	72	65		73	69	63		30	e :ramfarensic P
21E3EB50	61	73	73	20	ЗA	20	73	2E	31	32	33	34	35	36	20	34	ass : s.123456 4
21E3EB60	2D		6F	67	69	6E	20	03	10	83	F8	36	04	00	23	01	-login ∎ø6 #
21E3EB70	20	20	28	27	32	20	34	48	2F	20	27	03	14	83	F8	35	('2 4H∕ ' ∎ø5
21E3EB80	04	00	28	01	20	28	27	34	2F	20	2D	69	6D	61	67	65	+ ('4/ -image
21E3EB90	20	33	2D		13	83	F8	34	04	00	29	01	ЗA	20	2D	76	3- ∎@4) : -⊽
21E3EBA0	69	64	6F	20	63	6C	69	70	20	20	03	ЗD	83	F8	33	04	ido clip =∎ø3
21E3EBB0	00	7D	01	20	72	65	61	64	6C	69	6E	65	20	72	61	6D	<pre>} readline.ram</pre>
21E3EBC0	66	6F	72	65	6E	73	69	63	20	76	6F	6C	61	74	69	6C	forensic, volatil
21E3EBD0	69	74	79	20	72	61	6D	61	6E	6C	79	73	69	73	28	44	ity,ramanlysis(D
21E3EBE0	3A		2A	20	45	36	46	48	46	29	20	03	10	83	F8	32	:'* E6FHF) ∎ø2
21E3EBF0	04		23	01	20	20	34	48	2F	20	34	20	2A	27	20	03	# 4H 4 * '
21E3EC00	OD		00	00	12	00	46	00	03	EE	03	DS	03	C7	03	B2	F î Ø Ç ²
21E3EC10	03	AO	03	8F	03	7E	03	6B	03	52	03	2A	03	19	02	A1	~kR* i
21E3EC20	01	12	00	E2	00	C4	00	94	00	76	00	46	00	00	00	00	âĂ∎vF
21E3EC30	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
																	~

Figure 4. Memory analysis of Firefox shows the email Id and passwords as private browsing indicator

Offset	0	1	2	з	4	5	6	7	8	9	A	в	С	D	E	F	~
238453C0	39	5D	5C	5C	64	7B	32	7D	7C	33	28	ЗF	ЗA	5B	30	2D	9]\\d{2} 3(?:[0-
238453D0	38	5D	5C	5C	64	7C	39	5B	31	2D	39	5D	29	29	5C	5C	8]\\d 9[1-9]))\\
238453E0	64	7B	34	7D	29	22	2C	22	5C	5C	64	7B	39	7D	22	2C	d{4})","\\d{9}",
238453F0	2C	2C	22	38	31	30	31	32	33	34	35	36	22	5D	2C	OF	,,"810123456"],
23845400	OD	00	00	00	OC.	00	31	00	03	23	02	FD	02	C5	02	7D	1 # ý Å }
23845410	02	48	02	1 A	01	FA	01	CB	01	A7	01	50	00	7E	00	31	H úËSP~1
23845420	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
23845430	00	49	9D	E3	3A	05	00	81	13	01	3C	6D	65	74	61	20	Iã: <meta< td=""></meta<>
23845440	69	74	65	6D	70	72	6F	70	ЗD	22	6E	61	6D	65	22	20	itemprop="name"
23845450	63	6F	6E	74	65	6E	74	3D	22	65	68	73	61	6E	20	74	content="ehsan t
23845460	61	6E	68	61	65	65	20	66	72	65	65	20	6D	75	73	69	anhaee free musi
23845470	63	20	63	68	61	76	6F	73	68	69	65	20	64	OF	81	4D	c chavoshie d M
23845480	9D	E3	39	05	00	83	1B	01	3C	6C	69	6E	6B	20	69	74	ã9 ∎ <link it<="" td=""/>
23845490	65	6D	70	72	6F	70	3D	22	75	72	6C	22	20	68	72	65	emprop="url" hre
238454A0	66	3D	22	68	74	74	70	3A	2F	2F	77	77	77	2E	61	70	f="http://www.ap
238454B0	61	72	61	74	2E	63	6F	6D	2F	76	2F	4D	47	43	6F	68	arat.com/v/MGCoh
238454C0	2F	65	68	73	61	6E	5F	74	61	6E	68	61	65	65	5F	66	<u>/ehsan_tanhaee_f</u>
238454D0	72	65	65	5F	6D	75	73	69	63	5F	63	68	61	76	6F	73	ree_music_chavos
238454E0	68	69	65	5F	64	25	44	38	25	41	46	25	44	38	25	41	hie_d%D8%AF%D8%A
238454F0	37	25	44	39	25	38	36	25	44	39	25	38	34	25	44	39	7%D9%86%D9%84%D9
23845500	25	38	38	25	44	38	25	41	46	5F	25	44	38	25	41	37	%88%D8%AF_%D8%A7
23845510	25	44	38	25	41	44	25	44	38	25	42	33	25	44	38	25	%D8%AD%D8%B3%D8%
23845520	41	37	25	44	39	25	38	36	5F	25	44	38	25	41	41	25	A7%D9%86_%D8%AA%
23845530	44	39	25	38	36	25	44	39	25	38	37	25	44	38	25	41	D9%86%D9%87%D8%A
23845540	37	25	44	42	25	38	43	25	44	42	25	38	43	22	ЗE	OF	7%DB%8C%DB%8C">
23845550	53	9D	E3	38	05	00	81	27	01	20	20	20	20	3C	64	69	Sã8 '≺di
23845560	76	20	69	74	65	6D	73	63	6F	70	65	20	69	74	65	6D	v itemscope item
23845570	70	72	6F	70	3D	22	76	69	64	65	6F	22	20	69	74	65	prop="video" ite
23845580	6D	74	79	70	65	3D	22	68	74	74	70	3A	2F	2F	73	63	mtype="http://sc
23845590	68	65	6D	61	2E	6F	72	67	2F	56	69	64	65	6F	4F	62	hema.org/VideoOb
238455A0	6A	65	63	74	22	3E	OF	20	9D	E3	37	04	00	43	01	3C	ject"> ã7 C <
238455B0	64	69	76	20	69	64	3D	22	76	69	64	65	6F	4F	6E	65	div id="videoOne
238455C0	5F	70	6C	61	79	65	72	22	20	3E	OF	2B	9D	E3	36	14	plaver" > + ã6
238455D0	00	59	01	20	20	20	20	20	20	20	20	20	20	20	20	20	Y
238455E0	20	20	20	зC	2F	61	ЗE	3C	2F	64	69	76	зE	3C	2F	64	
238455F0	69	76	ЗE	3C	2F	64	69	76	ЗE	OF	10	9D	E3	35	04	00	iv> ã5
	100000																

Figure 5. Memory analysis of Google Chrome reveals the search items and sites visited during private browsing

6. CONCLUSION

This research proposed a new framework for physical memory forensics. The framework is consists of three stages, criteria for memory capture selection tool, criteria for memory analysis selection tool and steps for carrying out physical memory forensics.

The proposed RAM forensics framework was used to experimentally examine privacy feature of Firefox, IE, Chrome Incognito and Safari browsers when they are used in private mode. It was found that through memory forensics it is possible to retrieve forensically valuable information about suspect's activity, such as sites visited, Internet searches, attempt of secure sites login credentials, traces of email communication even after the browsers were closed. These artifacts are sufficient to constitute a link between the data and the suspect. The experiment shows that the Vendor's claim of privacy can be nullified through RAM forensics. In another word, the privacy claim of browsers vendors is not really true. If they want to deliver privacy they need to modify their browsers. Among the browsers under this experiment, Firefox is slightly better in terms of privacy but there are no differences among other three browsers.

7. FUTURE WORK

This research can be extended in several ways. First, determine better tools and methodologies for analyzing the volatile memory for data about terminated processes and closed files and programs. Second, repeat the same experience with different tool such as Volatility. Third, apply the RAM forensics framework to examine the private mode features of various portable web browsers. Fourth, extract information over an extended period of time instead of one specified browsing session. Fifth, do experiment with other browsers such as Opera and Amazon Silk.

8. ACKNOWLEDGMENTS

The author would like to thank the office of the UNG president for establishing and implementing the presidential semester award. Thanks also goes to the selection committee on the 2015 presidential semester award for having trust on my work and selecting me as one of the award recipients which made this research possible.

9. REFERENCES

- Aggarwal, G., Bursztien, E., Jackson C., & Boneh, D. ((2010). An analysis of private browsing modes in modern browsers. Proceedings of the 19th Usenix Security Symposium.
- [2] Amari, K., (2009). *Techniques and Tools for Recovering and Analyzing Data from Volatile Memory*. SANS Institute InfoSec Reading Room.
- Belksoft, *Live RAM Capturer* (2014). Retrieved on July 2014 from http://forensic.belkasoft.com/en/ram/download.asp
- [4] Davis, N. (2009). Live memory forensics for Windows Operating Systems. Eastern Michigan University, IA 328. Retrieved, January 2015 from

- https://www.emich.edu/ia/pdf/research/Live%20Memory%20 Acquisition%20for%20Windows%20Operating%20Syst ems,%20Naja%20Davis.pdf
- [5] Disk Wipe (2009). Retrieved on January 2015 from http://www.diskwipe.org/
- [6] DREWS (2008). Forensics challenge overview. Retrieved April, 2015 from http://www.dfrws.org/2008/challenge/index.shtml
- [7] Hejazi, S.M., Talhi, C. & Debbabi, M. (2009). Extraction of Forensically Sensitive Information from Windows Physical Memory. *Digital Investigation*, 6, 121-131. Elsevier publishing Co.
- [8] Koepi, D. (2010). Firefox Forensics. Retrieved November 2014 from http://davidkoepi.wordpress.com/2010/11/27/firefoxfore nsics
- [9] Mahendrakar, A., Irving, J., and Patel, S., (2010). Forensic Analysis of Private Browsing Mode in Popular Browsers. Retrieved August 2014 from http://mocktest.net/paper.pdf
- [10] Mandiant Redline User Manual (2014). Retrieved February 2015 from

https://dl.mandiant.com/EE/library/Redline1.7_UserGuide.pdf

- [11] Oh, O., Lee, S., and Lee, S. (2011). Advanced evidence collection and analysis of web browser activity. *Journal* of digital investigation 8, 62-70
- [12] Ohana, D.J. and Shashidhar, N. (2013). Do private and portable web browsers leave incriminating Evidence?: a forensic analysis of residual artifacts from private and portable web browsing sessions. *EURASIP J, on Inf.* S. 201, 6, 1-13
- [13] Ruff, N. (2008). Windows Memory Forensics. *Journal in Computer Virology*, 14, 83-100.
- [14] Said, H., Mutawa, A.H., Awadhi, A.I., Guimaraes, M. (2011). Forensic analysis of private browsing artifacts. *International Conference on Innovations in Information Technology* (IIT).
- [15] Satvat, K., Forshaw, M., Hao, F. and Toreini E. (2014). On the Privacy of Private Browsing – A Forensic approach. *Journal of Information Security and Application*, 19, 88-100.
- [16] Simons, M. and Slay, J. (2009). Enhancement of Forensics Computing Investigations Through Memory Forensics Techniques. *International Conference on Availability, Reliability and Security*.
- [17] Volatility Foundation: available online at: http://www.volatilityfoundation.org/
- [18] WinHex: available online at: http://www.xways.net/winhex/