Pypette: A Framework for the Evaluation of Live Digital Forensic Acquisition Techniques

B. Lempereur, M. Merabti and Qi Shi

School of Computing and Mathematical Sciences
Liverpool John Moores University
L3 3AF
e-mail : b.lempereur@2006.ljmu.ac.uk, {m.merabti,q.shi}@ljmu.ac.uk

Abstract

With the increasing scale of digital forensic investigations, there is a need for approaches that are capable of reducing the quantities of data forensic examiners are required to search. As this trend continues, traditional quiescent digital forensic analysis is in some cases becoming impractical; examiners must often rely on an in-situ investigation of the live computing environment. Numerous approaches to live digital forensic evidence acquisition have been proposed in the literature, but relatively little attention has been paid to the problem of identifying how the effects of these approaches, and their improvements over other techniques, can be evaluated and quantified. In this paper, we present Pypette, a novel framework enabling the automated, repeatable analysis of live digital forensic acquisition techniques.

Keywords

Digital forensics, live digital forensics, experimental computer science.

1 Introduction

Scale is a pervasive problem in Digital Forensics. In 1999, McKemmish (McKemmish 1999) published a report for the Australian Institute of Criminology in which he identified the volume of data and prevalence of digital devices as future research issues. More than ten years later, this is still the case (Haggerty & Taylor 2007; Distefano & Me 2008; Richard III & Roussev 2006). The growth in static storage has been "tremendous," and the number of embedded devices that could feasibly be used to participate in crime, often equipped with their own proprietary operating systems, is increasing (Mohay 2005).

Traditional approaches to digital forensic investigation are quiescent, in that they require the examiner to power-off the subject machine and make a bit-for-bit copy of non-volatile storage media before proceeding with any examination (Sammes Anthony & Jenkinson 2007). As the nature and scale of computing systems continues to change this approach is, in some cases, impractical; examiners must often rely on an in-situ investigation of the live computing environment (F Adelstein 2006; Hay et al. 2009).

Live digital forensics presents unique challenges with respect to maintaining forensic soundness, but also offers the ability to examine information that is unavailable to quiescent analysis, namely the operational state of the system. The evidence gained from this approach, however, lacks credibility (Wang et al. 2009). This problem is exacerbated by the possibility of malicious software altering the output from live digital forensic software (Rutkowska 2007). Despite this, there has been no systematic attempt to examine the side effects and accuracy of live digital forensic approaches to evidence acquisition.

We believe that live digital forensic evidence, which describes how a computer was actually used, is a useful addition to inferences drawn from artefacts in documents and files, and that if employed correctly it can be a significant aid to an investigation. In this paper, we propose a novel approach to evaluating the effects and accuracy of live digital forensic acquisition techniques. Where existing approaches have focused on evaluation based on a percentage of memory change before and after acquiring live forensic evidence, we consider the accuracy and effects of methods in terms of the artefacts forensic examiners actually need to extract from systems, and the mechanisms they use for achieving this. The result of this work is Pypette, a framework for performing automated, repeatable experiments on live digital forensic acquisition techniques.

The rest of this paper is organised as follows. General concepts in live digital forensics and initial work towards evaluating live digital forensics are discussed in §2. In §3 we present the design of our live forensic experimentation framework. The results from our implementation and feasibility assessment are given in §4. We conclude the paper in §5 with a summary of our approach and directions for further work.

2 Related Work

When a system cannot be powered off, because of legal, technical, or other reasons, analysts must perform a live forensic analysis. Regardless of whether the examiner is taking a quiescent or live approach, acquiring and analysing evidence in a forensically sound manner is paramount to the success of an investigation and the acceptance of evidence in court. To be considered forensically sound, processes must meet the following criteria (McKemmish 2008):

- The meaning of electronic evidence should not be altered by the process.
- Any errors should be identified and “satisfactorily explained.”
- Processes should be available for, and stand up to, independent verification.
- Analysts should have sufficient and relevant experience.

Casey (Casey 2007) states that a puritanical approach to forensic soundness, where any alteration to evidence renders it inadmissible, is unhelpful. When performing a live analysis, the state of a computer is necessarily altered, and this is consistent with other forensic disciplines.

Despite numerous publications, high-profile coverage (Richard III & Roussev 2006; F Adelstein 2006; Carrier 2006; Hay et al. 2009), and proven value (Hargreaves & Chivers 2008; Schatz 2007), the inadmissibility of evidence captured using live techniques remains the prevailing attitude among digital forensic researchers and practitioners (7Safe Information Security 2007). We believe this to be a combination of a social and a technical problem. Until the necessary techniques are developed to allow first-responders, who may not be experts, to perform live analysis it is likely (perhaps reasonable) that the techniques will remain a niche area of digital forensic practice, and consequently research.

The techniques used to acquire artefacts during live forensic analysis will be subject to examination and review when their conclusions are presented in court. Sutherland et al. (Sutherland et al. 2008) conducted an empirical study to determine the impact of live forensic tools on subject systems, using a limited set of measurements, including the amount of memory altered by the live acquisition technique, its impact on the Microsoft Windows environment, and dependency on dynamic libraries.

During their investigation into volatile memory forensics (Walters & Petroni 2007) conducted a very limited analysis of the impact of live memory acquisition tools on Microsoft Windows XP hosts. The decision to measure the effectiveness of memory acquisition tools as a percentage of bytes changed on the system after their execution is short-sighted. Once an image of memory and the page file have been captured, to the extent that the size of the page file is not automatically increased by the operating system, it is inconsequential how much memory is altered during live analysis. Therefore, a reliable way to detect changes would be to compare the system state before imaging with the image captured by the acquisition tool.

A more thorough examination of live digital forensic memory acquisition techniques was presented in (Inoue et al. 2011), using visualisation mechanisms to check for systematic errors. This approach was effective; however, they note a difficulty in obtaining "ground-truth" images of machine state against which to evaluate their method. We aim to address this problem by approaching the evaluation of these techniques as an experiment, using virtualisation to host the testing environment, allowing for the repeatable collection of a ground-truth image from the virtual machine manager.

3 Pypette

Existing research relies on the establishment of, and comparison to, a baseline figure of memory change as a function of time (Walters & Petroni 2007), or counting the number of memory pages altered by the live digital forensic acquisition technique. We believe this view oversimplifies the problem, and that the operational state of a system after acquisition is irrelevant so long as the captured state accurately reflects the original machine before a live digital forensic intervention. Therefore, we propose that a meaningful result can be obtained by viewing the process in terms of the artefacts examiners are attempting to extract using live analysis.

To gain a complete picture of accuracy and effectiveness, these artefacts should be extracted from the state captured by live forensic acquisition and compared to the resulting state of the system, its initial state, and the state the machine would be in had no intervention occurred. Finally, since computer systems are, in practice, rarely deterministic, this comparison should be performed multiple times; statistical analysis of the results will allow us to determine the most likely outcome of different types of acquisition and analysis in various scenarios.

Our goal when creating the framework is to provide a standard set of techniques that will enable experts to quickly design and execute new experiments. Parallel to this, a repository of scenarios for live evidence acquisition, provided as virtual machines, should be developed to allow for comparable results from different evidence acquisition methods. While existing research has shown that achieving this is difficult (Garfinkel et al. 2009), we believe that some degree of commonality between comparisons will be helpful for future live digital forensic research.

For the remainder of this section, we first discuss the implementation of our framework and the interface for implementing experiments. We conclude the section by presenting the mechanism we use to evaluate the accuracy of a live forensic technique, and the effects it has on its environment.

3.1 Conducting Live Forensic Experiments

Pypette is a framework for conducting experiments into live digital forensic acquisition and analysis techniques following the principles outlined above. From a developer's perspective, the framework is structured as three concepts: techniques, actors, and analysts. Experiments are written as Python programs, inheriting behaviour from the framework in a manner that lets developers "pick and choose" the most appropriate technique, actor, and analyst for their scenario, extending the framework where functionality is missing or complicated behaviour desired.

The technique acts as a factory, supplying instances of actors and analysts to the framework that will carry out the majority of the work. They should remain unaware of the specific details of the environment in which they will operate. All a technique need do is to check the feasibility of execution on a specific virtual machine, that is, whether the configuration is sufficient to allow the actor to perform its function without failure, and if the result will be meaningful to the analyst. Actors are responsible for intervening in the execution of a virtual machine, applying a live digital forensic acquisition or analysis technique, and extracting the results onto the host system. Analysts extract artefacts from, and collate the results of, the intervention, which is then fed into a statistical model that determines the accuracy of the technique.

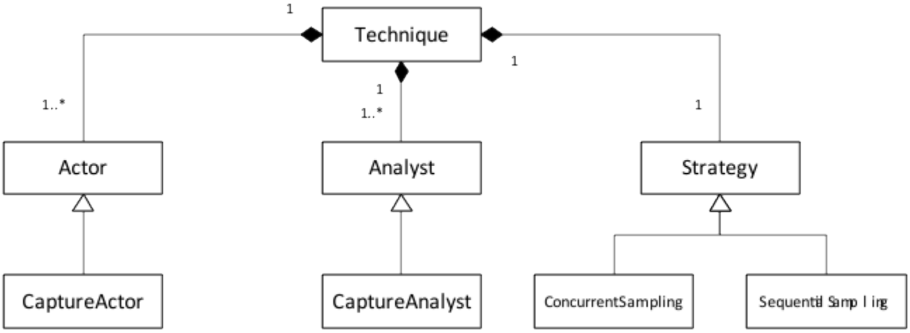

Figure 1: Component hierarchy of the experiment interface.

In our initial work, we assumed that control and live virtual machines should always run in parallel. Asides from more complicated cases where, for example, a suite of virtual machines are used to conduct an experiment, this may not always be the best approach even for single-machine cases. Instead, we allow the developer to determine the appropriate sampling strategy, selecting from a predefined library, or implementing their own where appropriate.

The responsibility of the sampling strategy is to coordinate the sampling of subject and control machines within the overall scope of an experiment. How this is arranged affects the functionality of both actor and analyst, to the extent that some sampling strategies may be incompatible with some techniques. With this in mind, we have provided a general interface for sampling strategies, and implementations of both concurrent and sequential strategies for the single-machine experiment case. This is sufficient for the majority of experiments that we foresee being conducted using the framework. Figure 1 shows the hierarchy that comprises a technique, along with the implementations of actors, analysts, and sampling strategies that are in-place for use by developers.

To provide the emulation platform for our experiments, we use the QEMU virtual machine manager, running in either kernel-assisted full-virtualisation, or user-land emulation modes. A software agent that communicates with the host using the virtual serial port controls actions within the virtual machine. These actions include spawning and capturing the output of processes, locating mounted volumes using a simple UUID-token placed in the root of the file system, and ensuring that correct ejection of removable media.

Figure 2 shows the communication architecture between the Pypette framework, the virtual-machine manager, and the in-host virtual machine agent. All messages between server and client are exchanged using a line-oriented, JSON-based protocol over UNIX domain sockets.

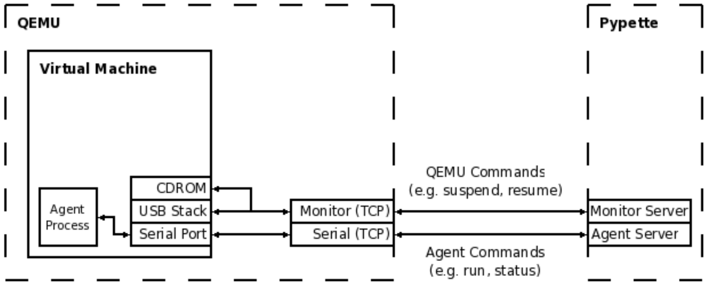

Figure 2: Virtual machine emulation and communication with the framework.

To allow for repeatable experiments, machines are instantiated from an image of their state that has been migrated to a file on disk. A machine state defines the situation for an experiment, and testing a technique in a variety of situations gives an impression of how we can expect it to perform. While a migration image contains the contents of a machine's memory, we must also provide the stored media, namely hard-disk images and removable media that contribute to a situation. For this purpose, we create and use only copy-on-write clones of the exemplar machine's storage, leaving the original media intact. These duplicates are also available for analysis, and in future work we will investigate the application of automated forensic storage analysis techniques to provide further information about the effects of live digital forensic acquisition methods.

3.2 Analysing the Results of Experiments

We employ a novel technique to evaluate the accuracy and forensic soundness of evidence acquisition methods that compares the imperfect, in-situ view available to live digital forensics, against a perfect system-image captured using a virtual machine manager. Figure 3 shows the comparisons we perform, and the information they provide about the live forensic technique. The dashed lines represent the different samples that are taken from the control and subject machines, with arrows indicating the comparison performed between them. By comparing the state of the machine at these three different points, we can:

- Measure the accuracy figure for the live digital forensic acquisition technique, by determining how closely its output reflects the initial state of the system.

- Measure forensic soundness through analysing the state in which the acquisition technique leaves the system, compared to both its initial state and the state it would be in had no intervention occurred.

- Establish a disturbance figure for the virtual machine, had no intervention occurred, a baseline against which we determine the accuracy and forensic soundness of the technique.

By collecting this information from a scripted experiment over a number of iterations, we can build a picture of how we expect a live digital forensic technique to perform in a given scenario.

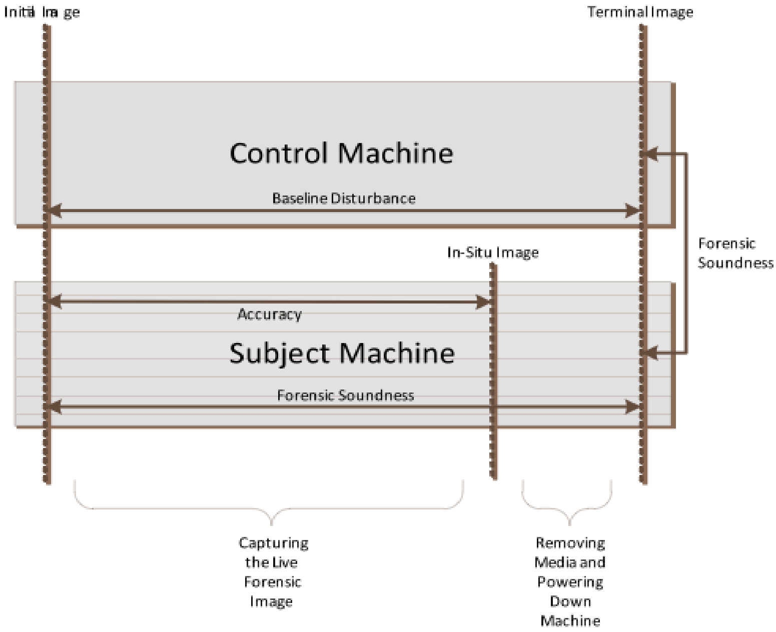

Figure 3: Sampling and analysis strategy for single-machine experiments.

We achieve support for the automated extraction of evidence from acquired memory images through an interface to the Volatility suite of memory analysis techniques. As part of our framework, we use this interface to provide the analysts for single-machine memory and artefact capture techniques. It is also available to developers implementing their own analysts.

4 Results

To determine the efficacy of our approach to live digital forensic experimentation, we designed a small-scale test using the memory acquisition tool “mdd” (ManTech International Corporation 2009). The platform for the experiment was a clean installation of Microsoft Windows XP SP3, on the i386 platform, configured with 1GB RAM, a graphics card, and no networking support. The sampling procedure was:

1. Introduce a copy of the acquisition tool to the subject machine as a CDROM image.

2. Instruct the in-situ agent to execute the capture tool within the subject machine, saving the contents to an additional drive.

3. Suspend the execution of both virtual machines and take the final memory snapshots.

4. Attach the additional drive image to the host machine, and copy the in-situ memory capture to a temporary location.

We selected to examine process table entries and their associated handles in the initial and in-situ captures as, with the machine being isolated from the network, we expected to see the highest degree of volatility in these areas. To analyse the memory images captured during the experiment, we developed an extension to the single machine capture analyst that exported results to JSON documents for later study.

The technique behaved consistently across the 211 samples we gathered, with no significant differences in machine state between experiment iterations. Table 1 shows an overview of the number of processes and handles gathered from each sample. It is interesting to note that the number and identity of processes remained constant across all samples, with a relatively small fluctuation in their associated file handles. Experiments took an average of 256 seconds to complete, with a minimum of 186 seconds, and a maximum of 329 seconds. Given that the purpose of the experiment was as a feasibility assessment of our approach, we believe this outcome is promising.

	Mean Processes	Min Processes	Max Processes	Mean Handles	Min Handles	Max Handles
Initial	21	21	21	4397	4397	4397
In-Situ	66	66	66	4578	4566	4588
Control	21	21	21	4305	4190	4370

Table 1: Presence of process table entries and their associated file handles in memory images, with the in-situ image extracted using "mdd" (n = 211).

There is insufficient space in this paper to present the results in full; however, we intend to publish further examinations of live digital forensic techniques, particularly those relating to memory extraction, in addition to examining the effects of turbulent, network connected, and malware-infected environments.

5 Conclusions and Further Work

In this paper, we have presented Pypette, a novel framework enabling the automated, repeatable analysis of live digital forensic acquisition techniques. Where existing approaches have focused on evaluation based on a percentage of memory change before and after acquiring live forensic evidence, we consider the accuracy and effects of methods in terms of the artefacts forensic examiners actually need to extract from systems, and the mechanisms they use for achieving this. Initial results have shown that our framework is capable of conducting repeatable experiments and generating consistent results.

In future work we will refine the model used to analyse the results of live digital forensic acquisition, in addition to providing further templates to ease the process of designing experiments for our framework. Parallel to this, we will conduct a series of experiments on various live digital memory acquisition techniques in the presence of turbulent, networking connected, and malware-infected environments.

We also intend to investigate the possibility of extending the framework to provide a training platform for first-responders. It would be possible to allow the user to interact with the live machine, either providing an interface to interact with machine hardware, or directly attaching devices from the host to the guest machine. Scripted training sessions would allow the collection of detailed metrics regarding user-performance and the accuracy of collected evidence.

6 References

7Safe Information Security, 2007. Good Practice Guide for Computer-Based Electronic Evidence.

Adelstein, F, 2006. Live forensics: diagnosing your system without killing it first. Communications of the ACM, 49(2), p.66. Available at: http://portal.acm.org/citation.cfm?id=1113034.1113070.

Carrier, B.D., 2006. Risks of live digital forensic analysis. Communications of the ACM, 49(2), pp.56–61. Available at: http://portal.acm.org/citation.cfm?id=1113034.1113069.

Casey, E., 2007. What does "forensically sound" really mean? Digital Investigation, 4(2), pp.49-50.

Distefano, A. & Me, G., 2008. An overall assessment of mobile internal acquisition tool. Digital Investigation, 5, pp.121–127. Available at: http://linkinghub.elsevier.com/retrieve/pii/S174228760800042X.

Garfinkel, S. et al., 2009. Bringing science to digital forensics with standardized forensic corpora. Digital Investigation, 6, p.S2-S11. Available at: http://linkinghub.elsevier.com/retrieve/pii/S1742287609000346 [Accessed August 4, 2011].

Haggerty, J. & Taylor, M., 2007. FORSIGS: forensic signature analysis of the hard drive for multimedia file fingerprints H. Venter et al., eds. New Approaches for Security, Privacy and Trust in Complex Environments, 232, pp.1–12. Available at: http://www.springerlink.com/index/21478KR877478805.pdf.

Hargreaves, C. & Chivers, H., 2008. Recovery of encryption keys from memory using a linear scan. In Proceedings of the 2008 Third International Conference on Availability, Reliability and Security. IEEE Computer Society, pp. 1369–1376. Available at: http://portal.acm.org/citation.cfm?id=1371602.1371819.

Hay, B., Bishop, M. & Nance, K., 2009. Live Analysis: Progress and Challenges. IEEE Security and Privacy, 7(2), pp.30–37. Available at: http://www.computer.org/portal/web/csdl/doi/10.1109/MSP.2009.43.

Inoue, H., Adelstein, Frank & Joyce, R. a., 2011. Visualization in testing a volatile memory forensic tool. Digital Investigation, 8, p.S42-S51. Available at: http://linkinghub.elsevier.com/retrieve/pii/S1742287611000302 [Accessed August 4, 2011].

ManTech International Corporation, 2009. MDD Physical Memory Acquisition. Available at: http://sourceforge.net/projects/mdd/.

McKemmish, R., 1999. What is forensic computing. Trends and Issues in Crime and Criminal Justice, 118. Available at: http://aic.gov.au/documents/9/C/A/{9CA41AE8-EADB-4BBF-9894-64E0DF87BDF7}ti118.pdf.

McKemmish, R., 2008. When is Digital Evidence Forensically Sound? Advances in Digital Forensics IV, pp.3–15. Available at: http://www.springerlink.com/index/048J747850234355.pdf.

Mohay, G., 2005. Technical challenges and directions for digital forensics. In Systematic Approaches to Digital Forensic Engineering, 2005. First International Workshop on. pp. 155–161. Available at: http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Technical+challenges+and+directions+for+digital+forensics#0.

Richard III, G.G. & Roussev, V., 2006. Next-generation digital forensics. Communications of the ACM, 49(2), p.80. Available at: http://portal.acm.org/citation.cfm?id=1113074.

Rutkowska, J., 2007. Beyond the CPU: Defeating hardware based RAM acquisition. Proceedings of BlackHat DC 2007. Available at: http://lollobox.org/lollobox/raw-attachment/wiki/TheProject/DoesntProtectFrom/bh-dc-07-Rutkowska-up.pdf.

Sammes Anthony, J. & Jenkinson, B., 2007. The Treatment of PCs. In Forensic Computing. London: Springer, pp. 277-299.

Schatz, B., 2007. BodySnatcher: Towards reliable volatile memory acquisition by software. Digital Investigation, 4, pp.126-134. Available at: http://www.citeulike.org/user/jksahani/article/6863807.

Sutherland, I. et al., 2008. Acquiring volatile operating system data tools and techniques. ACM SIGOPS Operating Systems Review, 42(3), pp.65–73. Available at: http://portal.acm.org/ citation.cfm?id=1368516.

Walters, A. & Petroni, N., 2007. Volatools: integrating volatile memory forensics into the digital investigation process. Black Hat DC. Available at: http://scholar.google.co.uk/scholar?q=volatools&hl=en&btnG=Search&as_sdt=2001&as_sdtp=on#0.

Wang, L., Zhang, R. & Zhang, S., 2009. A Model of Computer Live Forensics Based on Physical Memory Analysis. In 2009 First International Conference on Information Science and Engineering. IEEE, pp. 4647-4649. Available at: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5454440 [Accessed August 8, 2011].