2009 International Conference on Availability, Reliability and Security

Enhancement of Forensic Computing Investigations through Memory Forensic Techniques

Matthew Simon
Defence and Systems Institute (DASI)
University of South Australia
Adelaide, Australia
e-mail: matthew.simon@unisa.edu.au

Jill Slay
Defence and Systems Institute (DASI)
University of South Australia
Adelaide, Australia
e-mail: jill.slay@unisa.edu.au

***Abstract*— The use of memory forensic techniques has the potential to enhance computer forensic investigations. The analysis of digital evidence is facing several key challenges; an increase in electronic devices, network connections and bandwidth, the use of anti-forensic technologies and the development of network centric applications and technologies has lead to less potential evidence stored on static media and increased amounts of data stored off-system. Memory forensic techniques have the potential to overcome these issues in forensic analysis. While much of the current research in memory forensics has been focused on low-level data, there is a need for research to extract high-level data from physical memory as a means of providing forensic investigators with greater insight into a target system. This paper outlines the need for further research into memory forensic techniques. In particular it stresses the need for methods and techniques for understanding context on a system and also as a means of augmenting other data sources to provide a more complete and efficient searching of investigations.**

***Index Terms*— Computer forensics, Digital evidence, Digital investigation, Electronic evidence, RAM forensics, Volatile memory forensics.**

I. INTRODUCTION

The number and variety of electronic devices used in business and by consumers is growing rapidly. Moore's Law dictates that capacity of electronic devices increases exponentially. Connectivity of such devices is also growing rapidly; internet-based, local area and personal networks are common for both private and business users. Existing forensic processes do not scale at the same rate as that of technology growth. This leads to inadequate or overly complex forensic methodologies.

Forensic computing can be described as the investigation into criminal or improper activities that may have left digital evidence [1]. The incorporation of physical memory analysis into forensic computing investigations has the potential overcome some of the issues associated with conventional forensic processes. The acquisition and analysis of physical memory has not played a large role in conventional forensic computing methodologies. Conventional forensic computing methodologies focus on static media such as hard disks, optical media and flash memory. There are known processes for securing, preserving and analysing such sources. Physical memory, by contrast, is difficult for investigators to secure, preserve and analyse. The data that is contained within the two types of sources varies significantly. Static media is primarily used for long term storage and contains data such as executables, images, documents and browser history. Physical memory, by contrast, does not store files but rather is a temporary working space for data that are being used by the system. The major difference between the data sources in relation to a computer forensic investigation is the latter is a less tangible source of evidence.

While static based media continues to be the dominant source of evidence, the use of physical memory as a source of evidence may provide information that cannot be otherwise recovered. The development of technologies that impede forensic analysis and evidence recovery (*anti-forensics*) has become an issue for forensic computing investigators [2]. Conventional forensic computing methodologies may not be adequate for the recovery of data that is resident in a location other than the target system. The increased use of network applications such as *Voice Over IP* (VoIP), *instant messaging* and *social networking* may lead to data that is not stored on local static media.

The physical memory of a computer system is the temporary storage area for the operating system and applications running on the system. By offering insight into the current and previous state of the system, the data stored in physical memory may be of use in forensic computing investigations in several ways. Using this information, a picture of the actions that have been carried out on the system may be created. Current and remnant data used by applications or the system itself may leave a proverbial *breadcrumb trail*. The physical memory may also be instrumental in augmenting analysis of data on static media sources. Research into analysis of physical memory has been a serious area of research for little more than three years. Already there are positive results showing the viability of analysis of physical memory. This has been mostly evident in the recovery of passwords and encryption keys to defeat encryption software.

One of the issues currently faced in the analysis of physical memory is the recovery and use of application-level data. Current research in physical memory analysis has focused on the retrieval of operating system level information. Little has been conducted on recovering information at the application level. The focus towards application level data is necessary as the number and type of network-based applications and technologies continue to grow. Incorporating the analysis of physical memory into forensic computing methodologies can enhance forensic computing investigations. One way in which analysis of physical memory can do so is in the recovery of information about applications that leave little or no evidence on the systems static media. Data recovered from physical memory may also be able to enhance forensic computing investigations by fusing such data with data from other sources such as static-media or network capture.

II. Background

Memory forensics is the area of computer forensics that relates specifically to volatile memory. More specifically, it is the acquisition and analysis of physical memory [3, 4, 5, 6, 7]. Memory forensics is more challenging than disk-based forensics for several reasons: it is volatile in nature and therefore difficult to collect. It is also difficult to analyse, as memory does not use a set structure. Acquisition and analysis of the data represent separate distinct research areas and are the focus of much research within the discipline [4]. Although the concept of memory forensics has existed for some time, the catalyst for the current explosion in interested started as a result of the 2005 Digital Forensic Research Workshop (DFRWS) [5, 6]. Before this time, physical memory acquisition was primarily conducted in order to search for passwords [3]. Many tools and techniques have since been developed and released for both the acquisition and analysis of physical memory allowing much more than just passwords to be extracted.

Traditional forensic computing techniques are non-volatile storage-centric and do not collect and analyse data from volatile storage [4, 6, 8]. When a system is seized in a live state, a common course of action may be to power off the system by the pull-the-plug method; consequently, all of the volatile memory is lost. This method can be described as a "*snatch and grab approach*" as the computer is powered down and the static media is copied and made available for forensic examination [9]. In this way, the investigator can argue that there has been no interference with the collected evidence. If following accepted and forensically sound methods correctly, evidence will therefore be admissible in a court of law, as acceptable processes have been followed.

Memory forensics is now attracting both academic and practitioner attention [4] and this area is steadily evolving [10]. It is being noted that the pull-the-plug option is not always the best course of action as all information about the systems state is lost [5]. The importance of this evidence, not to replace but to enhance traditional sources of digital evidence, is now being realised. Adelstein [9] states that *"the pervasive nature of the internet makes contextual information more important*". Analysis of the system's memory greatly enhances understanding of the systems state. Memory forensics has the potential to recover much information about a system including its current and previous state that post-mortem disk-based analysis may not capture [7].

A. *The need for memory forensics*

There are a number of factors that have driven the development of memory forensics. The increased technological ability of criminals who are able to use technologies to conceal data or activity is a major factor [4]. Specifically, the increased use of hard drive encryption has lead to a need to recover encryption keys. Encryption keys stored in physical memory may be retrieved and used to decrypt data on static-media sources. A number of researchers have published very promising work in this area [11, 12].

Another reason for the increased attention to memory forensics is the growth in disk usage. Conventional methodologies may no longer be sufficient in dealing with the increasing storage ability [4]. Increases in disk capacity have lead to a greater difficulty and expense in collecting and analysing hard drives. Large RAID arrays, Network Attached Storage (NAS) and Storage Area Networks (SAN) can also be an issue for acquisition and analysis of disks [9]. Memory forensic techniques can be of use in these situations by providing another source of data if all or some of the hard disk data cannot be acquired.

Garner Jr. [13] believes that a fusion of multiple sources of evidence including traditional sources such as hard drives and non-traditional sources such as memory is advantageous within the forensic analysis process. The development of memory forensics in no way aims to replace existing forensic practices but rather to add to and create deeper understanding of all data that is collected. Volatile data may contain indications of recent system activity that is not otherwise evident [4]. For instance, illegal images may be found on a suspect's hard drive. Analysis of the memory image that was obtained may show evidence of an off-line storage repository where other illegal images may be stored. In many cases, the loss of volatile data may be detrimental to the ability to determine the previous activity of a system [8]. The ability to determine the prior activity of a system is the key to combating the shortfalls in purely disk-based forensic practices that focus solely on static storage media.

B. *Challenges in Memory Analysis*

The analysis of memory consists of two main challenges, finding and subsequently analysing the data structures [6]. Analysis presents a unique problem in memory forensics [14] for several reasons. As is with the acquisition phase, the issue with analysis is not related to the forensic process. Since the 2005 DFRWS challenge, there has been an active research community releasing tools to find information in physical memory images; many of these are open source. An issue with analysis of memory is that there is a lack of high-level analysis tools within the field. This means that analysts tend to spend more time on low level analysis and investigation rather than high level evidence gathering [10].

Most of the current tools have a low level focus, working only with a small subset of operating system memory structures. Memory is structured differently between operating systems and often varies between different versions of the same operating system [4]. Tools that rely on such specific low level structures must constantly be updated or use some other method of finding the information that can be made more generic, e.g. symbol sets [15]. In addition to the physical memory image, the memory swap file should be incorporated for a complete analysis of memory [4, 5, 6, 16]. The page file is commonly be locked by the operating system while the executing but the file can easily be obtained during post-mortem disk analysis [6]. One of the main issues with page file analysis is that memory captured using a method that leads to self-inconsistency (such as the application dd), will also lead to inconsistency between the memory image and the page file. Merging of the swap file and the physical memory image may be difficult as there are no freely available tools that perform this function. FATKit purportedly has functionality to do this although this has yet to be seen in the public domain [6].

C. Existing Memory Analysis Tools and Techniques

The development of analysis tools and techniques that can extract information from physical memory images is now occurring rapidly. This research and development is extremely important. Lack of high quality analysis tools and techniques will render the effort and risk of memory acquisition unwarranted irrespective of how forensically sound the processes are. Much of the analysis research has been geared towards Microsoft Windows physical memory images with little being done for Linux and less for Apple Macintosh. The primary objective of physical memory analysis is to understand the current and previous state of the target machine at the time of acquisition. Other objectives are to collect viruses, malware and rootkits for analysis. This is especially useful where encryption is used to protect the binary executables [8].

Early research into the analysis of physical memory images was performed by Burdach [15, 17] and the winners of the 2005 DFRWS challenge Chris Betz with his tool *memparser* and George M. Garner Jr. and Robert-Jan Mora with their tool *kntlist* [18]. Burdach's work looked at both memory analysis for Linux [15] and Microsoft Windows [17]. Burdach created the *Windows Memory Forensic Toolkit* (WMFT) as a proof of concept which is freely available to the public.

The memparser tool, created by Chris Betz, performs enumeration of the process list and dumps other information such as process environment information, DLL's in use and can extract memory space of individual processes [19].

Many tools and techniques have been developed since 2005 to extract various artefacts from physical memory images. Schuster [20] created a novel method for finding process objects, thread objects and many other object types. Previous methods, such as those published by Burdach [17] rely upon finding one or two processes by brute force and subsequently finding the remaining processes by following data pointers within the process structures. Hidden (via DKOM) and terminated processes will not be discovered when enumerating process threads using this method. This demonstrates that the analysis tools cannot necessarily be trusted to output the correct results [5]. Schuster's method overcomes this limitation. It scans the target memory image scan searching for known patterns, or signatures, of process objects. There are two advantages to this method to find structures in memory. The first being that objects hidden using DKOM will be recovered. The second advantage is that terminated processes may be recovered; this may be possible even after a system reboot [20].

An issue with tools such as *lspi.pl* that perform executable extraction is that the resultant extracted executable file is not exactly the same as the original executable. The differences in the files are due to writable sections within the executable [3]. Hashing methods such as MD5 cannot be used to compare the two files, as even a single bit difference will result in a different hash value. Kornblum [21] has made available a tool called *ssdeep* that implements "*context triggered piecewise hashing*". This can be used to compare files where some of the bits differ – such as executable files recovered from memory and the original file that is stored on the hard drive. The output of the tool is a percentage of how similar the files are. A higher value indicates a greater chance that two files were derived from the same location.

Several other tools exist that provide similar functionality to those already discussed. Other tools have been created that provide additional functionality to existing tools. *Procloc.pl* (derived from process locator) is another tool that scans a Microsoft Windows physical memory image in order to find process structures [5]; it is available under a GNU licence. *pmodump.pl* is a tool created by Joe Stewart for the *Truman Project*. The pmodump.pl tool extracts the virtual memory of a process by using a Microsoft Windows memory space characteristic. This characteristic is leveraged to find all of the memory pages that make up the target process's memory [6].

The cold boot attack research from Princeton University [11], in addition to the unique method of memory acquisition, has provided some very unique methods of recovering encryption keys in physical memory images. There are two unique facets to the research; the ability to correct random bit errors in the recovered keys and the ability to recover AES and RSA keys without knowing the in memory structure of the software which has stored the keys. The ability to repair random bit errors has been implemented for DES, AES, RSA and TWEAK encryption keys. The ability to find AES and RSA keys without knowing the memory structure of the software is a powerful feature. This allows the keys to be recovered without knowledge of the structure of the software that uses the keys. This is contrary to the method by Hargreaves & Chivers [12] that recovers keys from TrueCrypt encryption software based on the in-memory structures. Furthermore, the research by Halderman et al. [11] can detect the keys even in the event of bit.

A number of authors have discussed the need for the page file to be incorporated into analysis of physical memory [4, 5, 6, 16]. Memory pages that have been swapped out to the page file are just one of several issues faced when converting virtual addresses into physical addresses. Each virtual address can either have a valid or invalid status. Valid addresses will translate to a physical address in the physical memory. Invalid addresses can translate to a range of different options, a page file reference being just one. Kronblum [22] looks in detail at recovering additional data from memory images by retrieving data based on the type of invalid virtual address – he colloquially calls this "*using every part of the buffalo*". In experiments run by the author, additional information was recovered by translating just three types of invalid virtual addresses. Invalid virtual addresses that referenced page file entries were not used in the experimentation as a page file was not available [22]. The FATKit software contains support for page file integration during analysis [10].

Many of the techniques implemented in tools mentioned thus far are proof of concept tools. There is now a trend towards formal memory analysis tools. Such tools provide range of functionality rather than just a single purpose. The major players currently are Volatility, FatKit and KnTTools.

KnTTools, which includes the KnTList analysis tool, focuses not only on analysis but has substantial functionality for acquisition. This is in contrast to FATKit and Volatility that contain only analysis functionality. The KnTTools website lists the purpose of KnTList to analyse "*main computer memory by reconstructing the principle operating system-defined metadata elements that structure the memory, including the virtual address space of the system and other processes*" [23]. The method used to do this is based on Schuster's method's of finding structures in memory images [20]. KnTTools is a closed-source commercial product that is available only to military, civilian law-enforcement, civilian governmental agencies and higher educational institutions [23].

Volatility is both a framework and a set of tools for physical memory analysis. It is available to the public under a GNU licence. The philosophy behind Volatility is to be open source and free. The creators "*encourage people to modify, extend, and make derivative works, as permitted by the GPL*" as they do not believe that availability of such tools should be restricted [24].

As of version 1.3, Volatility is capable of recovering: image date and time, running processes, open network sockets, open network connections, DLLs loaded for each process, open files for each process, open registry handles for each process, a process' addressable memory and OS kernel modules. Volatility is also capable of mapping physical offsets to virtual addresses (can map strings to processes), finding Virtual Address Descriptor information, scanning for processes, threads, sockets, connections, modules and can extract executables from memory samples. Volatility also transparently supports a variety of sample formats (i.e., crash dump, Hibernation, DD) and provides automated conversion between formats [24].

The 2008 DFRWS challenge indicates the need for future research into memory forensics [25]. The challenge winners used a combination of PyFlag and Volatility to extract information from the supplied challenge data [26] – a capture of network traffic, a copy of a users Linux home directory and a full memory image of the Linux system. PyFlag was used for "*data carving, string extraction, network stream reassembly, browser history data parsing, and provide organization and aggregation for the*" [27]. Volatility was used to "*parse Linux kernel data structures, report key system and user state data, and provide context to the memory dump*" [27]. A number of other tools such as the Linux crash dump analysis tool were extended to extract information.

Physical memory analysis tools have been steadily evolving since interest piqued due to the 2005 DFRWS challenge. Pre-existing non-forensic based tools were initially used but their shortcomings were a driver for development for more specific tools. Both academia and practitioners have been responsible for the development of an array of ad-hoc tools used to extract information from physical memory images. Literature indicates that current developments are towards more holistic tools and frameworks that allow for the subtle differences between operating system versions and the not-so-subtle differences between operating system types. Much of this research has focussed on operating system structures and data while little has focussed on application specific data. Future research should address this lack of research.

III. Future Potential Research

The continued research within the domain of forensic computing is critical to the development of the field. There has been need identified for more sophisticated tools in which to investigate the increasingly complex nature of computer crime [28]. The increase in data volumes and digital devices, the use of encryption, reduced cost of technology, increased access to the Internet and increases in general computing literacy skills have all contributed to the need for such investigation tools [29, 30].

It has been shown in the literature that physical memory is now being considered as a viable evidence source in which to gain information than cannot be collected by other means. The use of physical memory as a source of evidence has been shown to help overcome some of the issues and complexities inherent computers and digital devices. An example of where physical memory analysis can aid forensic computing investigation is with the recovery of passwords. Simple tools such as Grep and Strings have shown to be of some use in the recovery of data from physical memory by recovering ASCII strings within the binary data [3, 5]. The issues, however, with using such tools are numerous. The primary disadvantages are that such tools may often recover too much data, limited types of data and offer little or no context about the data. Since 2005 there has been much research into tools and techniques that address the requirements of physical memory analysis. Many researchers have discussed methods or created tools to extract operating system information from physical memory images [3, 10, 15, 17, 20, 23, 24]. The advantage of tools that perform such functions is that they provide context and insight into the state of the system. Tools are currently available to extract a myriad of information such as processes, network objects and much other system data. This information is valuable to forensic computing investigators as it provides insight into the state of the operating system.

While current research is a leap forward from simple search tools, there are several disadvantages. Many of the current tools have a low level focus, working only with a small subset of operating system memory structures. Secondarily, the information that is extracted is low-level and directly related to the operating system itself. There is a lack of high-level analysis tools that gather information about non-system process. Investigators therefore spend much effort on low-level tasks rather than high level evidence gathering [10]. There is a need for tools and techniques that extract data from applications that reside in memory. Such tools and techniques are necessary to fully understand the current and previous state of the system.

It has been stressed that analysis of physical memory should be used to enhance rather than replace traditional sources of digital evidence [5]. It has also been stated that fusion of information from memory and other sources is advantageous to the field of forensic computing [13]. There is clearly a need for further research in the field especially in the area of application-level data extraction and fusion of data from multiple sources.

The use of memory forensic techniques is not suitable for all types of applications and technologies, nor is it suitable for all types of computer crime. It has been stated that the Internet is pervasive in nature and that conventional forensic computing methodologies may not identify certain activity that has taken place on a system [4, 9]. It has also been stated that increased technical ability of computer users enables data and activity to be more easily hidden such that it cannot be detected by conventional forensic computing techniques [4]. Applications and technologies that result primarily in disk-based data may not be suitable to memory forensic techniques. However, there are many applications and technologies enabled by the Internet, increased bandwidth and increased technical ability of users. Network-based applications and technologies may leave little evidence of their execution and use. This creates a situation where investigators may not detect that such applications and technologies have been used and therefore miss crucial evidence. There is a need for memory forensic techniques that can recover the remnant data from activity that has previously been carried out on a system that is not otherwise obtainable using conventional computer forensic methods.

The literature in the area indicates that the need for this research is strong. The literature in the area contains little in the way of research into extraction and analysis of high-level application data from physical memory.

IV. Conclusion

This paper highlights the need for memory forensics and the potential for future research. The need has been demonstrated by emphasising the weaknesses in conventional forensic computing methodologies, tools and techniques. The future research potential has been identified by a review of the relevant literature in the area.

Memory forensic techniques have the potential to recover digital evidence where conventional static-media based techniques cannot. As the digital evolution progresses, the challenges to forensic investigators are likely to expand. These challenges - the number and capacity of electronic devices, the network connectivity and bandwidth, and the potential for the use of anti-forensic techniques - are. There are also other challenges in the collection and analysis of devices technical challenges, such as the moving of data storage to off-site systems. Forensic computing research needs to meet these challenges with tools, techniques and processes to understand the potential of digital evidence and also to provide means to detect and analyse systems where these are utilised.

Memory forensic techniques have the potential to overcome some of these issues by augment existing evidence collection and analysis techniques. By using physical memory as a source of evidence, insight into the previous and current state of the system can be gained. Network applications that leave little evidence on static media may leave remnant data within the volatile memory store, allowing investigators to use this to identify previous actions of the system. Other actions such as the use of off-system storage repositories may also be identified by analysing physical memory. Anti-forensic measures such as encryption software may also be defeated by recovery of encryption keys and passwords.

Although much progress has been made, there is a need for further research into memory forensic techniques. Analysis of application level data is required to help counter the issues that have arisen due to the continued development of the digital environment. Current tools and techniques in memory forensics need to be expanded to encompass more than low-level analysis. The research need is for tools and techniques that can extract high level data about applications and technologies that are problematic for conventional forensic computing methodologies.

The research need has been identified in two areas – greater use and understanding of physical memory as a standalone source of evidence and use of physical memory in combination with other conventional sources of evidence as a means of providing greater understanding than possible from a single evidence source. The use of physical memory as a standalone source may offer forensic investigators insight into a target system. Applications that leave little evidence on static-media may be suited to this type of analysis. Fusion of physical memory with other sources of data has the potential to recover information that neither source could recover on its own.

V. References

[1] G. Mohay, "Technical challenges and directions for digital forensics," in International Workshop on Systematic Approaches to Digital Forensic Engineering, Taipei, Taiwan, 2005, pp. 155-161.

[2] V. Broucek and P. Turner, "Winning the battles, losing the war? Rethinking methodology for forensic computing research " Journal in Computer Virology, vol. 2, pp. 3-12, 2006.

[3] H. A. Carvey, Windows Forensic Analysis: Syngress, 2007.

[4] E. Huebner, D. Bem, F. Henskens, and M. Wallis, "Persistent systems techniques in forensic acquisition of memory," Digital Investigation, vol. 4, pp. 129-137, 2007.

[5] T. Vidas, "The Acquisition and Analysis of Random Access Memory," Journal of Digital Practice, vol. 1, pp. 315-323, 2006.

[6] N. Ruff, "Windows Memory Forensics," Journal in Computer Virology, vol. 4, pp. 83-100, 2008.

[7] B. Schatz, "BodySnatcher: Towards reliable volatile memory acquisition by software," Digital Investigation, vol. 4, pp. 126-134, 2007.

[8] L. Sutherland, J. Evans, T. Tryfonas, and A. Blyth, "Acquiring volatile operating system data tools and techniques," ACM SIGOPS Operating Systems Review, vol. 42, pp. 65-73, 2008.

[9] F. Adelstein, "Live Forensics: Diagnosing Your System Without Killing it First," Communications of the ACM, vol. 49, pp. 63-66, 2006.

[10] N. L. Petroni Jr., A. Walters, T. Fraser, and W. A. Arbaugh, "FATKit: A framework for the extraction and analysis of digital forensic data from volatile system memory," Digital Investigation, vol. 3, pp. 197-210, 2006.

[11] J. A. Halderman, S. D. Schoen, N. Heninger, W. Clarkson, W. Paul, J. A. Calandrino, A. J. Feldman, J. Appelbaum, and E. W. Felten. (2008). Lest We Remember: Cold Boot Attacks on Encryption Keys. [Online]. Available: http://citp.princeton.edu/memory.

[12] C. Hargreaves and H. Chivers, "Recovery of Encryption Keys from Memory Using a Linear Scan," in ARES, Bercelona, Spain, 2008.

[13] G. M. Garner Jr. (2008). Forensic Acquisition Utilities. [Online]. Available: http://gmgsystemsinc.com/fau.

[14] D. Farmer and W. Venema, Forensic Discovery. Upper Saddle River, NJ: Addison-Wesley, 2005.

[15] M. Burdach. (2005). Digital forensics of the physical memory. [Online]. Available: http://forensic.seccure.net/pdf/mburdach_digital_forensics_of_physical_memory.pdf.

[16] B. D. Carrier and J. Grand, "A hardware-based memory acquisition procedure for digital investigations," Digital Investigation, vol. 1, pp. 50-60, 2004.

[17] M. Burdach. (2005). An Introduction to Windows memory forensic. [Online]. Available: http://forensic.seccure.net/pdf/introduction_to_windows_memory_forensic.pdf.

[18] DFRWS. (2005). DFRWS 2005 Forensics Challenge. [Online]. Available: http://www.dfrws.org/2005/challenge/index.shtml.

[19] DFRWS. (2005). Memparser Analysis Tool by Chris Betz. [Online]. Available: http://www.dfrws.org/2005/challenge/memparser.shtml.

[20] A. Schuster, "Searching for processes and threads in Microsoft Windows memory dumps," Digital Investigation, vol. 3 (S), pp. 10-16, 2006.

[21] J. Kornblum, "Identifying almost identical files using context triggered piecewise hashing," Digital Investigation, vol. 3(S), pp. 91-97, 2006.

[22] J. D. Kornblum, "Using every part of the buffalo in Windows memory analysis," Digital Investigation, vol. 4, pp. 24-29, 2007.

[23] GMG Systems Inc. (2007). KnTTools with KnTList. [Online]. Available: http://gmgsystemsinc.com/knttools/.

[24] Volatile Systems. (2008). The Volatility Framework: Volatile memory artifact extraction utility framework. [Online]. Available: https://www.volatilesystems.com/default/volatility#overview.

[25] DFRWS. (2008). DFRWS 2008 Forensics Challenge Overview. [Online]. Available: http://www.dfrws.org/2008/challenge/index.shtml.

[26] Volatile Systems. (2008). PyFlag/Volatility Team Wins DFRWS Challenge! . [Online]. Available: http://volatilesystems.blogspot.com/2008/08/pyflagvolatility-team-wins-dfrws.html.

[27] DFRWS. (2008). DFRWS 2008 Forensics Challenge Results. [Online]. Available: http://www.dfrws.org/2008/challenge/results.shtml.

[28] M. Reith, C. Carr, and G. Gunsch, "An Examination of Digital Forensic Models," International Journal of Digital Evidence, vol. 1, Autumn 2002.

[29] B. Etter, "The Forensic Challenges of E-Crime," Australasian Centre For Policing Research2001 2001.

[30] R. McKemmish, "What is Forensic Computing," Australian Institute of Criminology, 118, 1999.