An Open-Source Forensics Platform

R. Koen and M. S. Olivier

Abstract— Digital forensics plays a crucial part in the investigation of crimes involving electronic equipment. Masses of digital evidence collected at a crime scene will have to be analyzed by digital forensics experts in an attempt to discover how a digital crime was committed and by whom. This is a labour-intensive and time-consuming process which can be improved using specially designed digital tools. This paper proposes an open-source forensics platform that may be used as a base for other digital forensics projects. The proposed forensics platform may be used by researchers as a base to develop digital forensics research prototypes and by industry to conduct digital investigations after it has become apparent that a digital crime has been committed. This aim of proposed platform project is enable researchers to develop forensic prototypes more rapidly and help to ensure the quality of the forensics tools making use of the platform.

Index Terms—Digital forensics, forensics platform architecture, forensic system design.

I. INTRODUCTION

Digital forensics plays a crucial part in the investigation of crimes involving electronic equipment. Digital evidence collected at a crime scene will have to be analyzed and connections between the recovered information, physical entities and physical events need to be made and proven. Investigators of digital crimes usually have a lot of complex questions to answer in a short amount of time [3]. The amount of time taken up by the investigations may be attributed to the complexity and mass of digital evidence collected. As computing technology improves and the storage capacities of digital devices increases, it may not be feasible to manually inspect devices with sizes ranging in gigabytes to terabytes [1]. Examiners therefore need to constantly improve their collection and examination methodology and tools in an effort to improve their efficiency [9].

This paper will propose an open-source digital forensics platform that may be used by academics to develop digital forensics prototypes and by industry to perform digital investigations.

The remainder of the paper is structured as follows. Section 2 and section 3 will discuss the need for a forensic platform while section 4 will expand on the functionality that needs to be implemented by such a platform. Section 5 will discuss a proposed architecture that will address forensic needs specified in literature. Section 6 will discuss the implementation of the specified architecture, while section 7 will discuss future work that needs to be conducted. This paper will finally be concluded with a brief summary in section 8.

R Koen is a member of ICSA, University of Pretoria, South-Africa (e-mail:renico@lantic.net).

M.S. Olivier is a member of ICSA and with the Department of Computer Science, University of Pretoria, South-Africa (e-mail: martin@mo.co.za).

II. THE NEED FOR A FORENSICS PLATFORM

Different digital forensics tools are available to help forensic examiners to perform forensic investigations. Although these tools may have been tested and proven to perform a specific task well in a specific environment [19], it cannot be assumed that these tools can perform equally well when used in conjunction with other digital tools. This is definitely a problem, since digital investigators tend to make use of a collection of tools to perform their investigations [5]. These tools were not necessarily designed to function together as a single cohesive unit to perform the acquisition and analysis of digital evidence which may lead to irregularities and inconsistencies in gathered evidence which may ultimately lead to the exclusion of the digital evidence in question from a case due to a lack of trustworthiness.

Walker [18] informs us that it is critical that all collected digital evidence is in an uncompromised state; a single file with a timestamp later than the time of evidence acquisition may lead to the situation where the evidence is excluded from a case due to inconsistencies. It is therefore crucial that the chosen tools used for the acquisition and analysis of data is able to work together without compromising the digital evidence in any way.

Some investigators may even attempt to create their own acquisition and analysis tools [17]. These noble attempts at creating tools that increase investigation efficiency are usually rewarded by scepticism in court due to the fact that it is extremely difficult to prove that a custom-made digital forensics tool is forensically sound.

Rogers and Seigfried [15] inform us that the U.S. Supreme court supplied certain criteria in the Daubert vs. Merrel case that may be used as guidelines by courts to determine whether or not evidence is admissible in court. The following criteria have been specified to judge if evidence is admissible, namely:

- Whether or not the evidence has been collected and analyzed using theory and techniques that have been tested thoroughly.
- Whether or not the techniques have been peer-reviewed.
- The potential rate of error experienced with the use of the chosen techniques.
- Whether or not the theory and techniques are generally accepted by the scientific community.

Although different countries will have different guidelines, the guidelines stipulated by the U.S. Supreme court still serve as an excellent reference framework when it needs to

be determined whether evidence is admissible or not. These guidelines may therefore be used to evaluate the admissibility of digital evidence collected by digital forensics tools.

Only a few investigators have the time and skill to evaluate and analyze their chosen tools to determine whether or not it conforms to the stated criterion [5]. Even if the tools do conform to the criterion and the tools perform perfectly in a trusted environment, it may still give inconsistent results in an untrustworthy environment [5]. This is largely due to the fact that software applications will rarely contain all the operating logic needed to perform basic functionality that can be supplied by external drivers or the operating system; the applications will rather rely on libraries or low-level drivers to perform trivial tasks. Unfortunately these libraries and drivers may be compromised to produce results that are inconsistent with the digital evidence.

III. Commercial forensic tools

Commercial easy-to-use forensic toolkits tend to be extremely expensive while their open-source (or free) counterparts tend to have limited functionality and are very difficult to use [11]. This scenario creates a problem since not every investigation team has the funds to invest in an extremely expensive software package and may have to turn to the available open-source alternative with less functionality which requires investigators with more technical skills.

Another problem with forensic tools at the moment is extensibility. Researchers continually invest their research efforts into finding ways to improve the forensic analysis process. Once a new theory has been developed it needs to be proven. Although various ways exist to prove a theory, it would make sense to ultimately create a prototype to demonstrate the theory in action. At the moment it is an extremely complicated and technically challenging process to actually create a prototype that conforms to a list of criteria needed to ensure that the results extracted by the prototype is admissible in court. This is especially true if the theory is not based on the bit or byte-level but on a higher, more abstract view of a computer system. This is largely due to the fact that the lower-level functionality also needs to be implemented to serve as a basis for the higher-level theoretical process developed through research that needs to be proven. It would be ideal to build on the already-existing functionality supplied by commercial forensic tools as the base functionality has already been tested and proven previously. Unfortunately it would be virtually impossible due to the fact that the source code of these tools is not available to the general public.

Only a handful of researchers possess the technical abilities to actually create a fully-functional forensics tool. Some would attempt to modify already available open-source tools to conform to their requirements; others may try to write a tool using a simulated environment to try to prove their theory. Although these are steps taken in the right direction, a solution is needed to allow researchers to perform rapid prototyping on a tried-and-tested forensics platform in an attempt to speed up digital forensics research efforts which will ultimately lead to more digital forensic science breakthroughs in shorter amounts of time.

IV. Functionality needed in a forensics platform

According to Eckstein [7], what the term “digital forensic analysis” entails depends largely on the source of evidence at hand. As an example, consider the two different digital crime scenarios: the first is a denial-of-service attack executed by an individual located at a remote location; the second is the unauthorized modification of a resource located on a local computer not connected to a network. Analysis of the first scenario will largely consist of scanning through network logs and captured network traffic while analysis of the second scenario will rely on the analysis of a captured hard drive image. Although the forensic analysis in the two different cases focuses on different forensic media types, they are both equally valid evidence sources which may be used to implicate the involved parties. It is therefore crucial that a forensics platform supports the analysis of sequential data (such as a captured network trace) as well as relational data (such as captured disk images).

A. *Digital evidence characteristics*

The characteristics of digital evidence should be taken into account when the possible functionality of a digital platform is defined in an attempt to capture the needed functionality more accurately. According to Wang [17], digital evidence has the following characteristics:

- Digital evidence can be copied easily; unfortunately the copying of digital evidence does not guarantee a consistent copy of the original evidence in question.
- Digital evidence is difficult to authenticate.
- Digital information is not well-perceived by the human senses. Humans will therefore experience difficulty understanding the captured digital evidence in question.

A forensics platform should accommodate the stated characteristics by supplying functionality that attempts to solve the issues experienced with digital evidence. A simple solution to the first two problems may be to supply the forensics platform with secure hashing algorithms that can be used to prove the authenticity of captured evidence. The last problem holds a bigger challenge: how to structure captured data in such a way that relevant information is more visible to an investigator than data that may not be helpful in solving a case. This characteristic is extremely important, as it is crucial for investigators to create an abstract view of the digital evidence in question [16]; without such an abstract view, much more time would be needed to analyze and interpret which ultimately increases the cost of the investigation process.

B. *Evidence timelines*

Another aspect that should be taken into account when examining evidence is the relationship that exists between the collected evidence and the time of collection. A clear distinction should be made between the various stages or timeframes that surround a digital investigation to create a clearer forensic vision of key aspects involved with a digital crime under investigation. These aspects may include: the possible suspects, the digital events as well as well as the connections between the suspects and the digital events that lead to the perpetration of the crime. Evidence collected at various stages will be related differently to these aspects. As

an example, consider evidence taken before the actual perpetration of a crime and evidence taken after the event in question occurred. Surely the type of information that will be extracted from the two different evidence sources will be different, each with different forensic intentions.

Evidence taken before the event took place will describe the functioning (or malfunctioning) system, its users and its environment. According to Pfleeger and Pfleeger [13], an attacker must have three things to be able to perform a malicious attack namely: method, opportunity and motive. Although the type of evidence taken before a crime has been committed may show indications that a computer crime will be committed in the very near future, it shows no concrete evidence of a crime that has been committed at the moment of capture. However, it may be used to indicate motive, opportunity or method which is needed to implicate possible suspects once a crime has finally been committed.
Evidence taken after the event in question will largely show the system's state after the event took place. By examining the captured system state, investigators should be able to deduce which individuals were responsible for committing the act in question.
The example illustrates that it is extremely important to make a distinction between evidence captured at different stages in an investigation. Three different stages have been identified to illustrate the distinction between the information conveyed by the captured data in different stages of the investigation. These stages are:

- Pre-incident stage
- Incident stage
- Post-incident phase

It should be noted that the first two stages is characterized by the capturing and analysis of live data while the last phase is characterized by the analysis of static data captured at a crime scene.

Fig. 1. Various incident stages.

The pre-incident stage focuses primarily on forensic readiness. Forensic readiness describes the extent to which a system is able to supply forensically-sound information to aid the digital investigation process [12]. Special software and hardware can be installed to monitor user actions and minimize the likelihood that the users of these systems can participate in mischievous activities without being noticed through policy management and the enforcement of restrictions. Suspicious activities may be captured and logged as required.
The incident stage is concerned with the capture of digital evidence while a crime is being committed. The incident stage is primarily responsible for the capture and archiving of events as they occur in real time. The primary goal of the incident-stage it to implicate involved parties by capturing identity-revealing information as the digital crime is being committed. This stage is likely to be associated with the capture of network traffic while a crime is being committed. According to Corey et al. [6], instead of capturing a subset of live data, it is better to capture all the available data and analyze a subset of the data at a later point in time. This is done to prevent potentially crucial pieces of evidence from being "tossed away" during the capture process which may cause investigators to reach false conclusions during later stages of the investigation.
The last stage is the post-incident stage in which the entire suspect and/or victim system's state is captured and analyzed after the digital crime has been committed. The phase is characterized by the mass-archiving of the states of the systems involved in the digital crime in an attempt to determine how the systems were used and by whom.
A digital platform should be able to make a distinction between the stages discussed previously to facilitate the possible automation of the identifying of links between evidence captured at the various stages of investigation that would ultimately help investigators to pinpoint crucial evidence located in masses of digital evidence data.

From a more technical point of view, a forensics platform should support formats commonly associated with digital forensics. These formats may include (but are not limited to) FAT or NTFS for file system formats, TCPDump format (the de-facto standard for captured network traffic [6]) and other formats that would be considered common. By supplying a platform that supports these file formats by default, the task of the researcher trying to develop a new and revolutionary forensics prototype would be simplified greatly, as he/she only needs to focus on the research question at hand, and not on the small details surrounding the research.

V. A PROPOSED PLATFORM ARCHITECTURE

According to Casey [4], there is a difference between the examination and the analysis phases of forensic evidence. The examination phase is concerned with the extraction of digital evidence from the scene of a crime while the analysis phase is focussed on finding relationships between the evidence, the events that took place as part of the crime and the involved parties. From a technical point of view the platform should physically support the examination phase and supply the low-level functionality needed to perform the analysis phase with ease. It would therefore be a good idea to develop a platform that supports the acquisition, storage and analysis of digital evidence.
It may often be the case that a few investigators will need to work together on the same case; investigators may need to add or analyze evidence simultaneously which means that simultaneous access is required to the facilities needed to store retrieve evidence. A conventional single-user system design would therefore not be sufficient; instead it was decided to use an architecture that allows multiple investigators to capture, store and analyze forensic evidence simultaneously while allowing researchers to quickly and easily extend the functionality of the overall system. A layered client/server architectural model was chosen to adhere to these requirements.

A. The layered model

A layered model was chosen to allow researchers to easily build their research prototypes on top of already-existing lower-level processing layers. Not only will this layered approach save researchers valuable time when

implementing a prototype, but it will also help to decrease the amount of errors introduced as a consequence of programming mistakes. This is due to the fact that lower-level layers are likely to have been tried and tested by many whereas freshly written code by a researcher may not have been analyzed for errors as vigorously as its layered counterparts.
Five different layers have been identified, namely:

- Physical
- Interpretation
- Abstraction
- Access
- Logging

The following diagram visually depicts the layers and the relationship that exists among them:

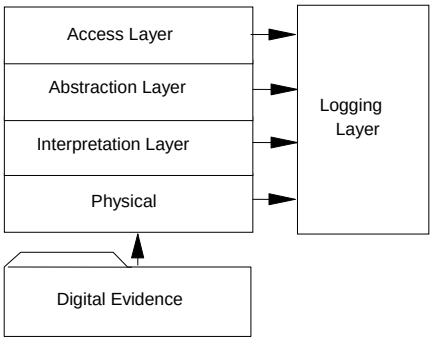

Fig. 2. The layered architecture.

1) The physical layer

Digital evidence in a popular forensic format (such as a disk image or TCPDump trace) is supplied to the physical layer. Because digital evidence are likely to be supplied in byte-by-byte copies of physical devices (RAM, ROM, hard drive) or the state of a physical communication devices such as an Ethernet card at a particular point in time, it would be useful to develop a software emulation layer that emulates the original physical device from which the evidence was captured. The advantage of this approach is that it may be possible to adapt already existing device driver implementations to use the supplied software emulation layer which may once again save implementation time. This is due to the fact that custom driver does not have to be developed scratch as already existing tried-and-tested driver code may be modified and used.

2) The interpretation layer

The interpretation layer will typically consists of the software performing the same task as traditional device-drivers on a conventional system. The purpose of the interpretation layer is to read the block or stream-level data supplied by the physical layer and convert it into a file or entry-level information which is commonly accepted by and understood by programs as well as individuals.

To prove the physical layer/interpretation layer concept, a prototype was developed by the author that made use of the FreeDOS32 [8] Fat12/16/32 file system driver to supply FAT support on the Interpretation-Layer level. The FreeDOS FAT driver was modified with very little effort to make use of the hardware emulation supplied by the physical layer. The configuration was tested by mounting a Fat12 image using the APIs supplied on the physical layer. The data could then be accessed by using the supplied APIs found on the interpretation layer. The results were excellent: with minimal effort it was possible to create an application that read from files stored in a FAT image. The experiment proved that it was possible to use open-source device drivers on the lower-level forensic layers with great success. This means that it would actually be possible to extract various types of device drivers from open-source projects and incorporate it into the framework to supply support for a wealth of different device formats.

3) The abstraction layer

The purpose of the abstraction layer is to supply functionality that is not specific to any operating system or computing platform in an attempt to hide unnecessary details that may obscure an investigator's perception of the information depicted by the digital evidence. Another purpose of the abstraction layer is to facilitate investigators in identifying relationships that may exist among different pieces of digital evidence. As Tallard and Levitt [16] informs us, this functionality may be crucial to help filter out data that is not relevant to help to create abstract objects that can be interpreted in a relational manner to other objects to save valuable investigation time.

4) The access layer

The purpose of the access layer is to supply investigators with access to the information interpreted by lower-level system layers. Search and indexing as well as access-control functionality is expected to be implemented on this layer. Visual abstraction may also be present on this layer in an attempt to display captured digital information in a way which is better perceived by humans. As discussed by Wang [17], digital evidence is not well-perceived by the human senses, and this functionality is therefore needed to help investigators understand the colleted evidence in question in a shorter timeframe.

5) The logging layer

According to the NIST reliable disk backup criteria [10], all tools that take part in the backup of disk data should log all errors that may occur as well as offer resolution to those errors. Logging is an extremely important part of forensic analysis and should not only include a list of system errors that occurred, but it should also contain a list of steps that the investigators executed to show how they got to their case results. These logs may later be used in court to prove or disprove that the investigators conducted the investigation in a manner that is forensically sound. Every layer should log events as they happen on that specific layer. In doing so the log entries could be used to determine what each of the different objects on different layers are busy with simply by looking at the information logged by the layers located underneath it. This functionality may be crucial in proving or disproving that an error exists in commercial closed-source software built on top of one of the higher-level layers by simply inspecting the log entries generated by the objects located in the layers located beneath it.

B. The client/server model

The Client/server architecture has been chosen to manage the storage requirements as well as some of the processing requirements that may be needed by the forensics platform. This decision will allow investigators in a possible distributed environment to collaborate and contribute to the same case simultaneously. The following diagram depicts the client/server the will be used by the framework:

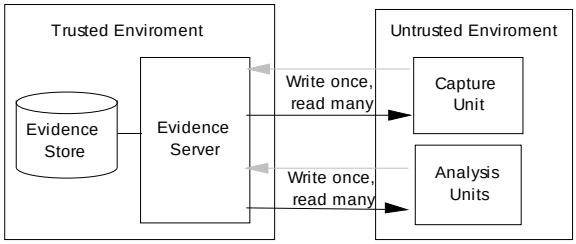

Fig. 3. The client/server system components and their operating environments.

The proposed architecture consists out of three distinct units namely:

- Evidence server
- Capture unit
- Analysis unit

These components will perform their duties in two different environments, namely a trusted and untrustworthy environment. It is assumed that the evidence server will always be operating in a trusted environment while the capture and analysis units may or may not operate in a trusted environment.

1) The evidence server

The evidence server will be responsible for supplying basic logging and storage facilities to capture and analysis units. It is important to note that data stored by the evidence server should be immutable, in other words not a single file ever sent to server to be stored may either be modified or deleted; instead a version control mechanism may be used to create a new version of the file in question every time that a change has been requested.

2) The capture unit

The purpose of the capture unit is to perform digital evidence acquisition of a crime scene. Although the captured evidence is stored on an evidence server, the capture unit should still have the capability to store evidence locally or on some form of storage medium as it may not always be possible (or a good idea) to connect to a remote evidence server to upload captured evidence from a crime scene.

According to Carrier [2], there exist two types of evidence acquisition methods, namely live and dead acquisition. In a live acquisition evidence is taken from a system while it is online, while the opposite holds for dead acquisition (also known as a snatch-and-grab approach [1]): the machine in question is taken offline first before an acquisition is performed. Evidence collected through live acquisitions, such as a list of open files or ports on a system may be very descriptive but unfortunately data captured through live acquisitions are known to have a lesser degree of trust associated with it, since the acquisition is performed in an untrustworthy environment [2] which may contain filter software such as root kits.

Although evidence collected through dead acquisition may be more forensically sound, there may be some instances where a live acquisition is more preferable. The capture unit should therefore be able to perform live as well as dead evidence acquisitions.

3) The analysis unit

The analysis unit will be used by investigators to study captured digital evidence in an attempt to uncover information that may implicate possible suspects. The analysis unit may or may not operate in a trusted environment which means that sufficient measures needs to be in place to ensure that the results obtained by the analysis units are not manipulated in any way.

C. Layer distribution

The distribution of layers may depend entirely on the scenario in which the platform needs to be used. One extreme may be to place the physical, interpretation and abstraction layer on the evidence server leaving the analysis and capture units with the access layers. This arrangement would in effect lead to a situation where thin-clients connect to a server that does all the processing. The biggest problem with this arrangement is cost in terms of network communication time, processing time and storage. The extreme opposite in which fat-clients only use the server as a storage medium may also not be ideal in all situations. Unfortunately there is no easy answer to the best way to distribute the layers between the client/server components. Fortunately the forensics platform will be designed in such a way to accommodate various configurations which would allow investigators to configure their systems according to their specific needs.

VI. PROTOTYPING

An open-source project called the Reco Platform [14] has been established to develop a forensic platform which conforms to the architecture specified in this document. An alpha version of the platform is available for download and can be used to develop prototypes. At the moment of writing, the physical, interpretation and abstraction layers are supported by the Reco Platform. The platform currently supports FAT as well as the EXT range of file systems and has the capability to extract meta information from well-known media types (which includes, but is not limited to, mp3, ogg, asf and avi files).

To prove that the platform minimizes the amount of time and skills needed to develop a forensics prototype, it has been decided to develop two applications that make use of the Reco Platform. The purpose of the first application was to extract meta-information from files located on a disk image. The purpose of the second application was to display the meta-information extracted with the first application. Amazingly, the first application consisted of a total of 40 lines of C++ code, while the second application consisted of 501 lines of C++ code. The amount of programming that was done was kept to a minimum due to the rich forensics library supplied by the platform. These two applications have shown that the Reco Platform can be useful in situation

where forensic prototypes needs to be developed as the platform enables developers to create prototypes with relatively little effort.

VII. Future work

The architecture of the planned forensic framework has been discussed in some detail. More prototyping as well as actual design and development of the various layers will take place in the near future. A definite need also exists for a secure logging mechanism that will allow external evaluators to actually determine if a closed-source software component is functioning correctly or compromising the integrity of captured data by simply reviewing the log file generated by components above or below it in the layering model.

VIII. Conclusion

This paper emphasized the great need currently experienced by the academic as well as the forensic investigation community for an open forensic-investigations platform on which forensic research prototypes can be built and tested. With the help of such a platform, the development time of forensic research prototypes could dramatically decrease leading to an increase in digital forensic investigation breakthroughs in a smaller amount of time.

The architecture has been defined based on industry needs defined in literature. The architecture consists of a layered client/server model. The layered approach will benefit researchers by supplying them with common forensic functionality that is needed so that they can take the focus off the technical aspects regarding forensics and focus on what is really important – their research questions.

The client/server model will benefit the forensic community by allowing investigators to perform investigations in a distributed team environment. Because collected evidence data is stored in a trusted, central location, various investigators will be able to simultaneously access and contribute to a single case leading to better investigation efficiency.

Overall the forensic framework address a need experienced in the community and development should therefore continue to ensure a better and safer digital environment for all.

References

[1] Adelstein, N. Live Forensics: Diagnosing your system without killing it first. 2006. Commun. ACM, 49, 2. ACM, 63-66.

[2] Carrier, B. D. Risks of live digital forensic analysis. 2006. Commun. ACM. ACM, 56-61.

[3] Casey, E. Investigating sophisticated security breaches. 2006. Commun. ACM 49, 2. ACM, 48-55.

[4] Casey, E. Network traffic as a source of evidence: Tool strengths, weaknesses, and future needs. 2004. Digital Investigations, 1. Elsevier, 28-43.

[5] Casey, E. Stanley, A. Tool review – remote forensic preservation and examination tools. 2004. Digital Investigations, 1. Elsevier, 284-297.

[6] Corey, V. Peterman, C. Shearing, S. Greenberg, M. S., Van Bokkelen, J. Network forensics analysis. 2002. Internet Computing 6, 6. IEEE, 60-66.

[7] Eckstein, K. Forensics for advanced UNIX file systems. 2004. Proceedings of the 2004 IEEE workshop on information assurance. IEEE, 377-385.

[8] FreeDOS32. Online: http://freedos-32.sourceforge.net as on 12 April 2007.

[9] Kerr, F. C. Media analysis based on Microsoft NTFS file ownership. 2006. Forensic Science International, 162. Elsevier, 44-48.

[10] Lyle, J. R. NIST CFTT: Testing disk imaging tools. 2003. International Journal of Digital Evidence 1, 4, Winter 2003.

[11] Manson, D. Carlin, A. Ramos, S. Gyger, A. Kaufman, M. Treichelt, J. Is the open way a better way? Digital forensics using open source tools. 2007. Proceedings of the 40th Annual Hawaii international conference on system sciences. IEEE, 266b.

[12] Mohay, G. Technical Challenges and Directions for Digital Forensics. 2005. Proceedings of the First International Workshop on Systematic Approaches to Digital Forensic Engineering. IEEE, 155-161.

[13] Pfleeger, C. P, Pfleeger, S. L. Security in computing, third edition. 2003. Prentice Hall, 209.

[14] Reco Platform homepage. Online: http://sourceforge.net/projects/reco as on 21 June 2007.

[15] Rogers, M. K. Seigfried, K. The future of computer forensics: a needs analysis survey. 2004. Computers & Security. Elsevier, 12-16.

[16] Tallard, T. Levitt, K. Automated Analysis for Digital Forensic Science: Semantic Integrity Checking. 2003. Proceedings of the 19th Annual Computer Security Applications Conference. IEEE, 160-167.

[17] Wang, S. Measures of retaining digital evidence to prosecute computer-based cyber-crimes. 2007. Computer standards & interfaces, 29. Elsevier, 216-223.

[18] Walker, C. Computer forensics: bringing the evidence to court. Online: http://www.infosecwriters.com/text_resources/pdf/Computer_Forensics_to_Court.pdf as on 12 April 2007.

[19] Wilsdon, T. Slay, J. Digital Forensics: Exploring validation, verification & certification. 2005. Proceedings of the first international workshop on systematic approaches to digital forensic engineering. IEEE, 48-55.

Renico Koen is a SAP application developer at EULabs, South-Africa. He is also a member of Information and Computer Security Architecture Research (ICSA) Group at the University of Pretoria. His research interests include software design and architecture as well as the utilization of open-source software in development of large projects.

Martin Olivier is a professor at the Department of Computer Science in the School of Information Technology at the University of Pretoria. In addition to normal teaching and research duties, he is the research coordinator of the School of Information Technology. His current research interests include privacy and digital forensics as well as database, application and system security.