Contents lists available at ScienceDirect

Forensic Science International: Digital Investigation

journal homepage: www.elsevier.com/locate/fsidi

DFRWS EU 2025 - Selected Papers from the 12th Annual Digital Forensics Research Conference Europe

A scenario-based quality assessment of memory acquisition tools and its investigative implications

Lisa Rzepka[a,*], Jenny Ottmann[b], Radina Stoykova[c], Felix Freiling[b], Harald Baier[a]

[a] *University of the Bundeswehr Munich, Germany*
[b] *Friedrich-Alexander-Universität (FAU) Erlangen-Nürnberg, Germany*
[c] *University of Groningen, the Netherlands*

ARTICLE INFO

Keywords:
RAM acquisition
Digital forensics
Volatile storage acquisition
Law enforcement
Criminal investigation
Tool testing
Reliability validation

ABSTRACT

During digital forensic investigations volatile data from random-access memory (RAM) can provide crucial information such as access credentials or encryption keys. This data is usually obtained using software that copies contents of RAM to a memory dump file concurrently to normal system operation. It is well-known that this results in many inconsistencies in the copied data. Based on established quality criteria from the literature and on four typical investigative scenarios, we present and evaluate a methodology to assess the quality of memory acquisition tools in these scenarios. The methodology basically relates three factors: (1) the quality criteria of the memory dump, (2) the applied memory forensics analysis technique, and (3) its success in the given investigative scenario. We apply our methodology to four memory acquisition tools (from both the open source and the commercial community). It turns out that all tools have weaknesses but that their inconsistencies appear to be not as bad as anticipated. Another finding is that unstructured memory analysis methods are more robust against low quality (i.e., inconsistent) memory dumps than structured analysis methods. We provide the measurement dataset together with the tool by which it was acquired and also examine our findings in the context of legal and international standards for digital forensics in law enforcement investigations.

1. Introduction

Memory dumps, i.e., copies of the contents of volatile random-access memory (RAM), are a rich source of forensically relevant information that usually cannot be obtained from persistent storage media such as hard disks or flash storage. For example, many types of malware are designed to operate only as memory resident programs and thus leave no traces if the system is shut down (Barnes, 2021). Other examples of data that can only be retrieved from RAM include artifacts of private browsing, keys to encrypted containers or indicators of active network connections.

In contrast to persistent storage acquisition, which is a routine operation robustly performed in almost any criminal investigation today, the acquisition of volatile memory is a much more delicate affair. For example, modern processors distinguish physical and virtual memory which defines multiple "views" on the contents of RAM. Moreover, investigators usually have the choice to acquire the (virtual) memory of a specific process or the (physical or virtual) memory of the operating system, a choice that is often confusing. Furthermore, since contents of RAM fade quickly once power is cut (Halderman et al., 2009), its contents are usually acquired when the target system is still running (Latzo et al., 2019). Since different parts of memory are acquired at different times, concurrent system activity leads to inconsistencies such as *page smear*, a situation where "the state of memory as described by the page tables [differs from] what is actually in those pages of memory" (Case and Richard III, 2017). These and other inconsistencies are commonly observed in memory dumps today (Pagani et al., 2019). In fact, with growing amounts of physical memory, such inconsistencies appear to be the rule rather than the exception in Linux (Ottmann et al., 2024) and in Windows (Rzepka et al., 2024) systems.

While inconsistencies in memory dumps appear unpleasant from a conceptual point of view, there has been some debate in the literature (Case and Richard III, 2017; Pagani et al., 2019; Ottmann et al., 2024) whether such inconsistencies are bad from an investigative point of view, i.e., whether they negatively interfere with the "prosaic business of hunting down cybercriminals and throwing them in jail" (Anderson et al., 2013). This question has many facets since an investigative lead not only depends on the quality of the memory dump, but also on the

* Corresponding author.
E-mail address: lisa.rzepka@unibw.de (L. Rzepka).

https://doi.org/10.1016/j.fsidi.2025.301868

Available online 24 March 2025

way in which it is analyzed. Here, two analysis techniques are commonly distinguished (Case and Richard III, 2017): *structured analysis* techniques interpret kernel data structures to enumerate running processes or analyze a process heap, whereas *unstructured analysis* techniques merely scan memory for patterns that "look like", say, a process table entry, a methodological antagonism also known from disk forensics where unstructured analysis is known as *file carving* (Richard III and Roussev, 2005). But since even state-of-the-art memory analysis tools like Volatility (Ligh et al., 2014) can only find relevant data if it is actually contained in the memory dump, it is clear that the analysis quality of memory forensic tools is bounded from below by the quality of memory acquisition tools. It is therefore vital to develop quality criteria for memory acquisition tools and put them into relation to the analyzability of their memory dumps, i.e., the ability to precisely and reliably answer relevant investigative questions using state-of-the-art analysis tools on these memory dumps, a condition of legal and societal relevance.

1.1. Related work

Early quality criteria for memory dumps, like *fidelity* by Schatz (2007) or *speed* of Inoue et al. (2011), informally expressed the requirement that the memory dump is "a precise copy [of] the original hosts's memory" (Schatz, 2007) or a "self-consistent […] snapshot of memory" (Inoue et al., 2011). Based on a formal system model, Vömel and Freiling (2012) defined the concept of *atomicity* to express that memory dumps should be causally consistent, informally meaning that they "should not show any [negative] signs of concurrent system activity" (Vömel and Freiling, 2012). A more practical criterion of *time consistency* was later proposed by Pagani et al. (2019) and formalized by Ottmann et al. (2024), demanding that an ideal snapshot be "instantaneous", meaning that it could have been taken from a frozen system.

Other work investigated consistency indicators, such as the virtual memory area (VMA) data structure of the Linux kernel (Pagani et al., 2019), the virtual area descriptor (VAD) data structure in Windows (Rzepka et al., 2024), or specially crafted linked lists that were created within the heap of a running program and that allowed to check for causal inconsistencies (Ottmann et al., 2024). While these works found that most memory snapshots were inconsistent in some way, they did not perform a comparative evaluation of memory acquisition tools and their evaluation of the effect of their findings on the success of memory analysis tools was limited.

Some prior comparative evaluation methods for memory acquisition have been suggested (Campbell, 2014; McDown et al., 2015; Vömel and Stüttgen, 2013; Gruhn and Freiling, 2016) and some do so in relation to formal quality criteria (Vömel and Stüttgen, 2013; Gruhn and Freiling, 2016). However, no prior work has related any measured quality criterion to the possibility of solving investigative questions in memory analysis. We also are not aware of any work that has discussed any of the proposed quality criteria in a legal context.

1.2. The need for scenario-based evaluations

While it is clear that memory dumps today are likely to contain inconsistencies, it is still not clear to what extent they influence the analysis success. During an analysis, the investigator needs to decide whether a found artifact is important for the case. To accomplish the decision, she assembles hypotheses which require testing before coming to a conclusion (Casey and Rose (2010); Cook et al. (1998)). Therefore, the use of hypotheses is essential in order to perform a goal-oriented analysis of a usually large amount of data. For this reason, we suggest to perform quality assessments of live memory acquisition tools based on *scenarios*, i.e., typical investigative circumstances where the analysis success can be measured. Osborne (2013, p. 15) states common forensic artifacts, e.g. executed processes or open files, which are needed by investigators to rebuild the system state at the moment of acquisition. In this work, we focus on four scenarios, detailed in section 4.3, that deal with the first four artifacts mentioned by Osborne (2013) as typical investigative questions, i.e. hypotheses, in memory forensics.

1.3. Contributions

To summarize, with this paper we make the following contributions:

1. We present a fully automated, scenario-based methodology to compare memory acquisition tools. The methodology can be expanded to include different scenarios or consistency indicators and can be used for closed source as well as open source tools.
2. For four tools and four typical scenarios, we provide measurement results that allow to compare the consistency indicators and artifact retrievability in each scenario for both structured and unstructured analysis approaches. We thereby show that all tools produce memory dumps with a considerable amount of inconsistencies, but that these inconsistencies mainly negatively affect structured analysis approaches.
3. We provide access to our resulting data set consisting of 1600 memory dumps produced by the four considered acquisition tools (i.e., 400 memory dumps per tool) for further analysis (Rzepka (2024)).
4. We also discuss the impact of our findings in a criminal investigation context.

1.4. Outline

This paper is structured as follows: Section 2 introduces prior evaluations and explains the consistency indicators which we later use for our evaluation. Next, Section 3 shows the current validation requirements for memory forensics. We present our assessment methodology in Section 4. Our evaluation results are given in Section 5. Finally, we discuss the results in Section 6 and conclude our paper in Section 7.

2. Background

Since our work uses the Windows operating system, we use two consistency indicators to quantify inconsistencies in the generated memory dumps, as proposed by Rzepka et al. (2024).

First, we look at causal inconsistencies (originally called violations of *atomicity* by Vömel and Freiling (2012)) in the heap of a test program implemented by Ottmann et al. (2024). The so-called pivot program generates and then continuously changes a linked list. The causal dependencies between the list elements are tracked using vector clocks (Mattern, 1989) saved within the elements. This allows identifying causal inconsistencies between the elements of the linked list captured in the memory dump.

Second, we count inconsistencies in a Windows memory management structure called *Virtual Address Descriptor* (VAD) tree. We use the VAD variable VadCount which holds the total number of VAD nodes in the tree and compare it to the number of manually determined nodes in the tree. If the two numbers are not equal, there is an inconsistency. This method is implemented using a Volatility 3[1] plugin.

3. Validation requirements for memory forensics

This section summarizes the international guidelines and best practices on memory forensics validation and requirements for reliability of digital evidence from RAM. It is demonstrated that international standards and guidelines define rather broad and imprecise conditions for memory acquisition, while concrete validation and testing schemes are largely missing.

[1] https://github.com/volatilityfoundation/volatility3

3.1. International standards

ISO17025 is the guiding standard for testing and the calibration of forensic laboratories (ISO/IEC, 2017). The UK Forensic science regulator is the only standardization body that has provided guidelines for its implementation to digital forensics labs. According to the guidelines, validation is defined as the "process of providing objective evidence that a method, process or device is fit for the specific purpose intended", where fit for purpose means that the method "is good enough to do the job it is intended to do, as defined by the specification developed from the end-user requirement". End-user requirements for methods and tools in memory forensics are derived primarily from forensic guidelines (Kent et al., 2006, 5.2.1) and can be summarized as follows:

Specialized training in memory forensics Most international standards and guidelines refer to the need of specialized training and knowledge for the identification and capture of live dynamic data in memory (European Network of Forensic Science Institutes (ENFSI), 2015; National Institute of Standards and Technology, 2022; International Criminal Police Organization (Interpol), 2019).

Identifying files of interests for law enforcement The computer system should be examined for potential sources of evidence that can merit a live acquisition. For example, if an investigation concerns identity theft, the contents of RAM might reveal social security and credit card numbers, programs used to obtain or encrypt data, password hashes, and methods that might have been used to obtain the information over a network (Kent et al., 2006).

Justifying live acquisition and choice of methods and tools Casey (2007) explains that during processing some data alterations are inevitable. The plethora of memory acquisition tools vary widely in the type of evidence they can capture, while the decision which tool to use in the concrete scenario is time sensitive and depends on the examiner expertise (Böhm et al., 2021). Therefore, guidelines recommend that the choice to perform live acquisition on a running system should be justified by the examiner (Kent et al., 2006). The tool should capture the data with minimal changes to the system, and any changes to the system should be identifiable and not affect the integrity of the data provided for evidence (Mocas, 2004) (International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC), 2012).

Chain of custody preservation Memory acquisition has inherent risks not only in terms of certain alteration on the system (forensic footprint of the tool) but also DLL changes and memory leaks by acquisition tools (European Network of Forensic Science Institutes, 2015), while a malicious party might have installed rootkits designed to return false information. Therefore, it is recommended that "the analyst should fully document what is seen on the screen before touching the system" (Kent et al., 2006). Documentation requirements are related to the tool and method selected and how they are applied to the concrete forensic task in the case. Tools should be reported based on name, version, configuration, and functions used. References to previous validation and verification testing and issuing known error reports, e.g. underlying system/software interpretation limitations, bugs in the version and the tool's ability to report errors in output are required. On a method level, the documentation should contain the logical sequence of steps intended to accomplish a defined task. This can refer to standard procedures or peer-reviewed techniques for RAM acquisition. Any deviation from standard procedure or method limitations must be recorded as well. Further documentation requirements are related to the application of the tool and method to the concrete task: date and time of acquisition, hypotheses and assumptions, what is seen on the screen and all forensic actions taken, examiner‘s interaction with the tool and assessment of tool results and confidence levels (Stoykova and Franke, 2023).

Minimize contact with the system To this end, forensic tools should be executed from a write-once CD or USB flash drive. This is important in order to minimize changes to the system and to avoid malicious processes set on the system. European Network of Forensic Science Institutes, 2015 (p.33) emphasizes in their guidelines that memory acquisition should be supported by a report on the expected forensic footprints of all tools and that "tools and processes that have the least forensic footprint for the current case requirements [should] be used in preference whenever possible to reduce loss of evidence".

Document authenticity and accuracy of the data It should be reported in the tool output or analyzed by the examiner if the data collected is likely to be accurate and authentic and whether rootkits and other malicious activities were present to corrupt it.

Avoid dependencies on the system Tools should consist of statically linked binary files to avoid usage of libraries installed on the target system.

Tool impact The analyst should know how each tool affects or alters the system before collecting the volatile data.

Verifying file integrity Message digest of each tool should be computed and stored safely to verify file integrity.

Tool testing and validation Typical approaches for assessing the correctness of tool results in digital forensics are: (i) dual tool verification or (ii) testing against known data sets with ground truth data (Hargreaves et al., 2024). However, dual tool verification in live memory acquisition is technically impossible since the system has changed. Therefore, testing the tools performance with a known dataset before applying it to real investigative scenario is essential to detect common errors (Lyle et al., 2022). Known errors should be taken into consideration during investigation and inform the decision to select a tool.

Apart from these general guidelines, none of the reviewed international standards and guidelines provide concrete validation scenarios or measurements for memory acquisition tools testing. National Institute of Standards and Technology (2017) pioneered the development of Computer Forensics Tool Testing (CFTT) which consists of a specific methodology to demonstrate the reliability of forensic results, to identify potential errors, and at the same time to support admissibility of evidence. CFTT testing is based on case studies which allows to test specific forensic function, rather than a tool type. However, for now only limited digital forensic tool functionalities are tested. Currently, there are no test results published on memory acquisition functionalities. Therefore, this paper provides further a framework and first validation results on memory acquisition tools.

3.2. A legal note on the reliability of digital evidence from RAM

The fair trial principle is recognized in all jurisdictions as a minimum guarantee for the quality of criminal procedures. Considering memory forensics in the context of this principle is important to inform forensic experts on how to document their findings for criminal proceedings. The specific nature of live acquisition of volatile data requires their consideration in the context of three core defense rights: (i) possibility to evaluate the legality and use of evidence; (ii) to receive expert report including findings in favor of the defense; and (iii) opportunity to challenge expert evidence reliability (Stoykova, 2023). As RAM can contain personal data such as encryption keys or social security and credit card numbers, the investigator must ensure that a proper authorization is obtained prior to the acquisition. Otherwise, the legality of the acquisition can be challenged in court. In contrast, live acquisition can be the only possibility, where the authorization does not permit the seizure of the system (Montasari, 2017). Further, the defense should have the opportunity to cross-examine and challenge the reliability of expert evidence on an informed basis. This means that the defense should be provided with full expert reports on the memory acquisition as well as any information that is relevant for the assessment of the reliability and integrity of it (Stoykova, 2023). The integrity of volatile data can be legally challenged based on inconsistencies in the acquisition such as incomplete or missing process lists or claims that the acquisition tool damaged/changed the memory data in a way that affects its integrity (Cook et al., 1998). Therefore, it is of key importance for the forensic examiner to provide documentation not only on the processes captured in RAM but also on their context such as justification for the need of live acquisition, inspection of the device up-time since the last re-boot, justification of the selected tool and method, and full work log (Watson and Jones, 2013). Information relevant to assess the integrity of the evidence is related to accounting for modifications to the running system and the assessment of the quality of the memory dump, reporting errors and uncertainties in output. The examiner does not have to account for all potential inconsistencies in the memory dump, but only for those that might impact the quality of the data adduced in evidence. Proper documentation of all running processes in RAM also allows to counter the so called “Trojan horse defense” where defendants argue that illegal content discovered on the device was unknown to them (Sammons, 2015). In addition, the forensic examiner should provide expert findings on exculpatory evidence. In relation to memory artifacts, the absence of illegal content or incriminatory processes must be documented, as this can support the defense stand.

4. Assessment methodology

In this section we explain our scenario-based assessment methodology which is a first attempt to measure the quality of memory dumps produced by different tools. Firstly, we present an automated procedure for our evaluation process (described in Section 4.1). Secondly, we show a suite of measurements which is applied on the acquired snapshots (see Section 4.2). In Section 4.3 we describe in detail four sample investigative scenarios, which we use for our evaluation in order to show the utility of our methodology.

4.1. RAM acquisition process

We now describe how we generated the RAM dumps. We used a QEMU virtual machine with Windows 10 in update version 22H2 installed. The Windows virtual machine operates six CPUs and 8 GB of RAM. The use of a page file is deactivated in the settings of Windows. To automate the scenario execution and the memory acquisition process, the data syntheses framework ForTrace++[2] (Wolf et al., 2024) is used to control the VM in the least invasive manner, from the outside. For this reason, the acquisition time of the tools is not measured exactly, as ForTrace++ needs to recognize the end of the acquisition process via the GUI output. In order to establish a communication between host and virtual machine, ForTrace++ makes use of the virtualization API libvirt, version 8.0.0.

For each scenario, we created a test setup in which a known ground truth was established (e.g., an image file was opened). The virtual machine is initialized in three steps in each scenario. After booting the Windows operating system, we wait for a fixed time, δ seconds, to give the system enough time to start up properly and to be in a stable state. Then, we initiate the pivot program provided by Ottmann et al. (2024) to track causal inconsistencies. Again, we give the operating system enough time to finish this step (i.e. δ seconds) before the corresponding scenario is executed (see Fig. 1). We chose 60 s as δ, which we determined by relying on previous experiments made with the Windows operating system (Rzepka et al. (2024)). In the scope of this work, we determined the different workload levels by experimenting with a

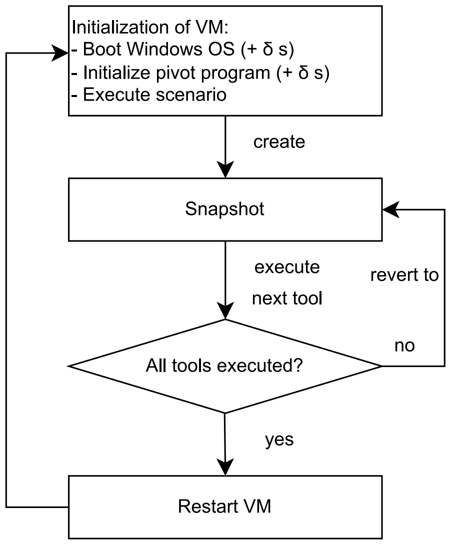

Fig. 1. Visualization of the RAM acquisition process. The process is implemented using ForTrace++.

different amount of applications which are opened after starting the virtual machine and (roughly) when the virtual machine reacts to the first opening request, i.e., how long we have to wait in order for the virtual machine to not skip the execution of the first requested application. This parameter is therefore a mere experimental result and can be easily adapted in future work to investigate its impact on the system, i.e., the amount of inconsistencies and whether the artifact of the scenario can be retrieved. Moreover, in order to avoid the problem of Windows updates, we checked for updates (and implemented them) before we conducted the experiment of a respective scenario. Updates were not installed after the experiment procedure for a scenario was started. The result of the initialization is a snapshot of the current state of the Windows VM, which is saved and then used as an identical basis for the actual RAM acquisition step.

For the first RAM acquisition tool, we load the snapshot within the Windows VM and then execute the respective tool to acquire the Windows main memory. After the first tool is finished, we set the virtual machine back to the snapshot state from the initialization step and restart it for the second RAM acquisition tool. This loop is done after each acquisition process (see Fig. 1). When all four memory dumps of one iteration are acquired, we restart the virtual machine. This process is repeated 100 times for each scenario leading to 400 memory dumps per scenario and 1600 RAM dumps throughout this work.

For each scenario we generated a ground truth by acquiring a small number of *instantaneous* memory dumps via `libvirt`, i.e., in a situation where the virtual machine is artificially paused. In such conditions we expect the memory dump to have no VAD or causal inconsistencies and to contain the corresponding artifact.

In this work we make use of four common RAM acquisition tools as listed in Table 5. We selected these tools due to their pervasiveness in the market. Furthermore, we target tools from both the closed source and the open source community.

4.2. Snapshot measurement

After all memory dumps are acquired, we apply a suite of measurements to each snapshot. More specifically, the following data points are gathered per memory dump file:

[2] https://gitlab.com/DW0lf/fortrace

Table 1
Overview of the results of scenario 1, including in how many memory dumps the process Discord.exe could be found, the number of VAD and causal inconsistencies, as well as in how many memory dumps those inconsistencies were found.

	Tool 1	Tool 2	Tool 3	Tool 4	Ideal Snapshot
Mean acquisition time in min.	2.25	3.03	9.18	3.19	–
Found process (structured)	96/100	96/100	69/100	94/100	10/10
Found process (unstructured)	100/100	100/100	100/100	100/100	10/10
No. analyzable memory dumps	90/100	71/100	72/100	97/100	10/10
No. VAD inconsistent memory dumps	89	71	72	97	0
Total VAD inconsistencies	61651	55705	137466	66345	0
Mean VAD inconsistencies	685	785	1909	684	0
No. analyzable memory dumps	96/100	97/100	74/100	91/100	10/10
No. causally inconsistent memory dumps	87	86	69	83	0
Total causal inconsistencies	1708	1672	3674	1897	0
Mean causal inconsistencies	18	17	50	20	0

1. The amount of causal inconsistencies in the memory dump (computed using a python script by Ottmann et al. (2024)).
2. The number of VAD inconsistencies of all processes in the system at the time of acquisition (using a Volatility 3 module provided by Rzepka et al. (2024)).
3. The fact whether the target artifacts of the specific scenario are retrievable using structured and unstructured analysis.

By doing so we can rank the four tools regarding their success rate in the given scenarios, and we can also better understand the impact of inconsistencies on the retrievability of the relevant artifacts in the given scenario. However, this work is a mere first attempt to measure the quality of a memory dump and we want to verify whether our proposed method is suitable. We also noted whether any of these data points were not measurable, e.g., because the heap of the pivot process (Ottmann et al., 2024) was not extractable. The entire analysis process is automated using multiple Python scripts.

4.3. Scenarios

During our evaluation we make use of four different use case scenarios, where a RAM acquisition and analysis is relevant. Each scenario evaluation is based on a different forensic artifact, which are listed in Table 6. The key idea of our evaluation is to make use of a suitable RAM analysis tool (e.g., a Volatility plugin (Ligh et al., 2014)) and to check whether it is able to successfully extract the forensic artifact relevant in the respective scenario. We furthermore aim at relating the quality of a RAM dump to both a structured and unstructured analysis approach, hence we apply different Volatility plugins, if possible, in the same scenario. We now give further details on these scenarios.

Scenario 1 is based on the use case scenario of process detection, i.e., software being executed on the system. This is a typical RAM analysis aspect to detect and inspect all executables present in the snapshot. As a structured analysis approach we make use of the Volatility plugin `pslist` to decide if our specific process (in our case *Discord.exe*) is detected or not. The `pslist` plugin makes use of Windows Kernel structures to find all processes, it walks along the ActiveProcessLinks list. Additionally, we use the plugin `psscan` as an unstructured analysis approach. The plugin uses pool header tag scanning (Schuster, 2006) to find any running processes. The technique of pool header tag scanning makes use of a certain data structure present in memory, the pool header. The pool header holds a field called *tag* which specifies the type of the object following the header (here: *proc*).

Next, Scenario 2 addresses the RAM analysis aspect of open network connections. This aspect helps assessing, for example, the possibility of file up-/downloads or communication with specific platforms. When starting our scenario, we initiate a secure shell (ssh) connection, which is open during RAM acquisition. Our structured detection of the ssh connection makes use of the Volatility plugin `netstat`, which extracts information from the tcpip.sys driver to find open network connections. For an unstructured analysis approach, the Volatility plugin `netscan` is used, which is based on pool tag scanning.

In Scenario 3 we turn to anti-forensic procedures like encryption, where RAM analysis enables access to plain text structures or the decryption key which, for example, will ease further searches for relevant information in encrypted storage. During scenario execution, we mount a VeraCrypt container, which contains encrypted data. As there is no Volatility plugin to print the key of a mounted VeraCrypt container, we used the tool `aeskeyfind`[3] which produces a list of key candidates from the memory dump based on block entropy. We consider this method to be unstructured. We then search through the list of candidate keys for the correct one, which we verify by mounting the VeraCrypt container on a Linux machine using the `cryptsetup` utility and printing the master key. Although there seems to be a structured approach to extract keys for TrueCrypt (Hargreaves and Chivers (2008)), we are not aware of any tool finding encryption keys for VeraCrypt in a structured way.

Finally, Scenario 4 looks at the extraction of file handles opened by a process. This information is required to, for example, establish the date and time of active user involvement with illegal material residing on the computer. In this scenario, we open a jpg image during scenario execution and make use of the Volatility plugin `filescan` to decide if the image is found during RAM analysis. The plugin does not retrieve the actual picture, but the file handle which includes the name and the path to the picture. The `filescan` plugin is based on pool header tag scanning and prints the filename as well as the physical offset of the file. We are not aware of a tool that performs the analysis in a structured way.

5. Evaluation

We now present the evaluation results for each of our four scenarios.

5.1. Scenario 1: Searching for an executed process

As explained above, we executed the Discord.exe as a sample process while acquiring RAM using the tested tools. We used the Volatility plugins `pslist` as structured and `psscan` as unstructured analysis approach to find this process in the memory dump. The measurement results are summarized in Table 1, which we now explain in detail. The table lists measurements for the individual tools in their respective columns. The rightmost column shows analysis results for the idealized instantaneous snapshot acquired after freezing the system using the hypervisor.The row *mean acquisition time* depicts the average time in minutes it takes to acquire one RAM instance by the respective tool. We see that while Tool 1 only requires roughly 2 minutes to dump memory, Tool 3 is rather slow and needs about 9 minutes on average.

The next two lines summarize the fraction of memory dumps in

[3] https://www.kali.org/tools/aeskeyfind/

Table 2
Overview of the results of scenario 2, including in how many memory dumps the network connection could be found, the number of VAD and causal inconsistencies, as well as in how many memory dumps those inconsistencies were found. Additionally, as netstat reports page errors, the overall number is listed for each tool. However, the relationship between those page errors and other analysis problems is currently unclear.

	Tool 1	Tool 2	Tool 3	Tool 4	Ideal Snapshot
Mean acquisition time in min.	3.32	4.74	11.52	4.53	–
Found connection (structured)	98/100	98/100	90/100	94/100	9/10
No. page errors reported by netstat	67/100	58/100	63/100	64/100	0
Found connection (unstructured)	100/100	98/100	92/100	96/100	10/10
No. analyzable memory dumps	91/100	59/100	69/100	91/100	10/10
No. VAD inconsistent memory dumps	91	59	62	87	0
Total VAD inconsistencies	67214	66948	188633	86614	0
Mean VAD inconsistencies	739	1135	2734	952	0
No. analyzable memory dumps	94/100	87/100	53/100	86/100	10/10
No. causally inconsistent memory dumps	87	79	49	78	0
Total causal inconsistencies	2619	2666	3026	3005	0
Mean causal inconsistencies	28	31	56	34	0

Table 3
Overview of the results of scenario 3, including in how many memory dumps the encryption key of the VeraCrypt container could be found, the number of VAD and causal inconsistencies, as well as in how many memory dumps those inconsistencies were found.

	Tool 1	Tool 2	Tool 3	Tool 4	Ideal Snapshot
Mean acquisition time in min.	3.04	4.26	12.43	10.17	–
Found encryption key (unstructured)	100/100	100/100	99/100	100/100	10/10
No. analyzable memory dumps	97/100	66/100	80/100	89/100	10/10
No. VAD inconsistent memory dumps	96	65	76	89	0
Total VAD inconsistencies	49508	59850	199495	45634	0
Mean VAD inconsistencies	510	907	2494	513	0
No. analyzable memory dumps	98/100	95/100	87/100	96/100	10/10
No. causally inconsistent memory dumps	91	88	83	90	0
Total causal inconsistencies	1841	2175	4345	1854	0
Mean causal inconsistencies	19	23	50	19	0

which the analysis tool succeeded to positively answer the investigative question, i.e., to find evidence of the running process `Discord.exe`. Using the unstructured approach, all tools were successful in all 100 memory dumps. Using the structured analysis approach, however, we can see that almost all the memory dumps generated by tool 1, 2 and 4 contain the process (e.g., in 96 of the 100 dumps acquired with Tool 1). In dumps generated by Tool 3, however, analysis is successful in only 69 of 100 dumps.

The next section of the table addresses results regarding VAD inconsistencies. For each of the four tools, the row *No. analyzable memory dumps* contains the number of RAM dumps where the VAD inconsistency approach actually works. For instance, we are able to apply the approach to measure the number of VAD inconsistencies in 90 of the 100 RAM dumps acquired by Tool 1. Overall for all tools, VAD analysis failed in $10 + 29 + 28 + 3 = 70$ out of 400 memory dumps. For those dumps that could be analyzed, the following rows show the total and mean number of VAD inconsistencies. The row *No. VAD inconsistent memory dumps* exhibits the number of VAD analyzable RAM snapshots, which contain at least one inconsistency. We see that there is only 1 memory dump generated by Tool 1 which contains no VAD inconsistencies. Noteworthy is also the relatively large number of inconsistencies of Tool 3, where all evaluable 72 RAM dumps contain more than 137.000 inconsistencies in total with a mean twice as high as the next best tool (Tool 4).

The final section of Table 1 presents our results with respect to the number of causal inconsistencies. Similar to the section above, we first provide for each tool the fraction of RAM dumps that were analyzable. For instance, we are able to apply the causal analysis approach in 96 of the 100 RAM dumps acquired by Tool 1, and 87 of them contain at least one causal inconsistency. Overall, $4 + 3 + 26 + 9 = 42$ memory dumps could not be analyzed for various reasons. These reasons were as follows: The heap of the pivot program could not be extracted from 39 memory dumps (i.e., the Volatility plugin `pslist` did not find the pivot process). Additionally, in 3 memory dumps the vectors necessary to compute the number of causal inconsistencies could not be found.

5.2. Scenario 2: Looking for open network connections

The investigative goal in Scenario 2 was to find evidence of an open network connection within the memory dumps. The measurement results are summarized in Table 2. As can be seen, almost all memory dumps generated by the four tools contain the open network connection. Memory dumps generated by Tool 1 and Tool 2 include the connection most frequently, while memory dumps generated by Tool 3 include the connection the least. Tool 3 produces the highest number of VAD and causal inconsistencies in this scenario as well. Regarding the VAD analysis, 90 memory dumps are excluded in which the process list could not be acquired correctly (a necessary requirement to compute the number of VAD inconsistencies). Furthermore, 80 memory dumps are excluded from the analysis of causal inconsistencies for multiple reasons. Most often the heap could not be extracted, as the plugin `pslist` did not find the pivot process. Moreover, in 3 of these 80 memory dumps no causal measurement vector could be extracted. In one ideal memory dump the structured analysis fails to find the ssh connection. This seems to be a problem with Volatility and needs further investigation.[4]

5.3. Scenario 3: Seeking an encryption key of VeraCrypt

The investigative goal in Scenario 3 was to find the encryption key of a mounted VeraCrypt volume. The only analysis tool available to us used an unstructured approach. As shown in Table 3, the encryption key

[4] The port of the open connection is printed in the debug output of the plugin `netstat` and the ssh process is in the output of the plugin `pslist`. However, `netstat` does not list the connection in the expected way.

Table 4
Overview of the results of scenario 4, including in how many memory dumps the picture could be found, the number of VAD and causal inconsistencies, as well as in how many memory dumps those inconsistencies were found.

	Tool 1	Tool 2	Tool 3	Tool 4	Ideal Snapshot
Mean acquisition time in min.	2.37	3.18	8.59	3.21	–
Found picture (unstructured)	49/100	49/100	99/100	45/100	10/10
No. analyzable memory dumps	92/100	56/100	78/100	90/100	10/10
No. VAD inconsistent memory dumps	92	56	78	90	0
Total VAD inconsistencies	46092	59686	119946	67146	0
Mean VAD inconsistencies	501	1066	1538	746	0
No. analyzable memory dumps	95/100	97/100	86/100	93/100	10/10
No. causally inconsistent memory dumps	71	73	74	72	0
Total causal inconsistencies	1569	1077	3628	1385	0
Mean causal inconsistencies	16	11	41	15	0

could be extracted in all memory dumps except one generated by Tool 3. Tool 3 also produces the highest number of VAD and causal inconsistencies and has the longest execution time. In relation to the other scenarios, where the mean execution time of Tool 4 was similar to the times of Tool 1 and Tool 2, here it is closer to Tool 3. Overall, $3 + 34 + 20 + 11 = 68$ memory dumps were excluded from the VAD analysis, as in those memory dumps the process list could not be extracted correctly. In contrast, only $2 + 5 + 13 + 4 = 24$ memory dumps were excluded from the causal analysis for several reasons which are as follows: In 19 of those 24 memory dumps the program was not listed in the output of the plugin `pslist` which is used to extract the process. Moreover, 4 memory dumps did not contain a causal measurement vector in the heap of the program. Finally, 1 memory dump was not analyzable at all with Volatility.

5.4. Scenario 4: Finding an opened jpg file

The investigative goal of Scenario 4 was to find evidence of an opened image file. As shown in Table 4, only the memory dumps generated by Tool 3 contain the opened picture in almost all memory dumps, even though Tool 3 takes the longest time to acquire the memory and generates therefore much more inconsistencies than the other tools. In regard to the VAD inconsistencies, 84 memory dumps are missing in the evaluation, as the process list was not acquired correctly. In contrast, only 29 memory dumps are excluded from the causal analysis for various reasons. In particular, in 8 of the 29 dumps the process list was not acquired correctly. Moreover, the pivot program was not listed in the process list, which was acquired with the Volatility plugin `pslist`, of 17 memory dumps. Finally, 1 memory dump did not contain the pivot process at all. In all mentioned cases, the heap of the pivot program could not be extracted and analyzed in relation to inconsistencies. Additionally, 3 memory dumps did not contain a causal measurement vector and for this reason, no inconsistencies could be found.

6. Discussion

This section first discusses the technical aspects of the evaluation. Then, the procedural aspects of those results are explained.

6.1. Technical aspects

For each scenario, there seems to be a correlation between the memory acquisition time and the number of inconsistencies in the respective memory dump when plotting the acquisition time and the number of inconsistencies, i.e., a tool with a longer execution time produces more VAD and causal inconsistencies. This supports the findings of Ottmann et al. (2024) and Rzepka et al. (2024): With a longer execution time, the pages of the physical memory are more likely to change which leads to a higher amount of inconsistencies. The effect on the retrievability of the target artifact of each scenario are, however, not so clear.

For Scenario 1, there seems to be a relationship between the number of VAD and causal inconsistencies and how often the process can be found using structured analysis when looking at a box plot graph. With a higher amount of inconsistencies, the process is found less often. The result supports our assumption that with a higher amount of inconsistencies, more data structures relevant to the analysis are affected, which means that the artifact is found less frequently in the memory dump using structured analysis. There were, however, no negative effects when unstructured analysis was performed regarding whether the process can be found.

Similar insights can be drawn from Scenario 2. Memory dumps generated by Tool 3, which has the highest amount of both VAD and causal inconsistencies, contain the network connection the least when analyzing the memory dumps using structured analysis (in 90 out of 100 memory dumps). However, as the network connection is still found in 90 % of the times, inconsistencies appear not to have such a bad influence on the analysis method as in other scenarios. Similar to Scenario 1, the unstructured analysis approach is consistently more successful than the structured approach. Also, execution time appears to be correlated with retrievability of information on the network connection.

A similar behavior can be observed for the third scenario as well. The encryption key of the VeraCrypt container can be found in all memory dumps except for one memory dump generated by Tool 3, which is rather peculiar since the tool `aeskeyfind` also uses an unstructured analysis approach by scanning the whole memory dump for possible encryption keys based on entropy. Manual analysis confirmed that this one memory dump in which the key cannot be found seems to be broken, as no process list can be printed, neither with the plugin `pslist` nor with `psscan`. In conclusion, the unstructured analysis approach seems to be more robust against inconsistencies than the structured analysis approach which relies on kernel data structures in the respective memory dumps.

In relation to the results of scenarios 1 to 3, we would expect that the opened picture is either found rarely with a higher amount of inconsistencies or is reliably found for all tools, as the plugin `filescan` is an unstructured analysis method. But rather interestingly, neither is the case when looking at the results of Scenario 4. Contradicting our assumption, Tool 3, which has the highest amount of inconsistencies, is able to generate memory dumps in which the opened picture can be found in almost all of them. Additionally, the memory dumps generated by the other three tools contain the picture in less than 50 % of the overall number of dumps, a rather startling fraction of cases. Therefore, we assume that Tool 3 acquires the pages of the physical memory in another way compared to the other three tools. Unfortunately, Tool 3 is a closed-source tool which makes it hard to investigate. Additionally, Volatility is missing a plugin for a structured analysis approach to find opened files in the memory dump. In conclusion, the results of Scenario 4 need much more detailed consideration in future experiments.

In summary, all tools produce memory dumps with varying quality with respect to retrievability of target artifacts. What can be observed is, however, that the unstructured analysis approach seems to be more robust against inconsistencies. But an unstructured approach gives no context in terms of where this information was found. The number of inconsistencies seems to influence structured analysis methods, as they rely on operating system kernel data structures which are more vulnerable to changes during memory acquisition. Additionally, even though our method to assess the number of inconsistencies does not work for all generated memory dumps, this does not mean that the memory dump is not analyzable at all. While all tools produce dumps with partly impressive numbers of inconsistencies, their success rate regarding unstructured analysis appears relatively good. Still, there appears to be a large quality gap in comparison to ideal instantaneous snapshots. Moreover, we want to highlight that our method, especially the two metrics used to calculate inconsistencies, is a first attempt to measure the quality of a memory dump and may lead to misrepresentations. In particular, more research is needed to verify whether the amount of causal and VAD inconsistencies correlates with inconsistencies in other kernel data structures. Here, the physical location of the pages containing the data structures could be taken into account as well.

6.2. Procedural aspects

This study shows that scenario-based testing is a suitable validation method for memory acquisition tools used in criminal proceedings. We demonstrate that tools show very different performance in acquiring RAM data and not all memory artifacts of interest to law enforcement can be recovered with the same tools. Our scenario-based methodology can satisfy international and legal standards for forensic validation as it presents: *(i)* four memory artifacts of interest to the investigation; *(ii)* four scenarios that justify an investigative decision to perform live forensics; *(iii)* results on tool/method limitations and errors; *(iv)* and a knowledge base for specialized training in memory forensics for technical and legal personnel. Our study shows that selecting the appropriate acquisition tool is time-sensitive and depends on the expertise of the examiner and the specifics of the case, as different tools are cable of acquiring different artifacts and not one tool was able to recover all types of data. It is therefore desirable for tool vendors to adopt scenario-based testing and to provide information about the types of artifacts that can be recovered with their tools.

As mentioned before (see Section 6.1), structured analysis provides the context of the discovered artifact, but data structures used by this method are more vulnerable to inconsistencies. To the contrary, unstructured analysis is more robust against inconsistencies but does not give contextual information. Therefore, in order to comply with legal requirements for validation and to ensure that no evidence is missed, it is beneficial to perform both types of analysis and compare results for different tools.

We note that all live memory acquisitions introduce changes to the system due to the need to get administrative rights, plug in the USB drive, and execute the tool on a running system. A common requirement for the quality of acquisition is the expert to be able to identify any changes on the system caused by the acquisition process. This is however not possible in live memory acquisitions without access to the source code or reverse engineering of each tool. Current tools do not display any information on changes to the system during acquisition. In addition, none of the tools reports errors in acquisition. Due to the sensitivity of live acquisitions, there is no guarantee that the accuracy and integrity of the memory artifacts is preserved. Hashing can ensure that post-acquisition examination analysis does not alter data. However, the hash is no guarantee that all data in memory is acquired, since you cannot generate a hash for a running system to compare with the acquired file (Watson and Jones, 2013).

Scenario 3 and 4 showed that there is no possibility for structured analysis for some artifacts like encryption keys and images. The method of scanning the memory for keys based on entropy provides a long list of possible key patterns but it remains unclear where they are used. Further contextual information from the investigation is needed to assess how the recovered key can be used. An important legal question is how the key is discovered and how it was matched to the specific application.

Images opened in the moment of acquisition can be of interest if the suspect was viewing illegal or incriminating photos like how to make a bomb, or a photo of a crime scene taken by the suspect. In scenario 4, all tools except tool 3 performed poorly on recovering a picture in memory. As tool 3 is a closed-source tool, source code access or reverse engineering is needed to understand how the tool recovered images in RAM. However, this is indicative that dual-tool verification might not be sufficient to validate memory acquisition, and does not clarify why certain tools perform better than others. Nevertheless, the inconsistencies in acquisition emphasize the importance of proper documentation of all investigative actions and justification of the selected tool. Scenario-based testing allows judges and lawyers to be aware of the need to compare multiple tool results and different methods of analysis in memory forensics in order to avoid loss of potentially relevant inculpatory or exculpatory evidence. Memory artifacts can be very useful in investigations where viewing of content or running of certain processes is incriminating. The study can assist law enforcement to counter *My client never knew about this!* defense strategies.

6.3. Limitations

There are some factors that limit the ability to generalize our results. For example, they only relate to the Windows operating system in the measured version. Moreover, in the scenarios only the pivot program to track causal inconsistencies and a program corresponding to the artifact of the scenario are started, which is a very artificial and low workload. Additionally, the acquisition time of the tools is not measured exactly, as the time is dependent on ForTrace++ to recognize the end of the acquisition process in the exact same second. The acquisition time is additionally skewed, as we used nested virtualization for our evaluation which leads to much slower execution times. However, as the prerequisites are the same in each iteration, the comparison of the acquisition times and the respective memory dumps is still valid. It also needs to be noted again that the analysis results regarding the retrievability of a forensic artifact not only depend on the quality of the acquisition tool but also on the quality of the analysis tool.

7. Conclusion

The quality of a memory acquisition tool depends on many factors, some of which have been investigated in this paper. Our scenario-based assessment of memory acquisition tools shows that memory dumps with more inconsistencies are more likely to cause problems when performing structured analyses. Unstructured analyses seem to be not or less influenced. However, we applied an overall low workload. Moreover, our metric does not consider inconsistencies in other kernel data structures and how the inconsistencies relate to each other. In future work, an assessment with additional workload can be performed to investigate the relationship between unstructured analyses and inconsistencies. We conjecture that high workload may negatively affect the success of unstructured methods because the overall analyzability of memory dumps will deteriorate. Besides, different paths to perform a structured analysis of specific artifacts can be implemented to compare the impact of inconsistencies on different data structures as suggested by Pagani and Balzarotti (2019).

The structured analysis methods do not rely on the VAD tree in every possible scenario. Therefore, there is also a need for additional and different consistency indicators to ultimately find a reliable metric for assessing the quality of memory dumps. Moreover, especially the results of Scenario 4 show the need for further research regarding the ability of memory acquisition tools to capture artifacts of opened (image) files.

Scenario-based testing can assist legal professionals to assess where memory acquisition can advance the investigation and which limitations of forensic memory acquisition tools can impact the quality of volatile digital artifacts. Memory forensics can be very useful in serious crime investigations like possession or dissemination of illegal content, illegal web hosting, and criminal use of encryption. Thus, law enforcement should increase their expertise in multiple memory acquisition tools validation and results interpretation.

Acknowledgments

We would like to thank Dennis Wolf for his help with setting up ForTrace++ for the evaluation.

Appendix

A. Software Versions

Table 5
Evaluated RAM acquisition tools.

Name	Version	Pseudonym	Open/Closed Source	Commercial/free
Belkasoft RAM Capturer	downloaded 07.02.2024	Tool 1	closed	free
FTK Imager	v4.7.1	Tool 2	closed	free
Magnet RAM Capture	v1.2.0	Tool 3	closed	free
WinPmem	winpmem_mini v4.0 RC2	Tool 4	open	free

Table 6
Scenarios and the corresponding analysis tools used in our evaluation.

	Use case scenario	Structured analysis tool	Unstructured analysis tool
1	Executed software	Volatility *pslist*	Volatility *psscan*
2	Active ssh connection	Volatility *netstat*	Volatility *netscan*
3	Open VeraCrypt container	–	*aeskeyfind*
4	Open files (jpg)	–	Volatility *filescan*

References

Anderson, R.J., Barton, C., Böhme, R., Clayton, R., van Eeten, M.J.G., Levi, M., Moore, T., Savage, S., 2013. Measuring the cost of cybercrime. In: Böhme, R. (Ed.), The Economics of Information Security and Privacy. Springer, pp. 265–300. https://doi.org/10.1007/978-3-642-39498-0_12.

Barnes, E., 2021. Mitigating the risks of fileless attacks. Computer Fraud & Security 2021 20. https://doi.org/10.1016/S1361-3723(21)00044-0, 20.

Böhm, F., Englbrecht, L., Friedl, S., Pernul, G., 2021. Visual decision-support for live digital forensics. In: 2021 IEEE Symposium on Visualization for Cyber Security (VizSec), pp. 58–67. https://doi.org/10.1109/VizSec53666.2021.00012.

Campbell, W., 2014. Volatile memory acquisition tools - a comparison across taint and correctness. In: Proceedings of the 11th Australian Digital Forensics Conference, ADF 2013, pp. 10–19.

Case, A., Richard III, G.G., 2017. Memory forensics: The path forward. Digit. Invest. 20, 23–33. https://doi.org/10.1016/j.diin.2016.12.004.

Casey, E., 2007. What does "forensically sound" really mean? Digit. Invest. 4, 49–50. https://doi.org/10.1016/j.diin.2007.05.001. https://linkinghub.elsevier.com/retrieve/pii/S1742287607000333.

Casey, E., Rose, C.W., 2010. Forensic analysis. In: Casey, E. (Ed.), Handbook of Digital Forensics and Investigation. Elsevier, pp. 21–62.

Cook, R., Evett, I., Jackson, G., Jones, P., Lambert, J., 1998. A hierarchy of propositions: deciding which level to address in casework. Sci. Justice 38, 231–239. https://doi.org/10.1016/S1355-0306(98)72117-3. https://www.sciencedirect.com/science/article/pii/S1355030698721173.

European Network of Forensic Science Institutes (ENFSI), 2015. Best practice manual for the forensic examination of digital technology. http://enfsi.eu/wp-content/uploads/2016/09/1._forensic_examination_of_digital_technology_0.pdf.libraryCatalog: enfsi.eu.

Gruhn, M., Freiling, F.C., 2016. Evaluating atomicity, and integrity of correct memory acquisition methods. Digit. Invest. 16, S1–S10. https://www.sciencedirect.com/science/article/pii/S1742287616000049.

Halderman, J.A., Schoen, S.D., Heninger, N., Clarkson, W., Paul, W., Calandrino, J.A., Feldman, A.J., Appelbaum, J., Felten, E.W., 2009. Lest we remember: cold-boot attacks on encryption keys. Commun. ACM 52, 91–98. https://doi.org/10.1145/1506409.1506429.

Hargreaves, C., Chivers, H., 2008. Avoiding live imaging of large encrypted volumes by recovering keys from memory.

Hargreaves, C., Nelson, A., Casey, E., 2024. An abstract model for digital forensic analysis tools - a foundation for systematic error mitigation analysis. Forensic Sci. Int.: Digit. Invest. 48, 301679. https://doi.org/10.1016/j.fsidi.2023.301679. https://www.sciencedirect.com/science/article/pii/S2666281723001981.

Inoue, H., Adelstein, F., Joyce, R.A., 2011. Visualization in testing a volatile memory forensic tool. Digit. Invest. 8, S42–S51. https://doi.org/10.1016/j.diin.2011.05.006.

International Criminal Police Organization (Interpol), 2019. Global guidelines for digital forensics laboratories. https://www.interpol.int/en/content/download/13501/file/INTERPOL_DFL_GlobalGuidelinesDigitalForensicsLaboratory.pdf.

International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC), 2012. ISO/IEC 27037 eForensics Guidelines for identification, collection, acquisition and preservation of digital evidence. https://www.iso27001security.com/html/27037.html.

ISO/IEC, 2017. ISO/IEC 17025:2017 General requirements for the competence of testing and calibration laboratories. http://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/06/69/66912.html.

Kent, K., Chevalier, S., Grance, T., Dang, H., 2006. NIST SP 800-86, Guide to integrating forensic techniques into incident response. https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-86.pdf.

Latzo, T., Palutke, R., Freiling, F.C., 2019. A universal taxonomy and survey of forensic memory acquisition techniques. Digit. Invest. 28, 56–69. https://doi.org/10.1016/j.diin.2019.01.001.

Ligh, M.H., Case, A., Levy, J., Walters, A., 2014. The art of memory forensics: Detecting malware and threats in Windows, Linux, and Mac memory. Wiley.

Lyle, J.R., Guttman, B., Butler, J.M., Sauerwein, K., Reed, C., Lloyd, C.E., 2022. NISTIR 8354-draft: Digital investigation techniques: A NIST scientific foundation review. . https://doi.org/10.6028/NIST.IR.8354-draft.

Mattern, F., 1989. Virtual time and global states of distributed systems. C.M. (Ed.) et al., Proc. Workshop on Parallel and Distributed Algorithms, North-Holland/Elsevier. pp. 215–226. In: Yang, Z., Marsland, T.A. (Eds.), Global States and Time in Distributed Systems, vol. 1994. IEEE, pp. 123–133.

McDown, R., Varol, C., Carvajal, L., Chen, L., 2015. In-depth analysis of computer memory acquisition software for forensic purposes. J. Forensic Sci. 61. https://doi.org/10.1111/1556-4029.12979.

Mocas, S., 2004. Building theoretical underpinnings for digital forensics research. Digit. Invest. 1, 61–68. https://doi.org/10.1016/j.diin.2003.12.004. http://www.sciencedirect.com/science/article/pii/S1742287603000057.

Montasari, R., 2017. A standardised data acquisition process model for digital forensic investigations. Int. J. Inf. Comput. Secur. 9, 229. https://doi.org/10.1504/IJICS.2017.085139. http://www.inderscience.com/link.php?id=85139.

National Institute of Standards and Technology, 2017. NIST disk imaging. https://www.cftt.nist.gov/disk_imaging.html. Last update March 2024, last accessed 10.10.2024.

National Institute of Standards and Technology, 2022. Digital investigation techniques: a NIST scientific foundation review. https://www.nist.gov/forensic-science/digital-investigation-techniques-nist-scientific-foundation-review.lastModified:2022-07-20 T10:20-04:00.

Osborne, G., 2013. Memory forensics: Review of acquisition and analysis techniques. Technical Report DSTO-GD-0770. Australian Department of Defence, Defence Science and Technology Organisation.

Ottmann, J., Breitinger, F., Freiling, F.C., 2024. An experimental assessment of inconsistencies in memory forensics. ACM Trans. Priv. Secur. 27 (2), 1–2:29. https://doi.org/10.1145/3628600.

Pagani, F., Balzarotti, D., 2019. Back to the whiteboard: A principled approach for the assessment and design of memory forensic techniques. In: 28th USENIX Security Symposium (USENIX Security 19). USENIX Association, Santa Clara, CA, pp. 1751–1768. https://www.usenix.org/conference/usenixsecurity19/presentation/pagani.

Pagani, F., Fedorov, O., Balzarotti, D., 2019. Introducing the temporal dimension to memory forensics. ACM Trans. Priv. Secur. 22 (9), 1–9. https://doi.org/10.1145/3310355, 21.

Richard III, G.G., Roussev, V., 2005. Scalpel: a frugal, high performance file carver. In: Refereed Proceedings of the 5th Annual Digital Forensic Research Workshop, DFRWS 2005, Astor Crowne Plaza. New Orleans, Louisiana, USA, August 17-19, 2005. URL: http://www.dfrws.org/2005/proceedings/richard_scalpel.pdf.

Rzepka, L., 2024. A scenario-based quality assessment of memory acquisition tools and its investigative implications: Experimental data. https://doi.org/10.5281/zenodo.14260323.

Rzepka, L., Ottmann, J., Freiling, F., Baier, H., 2024. Causal inconsistencies are normal in windows memory dumps (too). Digital Threats. https://doi.org/10.1145/3680293.

Sammons, J., 2015. Collecting evidence. In: The Basics of Digital Forensics. Elsevier, pp. 47–64. https://linkinghub.elsevier.com/retrieve/pii/B9780128016350000048.

Schatz, B.L., 2007. Bodysnatcher: towards reliable volatile memory acquisition by software. Digit. Invest. 4, 126–134. https://doi.org/10.1016/j.diin.2007.06.009.

Schuster, A., 2006. Pool allocations as an information source in Windows memory forensics. In: IT-incident Management & IT-Forensics - IMF 2006. Gesellschaft für Informatik e. V., Bonn, pp. 104–115.

Stoykova, R., 2023. The right to a fair trial as a conceptual framework for digital evidence rules in criminal investigations. Comput. Law Secur. Rep. 49, 105801. https://doi.org/10.1016/j.clsr.2023.105801. https://www.sciencedirect.com/science/article/pii/S0267364923000110.

Stoykova, R., Franke, K., 2023. Reliability validation enabling framework (RVEF) for digital forensics in criminal investigations. Forensic Sci. Int.: Digit. Invest. 45, 301554. https://doi.org/10.1016/j.fsidi.2023.301554. https://www.sciencedirect.com/science/article/pii/S266628172300063X.

Vömel, S., Freiling, F.C., 2012. Correctness, atomicity, and integrity: Defining criteria for forensically-sound memory acquisition. Digit. Invest. 9, 125–137. https://doi.org/10.1016/j.diin.2012.04.005.

Vömel, S., Stüttgen, J., 2013. An evaluation platform for forensic memory acquisition software. Digit. Invest. 10, S30–S40. https://doi.org/10.1016/j.diin.2013.06.004.

Watson, D., Jones, A., 2013. Case processing. In: Digital Forensics Processing and Procedures. Elsevier, pp. 367–420. https://doi.org/10.1016/B978-1-59749-742-8.00009-1. https://linkinghub.elsevier.com/retrieve/pii/B9781597497428000091.

Wolf, D., Göbel, T., Baier, H., 2024. Hypervisor-based data synthesis: on its potential to tackle the curse of client-side agent remnants in forensic image generation. Forensic Sci. Int. Digit. Investig. 48, 301690. https://doi.org/10.1016/j.fsidi.2023.301690.