Chapter 9
Pypette:
A Platform for the Evaluation of Live Digital Forensics

Brett Lempereur
School of Computing and Mathematical Sciences, Liverpool John Moores University, Liverpool, UK

Madjid Merabti
School of Computing and Mathematical Sciences, Liverpool John Moores University, Liverpool, UK

Qi Shi
School of Computing and Mathematical Sciences, Liverpool John Moores University, Liverpool, UK

ABSTRACT

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. Any perturbation of a live operating system by a forensic examiner will have far-reaching effects on the state of the system being analysed. 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, the authors present Pypette, a novel platform enabling the automated, repeatable analysis of live digital forensic acquisition techniques.

INTRODUCTION

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. 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 (Adelstein, 2006; Hay, Nance, & Bishop, 2009).

Live digital forensics presents unique challenges with respect to maintaining forensic soundness, but also offers the ability to examine

DOI: 10.4018/978-1-4666-4006-1.ch009

information that is unavailable to quiescent analysis, namely the operational state of the system. The evidence gained from this approach, however, lacks credibility (Wang, Zhang, & Zhang, 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.

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 (Distefano & Me, 2008; Haggerty & Taylor, 2007; Richard & 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).

Figure 1 illustrates the scale of the problem through a visualisation of the interactions that occur between processes and files in a typical Microsoft Windows machine during a 25-minute period. Vertices in the diagram represent the files and processes on the system, with edges indicating process creation and operations performed on files. There is a high-degree of interdependence, and any perturbation of a live operating system by a forensic examiner will have far-reaching effects on the state of the system being analysed.

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

Figure 1. Visualisation of inter-process communication and file handle interaction