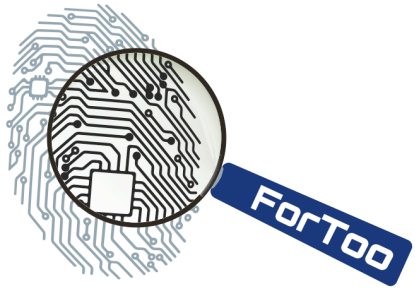
European Commission
Directorate-General Home Affairs
Prevention of and Fight against Crime Programme



HOME/2010/ISEC/AG/INT/002

ForToo – Forensic Tools against Illegal Use of the Internet

D1: Design Specification of the |Digital Forensics Toolset

Work package:	WP1: Design
Contractual delivery date:	31 Oct 2011
First publication date:	**29 Jan 2012**
Actual delivery date:	29 June 2012
Leading partner:	University of Bristol
Contributing partners:	UGCS, FORTH
Editor:	Georgios Oikonomou
Contributors:	Konstantinos Xynos, Aris Tzermias, Dana Polatin-Reuben, Panagiotis Andriotis
Internal Reviewer(s):	Theo Tryfonas
Version:	1.0

Executive Summary:

This document describes the overall design of the ForToo toolkit. The toolkit is made up of a set of forensic tools, which can assist with mobile and social network forensic investigations. The DEViSE architecture and visualisation tools are used to integrate the individual forensic tools into a single architecture. In this document, we present the detailed design of each individual forensic tool as well as of the visualisation and integration architecture.

With the support of the Prevention of and Fight against Crime Programme
European Commission - Directorate-General Home Affairs

This project has been funded with the support of the *Prevention of and Fight against Crime* Programme of the European Commission - Directorate-General Home Affairs. This publication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Table of Contents

REVISIONS AND QUALITY CONTROL

Ver.	Date	Lead contributor	Action summary
0.1	13 April 2012	Georgios Oikonomou	First Draft
0.2	16 April 2012	Konstantinos Xynos, Aris Tzermias	Social Forensics Tool Design, Visualisation Tool design
0.3	23 May 2013	Konstantinos Xynos	DEViSE Architecture
0.4	25 May 2013	Georgios Oikonomou	Added Steganalysis tools
0.5	19 June 2013	Panagiotis Andriotis	Added Android Forensics
1.0	29 June 2013	Theo Tryfonas	Final review completed

1 Introduction

This document describes the overall design of the ForToo toolkit. The toolkit is made up of a set of forensic tools, which can assist with mobile and social network forensic investigations. The DEViSE architecture and visualisation tools are used to integrate the individual forensic tools into a single architecture.

This document starts with an overview of some related technologies which provided useful during the toolkit's design phase. Subsequently, in Section 3, we present the toolkit's detailed design, broken down to individual sections for each of its components. Appendices have been used for code snippets and examples.

2 Forensic Tool Classification

2.1 Proactive Components

2.1.1 Proactive Forensics?

There has to be a way to drastically improve the efficiency of the investigative effort, and the notion of Proactive Forensics is suggested to be one of them. As opposed to the historically reactionary endeavour that digital forensics has always played, which focuses on the post-incident side of investigation; Proactive Forensics as its name suggests, adopts automation to make the forensic evidence gathering process proactive (Shield, 2010), allowing the digital devices (computer) to adaptively focus resources on identifying and collecting possible traces to potential transgressors (Bradford *et al*, 2004), in advance of an incident alert or evidence request (Grobler *et al*, 2010).

Proactive Forensics is a relatively new branch of the Digital Forensics tree, and is still in its sprouting stage. The term was first sighted during the literature review in the proceeding of an International Conference on Information Technology by Bradford *et al.* in 2004. It is the same as traditional digital forensics in the sense that it still involves the analytical and investigative techniques used for the identification, extraction, preservation, documentation, analysis and interpretation of digital evidence. As the literature will show, the topic of Proactive Forensics is presented and an attempt to define, as well as interpret the scope of it. Even though Proactive Forensics is still a young branch of Digital Forensics, the academic researchers have already had a reasonably different perspective on the focus and spirit of Proactive Forensics.

2.1.2 Definition of Proactive Forensics

For instance, Daniel (2007) in his PhD thesis defined Proactive Forensics as "the prediction of attacks and changing the evidence collection behaviour before an attack takes place", while Bradford *et al.* (2004) stressed that it is not much about predicting behaviour, but adaptively focusing resources on potential transgressors, by "designing, constructing, and configuring systems to make them most amendable to digital forensics analysis in the future". This is similar to McQueen's (2009) understanding that it is "an idea of anticipating a hacker/exploits path and detecting them in the process, which allows the instant collection of proper evidence that are valuable for judicial review, as well as threatening the hackers, to enhance security". Shields (2010) also further emphasised the idea of installing software on target system ahead of any incident, so that information can be preserved proactively and stored until needed for examination.

Although these researchers above have a different perception of whether Proactive Forensics is about predicting attacks, there appears to be a mutual recognition that the effort and resources, such as computer systems in an organization and the software installed on top of it, are to be focused on the pre-emptive evidence (track) collection as well as preservation, to assist in digital forensics analysis in the future,

before an incidents can occur. Grobler *et al.* (2010) gave a more comprehensive definition covering all of the points above:

"Proactive Digital Forensics ensures that sufficient processes, procedures, technologies and comprehensive digital evidence are in place to enable a cost effective, successful investigation, with minimal disruption of business activities, prevention of anti-forensic activities, and the use of digital forensics technology to enhance security posture of the organization and demonstrate good governance."

Combining the essence of the definitions presented above, for the purpose of this project, the authors have defined Proactive Forensics as –

"*a forensic approach that focuses effort and resource to facilitate dynamic evidence collection behaviour upon detection of activities prior to an incident, storing the evidence proactively to make them amendable for future analysis and judicial review.*"

This highlights three main special features of Proactive Forensics: dynamic evidence collection, storage of relevant information, and judicial review standard evidence.

2.1.3 Proactive Network Forensics

Proactive network forensics involves tools and methods for identifying intrusions by constantly monitoring network traffic for the presence of attacks and anomalies. Network traffic is volatile; it is transmitted from sender to receiver and then lost. Attackers may destroy evidence on victim's host to hinder detection such as deletion of log files. Therefore, monitored traces from network traffic can become valuable to identify potential attacks.

Proactive network forensics is strictly tied with intrusion detection systems and monitoring of network traffic in general.

An Intrusion Detection System (IDS) is a specialized hardware or software equipment that constantly monitors network or system resources for existence of malicious activities. When an intrusion is detected, the IDS alarms the user (network administrator) and logs details of the attack. Some systems make a step further and try to prevent the attack from happening. Such systems are called intrusion prevention systems (IPS).

IDSes are segregated into two main categories; Network-based Intrusion Detection Systems (NIDS), and Host-based Intrusion Detection Systems (HIPS). The former category detects attacks "on the wire" by examining network traffic from multiple hosts. Usually, NIDS are located on crucial points of a network, like gateways. The latter category identifies intrusions on a specific host, by analyzing system calls, inspecting modifications on the file system and monitoring various activities.

Furthermore, IDSes can be classified by the detection technique they use, as well. Signature-based IDSes, compare monitored traffic using pre-configured attack patterns, called signatures. On the other hand, anomaly detection IDSes, classify activity as normal or anomalous according to specified heuristics (e.g. excessive use of memory, abnormal network activity etc.) The state-of-the-art IDSes are Snort (Snort) and Bro (Bro)

As NIDS are deployed on exit nodes of a network (e.g gateways) they must able to detect attacks without degrading network speed. The ever-growing speed of networks

and the complexity of attacks impose a challenge to NIDS implementations. Intrusion detection systems rely on pattern matching which is a computationally intense operation. Highly parallel architectures, such as modern general-purpose GPUs are a compelling alternative for offloading computationally-intensive operations from the CPU. The release of software development kits from GPU vendors such as NVIDIA and ATI, pushed developers towards this end.

One of the first approaches was Gnort (Vasiliadis *et al*, 2008), which utilizes the GPU to offload pattern matching execution. In particular the Aho-Corasick algorithm, which is used by Snort IDS, was ported to run on GPU. Evaluation showed that Gnort achieved a maximum throughput of 2.3 Gbit/s on synthesized traffic, and outperformed conventional Snort by a factor of two in real network traffic. Furthermore, Vasiliadis *et al* (2009, 2011), implemented libraries running on the GPU, for pattern matching and string searching operations. Results showed that the GPU-based pattern matching engine can achieve content scanning in multi-gigabit rates. In particular, GPU algorithms are 30 times faster than a single CPU core, reaching a maximum throughput of about 30 Gbit/s.

Apart from GPUs, other hardware components can used for parallelization. MIDeA (Vasiliadis *et al*, 2011), presents a multi-parallel intrusion detection architecture capable of analyzing attacks in high speed networks. MIDeA parallelizes the processing of network packets using a three-layer parallelization: at the NIC, at the CPU and at the GPU. A multi-queue NIC is used to distribute network traffic to a set of multi-core CPUs for packet processing and analysis. For better performance, content inspection operations are offloaded to a set of GPUs. Experimental evaluation results showed that the three-layered parallelization used in MIDeA can achieve processing speeds of up to 5 Gbit/s with zero packet loss.

Network-based Intrusion Detection systems are suscetible to overload attacks, where the system may consume all the available resources and drop most of the monitored traffic till system's overload recovery. Attackers may leverage overloads to evade detection. Papadogiannakis *et al*, (2011) proposed Selective Packet Paging, a two-level memory management approach that detects packets which slowdown the system and buffer them in secondary storage for later processing. Moreover, it uses randomization to avoid predictable evasion from sophisticated attacks. Selective Packet Paging (SPP) was implemented within Libpcap, and evaluated using Snort IDS. Results showed that Snort is able to detect all the attacks under overload, tolerating both algorithmic complexity attacks and traffic bursts. Another approach to improve the processing time of passive monitoring applications, including NIDS, is proposed in (Papadogiannakis *et al*, 2007). The authors present a technique called locality buffering, where captured packets are reordered before they are delivered to the monitoring application to improve the memory access locality and cache usage. Reordering is achieved by clustering packets with the same source and destination port numbers together. This results in improved code and data locality, and thus increased packet processing throughput. Locality buffering has been implemented within the libpcap packet capturing library, allowing applications to use it transparently. Experimental evaluation showed that Snort IDS, exhibits a 21% increase in its packet processing throughput, and it can handle 67% higher traffic rates with no packet loss.

To improve efficiency of current NIDSes, Antonatos *et al.* (2005) described an algorithm for pattern matching tailored for intrusion detection solutions, called

Piranha. It is based on the observation that if the rarest substring of a pattern does not appear, then the whole pattern will not match. Deployment to Snort and experimental evaluation indicated an increase of 11% to 28% of processing time compared to existing NIDS pattern matching algorithms.

2.1.4 Value of Proactive Forensics

Evidence Collection Automation is not the sole meaning of Proactive Forensics, the value of Proactive Forensics doesn't stop here. Having gained a general understanding of Proactive Forensics, there is another problem that the authors have recognized Proactive Forensics as a conceptually-possible response to.

Since the enactment of the British legislation, Computer Misuse Act in 1990, there has been less than 20 prosecutions charged under it every year. The evaluation of the reason behind this from the computer law class suggested that, this small number has certainly nothing to do with the actual number of unauthorized access or impairment offences, which can potentially be charged. Rather, it is believed that in terms of internal offences within the corporate sectors, the adverse publicity prospect resulting from a prosecution is often at times far more significant than the benefits of bringing a case (Fafinski, 2007). This may include things like loss of consumer confidence, and the fees and effort needed in order to bring up a case, which could not be recovered even when a case is won. The main idea here is this: when the law cannot effectively protect those who need it, what other ways could the potential victims seek to counter these illegitimate users in an organization? When a malicious user can hold an organization's reputation at hostage to not have him prosecuted as a legal offender, what could the organization have initially done to dismiss him before he embarrasses the organization?

When it is understood that while system security is to block malicious users from committing the breach, and that forensics seeks to collect evidence to prosecute the offender; the proactive approach of forensics could be an effective solution to threaten these malicious users. Investigation and evidence collection being part of litigation usually commences after a case is brought up; Proactive Forensics in turn makes an attempt to track and collect an offender's footsteps as he progresses through his bad intents, before the actual crime happens. For instance, the traditional security or forensics approaches could result in two situations: a security measure such as IPS could block a user's malicious trial and error nine times, but if one try out of 10 successfully penetrates the target system, the organization is in trouble. Else, even though there are logging mechanisms on the system, the data they capture could be overwhelming. Kornexl *et al.* (2005) pointed out in their article that bulk of the traffic in high-volume streams comes from just a few connections; there has to be a solution to filter these data to keep only those that are necessary. Otherwise, when an investigation is taking place, investigators are overwhelmed by a sheer amount of logs and files they have to go through, using much of the precious investigation time, and increasing storage cost at the same time. As Proactive Forensics is detection-oriented and dynamic, it makes sure that only relevant evidence is collected, either from its own algorithm or from the existing log systems, and isolated in a case-by-case repository to assist in investigation efficiency. When this collection of evidence is presented to the offender, there won't be a dilemma of whether to spend the money and the effort to take chance in prosecuting the offender, because both of the parties will know, the offender doesn't stand a chance of winning. The organization could

dismiss the offender with an undeniable excuse, and prevent the possible upcoming problems once for all.

"We have a complete irrefutable evidence collection of what you have done, what now do you have to say?" This is the true value of Proactive Forensics; a defence pre-investigation in nature, with prosecuting power.

2.1.5 Related Disciplines of Confusion

While the science of forensics has always been reactive and post-incidental, it may appear contradictory to foresee forensics being proactive, in the sense that it might in turn be a duplicate approach to other disciplines. This section sets out a list of these disciplines that are closely related, however not considered equivalent, to Proactive Forensics, which often causes controversy to whether Proactive Forensics is really that relevant as a new approach to perform the same methods that already exist. This clarification also draws out the clear feature of Proactive Forensics.

Information Security

Information Security is similar to Proactive Forensics in the sense they are both *preventive* in nature. In information security, measures are put in place to protect information systems and the information they contain from loss of confidentiality (secrecy), integrity (authenticity), and availability, often as a result of unintentional negligence in operation, or intentional intrusion (Williams, 2008). Proactive Forensics in turn focuses on collecting appropriate data to provide good investigation leads to prevent a computer incident from happening that could well lead to a crime (Bradford *et al.*, 2004).

Forensic Readiness (FR)

Forensics Readiness (FR) is defined as the ability of an organization to maximise its potential, in terms of preparing its system, physical and procedural security of data, and staff security-awareness, to gather data of evidential standards for admissibility, and at the same time minimizing the cost of investigation (Rowlingson, 2004). Indeed, these are the partial main goals that Proactive Forensics is looking to achieve. In fact, the term FR is often taken synonymously with Proactive Forensics (Mouhtaropoulos *et al.*, 2011); other instances take FR as one way of Proactive Forensics (Garcia, 2005). However, Garcia in her presentation drew a distinction between these two disciplines by defining an additional element of automation to Proactive Forensics. While Forensic Readiness sets a system ready for collecting comprehensive evidence, standby-ing for their turn of performance on an investigation stage as an incident occur, automated forensics uses these collected data in a proactive manner with incident-prevention in mind.

Intrusion Detection

Intrusion Detection is a way of security management for computer system including networks, which gathers and analyse information to identify possible breaches by both external (intrusion) and internal (misuse) attacks (Rouse, 2007). Bradford *et al.* (2004) made a distinction between these two by pointing out that intrusion detection generally targets quick detection and understanding of intruders taking succinct action, while Proactive Forensics works over the long term setting alerts and adjusting system parameters as appropriate. In a way Proactive Forensics is more to

do with regulating the proper and legal use of computer systems over time by the internal users, rather than instant detection of break-in by the external users.

Active (Live) Forensics

Active (Live) Forensics is another big branch of digital forensics gaining notice in the recent years as an alteration to the traditional digital forensics. However, its focus is on gathering relevant evidence while minimizing the effect of the incident during an on-going incident (Grobler *et al.*, 2010). As opposed to traditional (dead) digital forensics, live forensics works to achieve retention of volatile data, and countermeasures for encrypted files on a live system, while the incident is taking place (Lessing and Solms, 2008). The difference of Proactive Forensics is clear here in terms of when they are applicable in an incident.

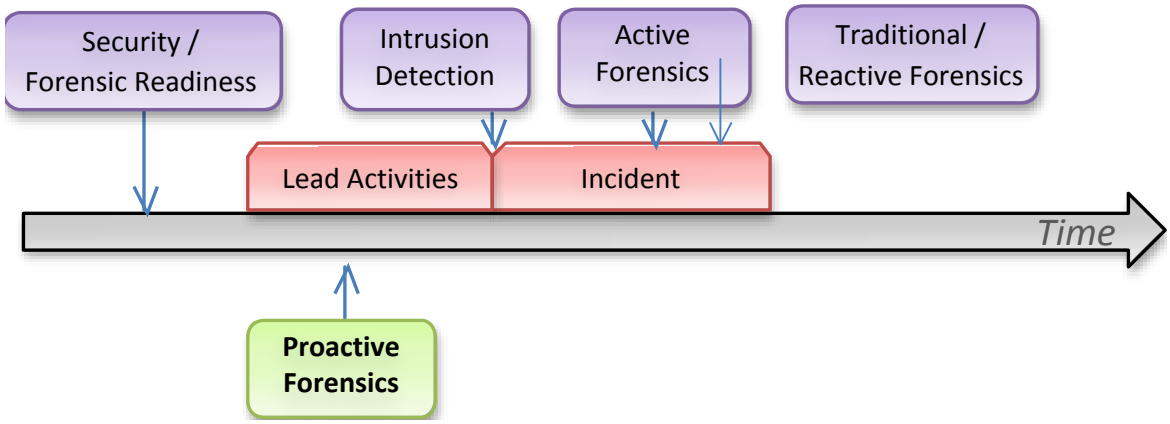


Fig. 1 - Timeline of Application of Different Approaches

The figure shows a timeline which different approaches above come in place in relation to an incident. So far the ideal existence of Proactive Forensics has been discussed through different researchers' view on its focus, the elements that it deals with, as well as how Proactive Forensics fits into the different current technologies.

2.2 Mobile Forensics and the Android Structure

Smartphones and tablets are digital devices with great capabilities and they can be treated as computers despite their difference in the hardware implementation. Thus, smartphones and tablets can be included in the area of Digital and Computer Forensics.

Android is an operating system developed by the Open Handset Alliance. Its architecture has four main levels: Linux Kernel, Libraries and Android Runtime, Application framework and Applications. Lessard and Kessler (2010) state that the Linux system devices default to the first physical hard drive (or /dev/hd0) and are only capable of understanding character and block devices, like keyboards and disk drives. Thus, the use of a Flash Transition layer is needed in order to provide the mobile phone its functionality, due to the fact that flash memory devices are neither character nor block devices. The connection between the Linux kernel and the physical flash device is accomplished using a Memory Technology Device (MTD). Vidas et al. (2011) add that a lot of Android smartphone devices use MTD, an

abstraction layer for raw devices which permits software to use an interface in order to use various flash technologies. In addition, they underline that useful information can also be stored on SD cards. They also illustrate that most Android devices have various partitions that are usually mapped to MTD devices. *User data, cache, boot* and *recovery* are some of the partitions in a typical Android system. The most common file systems an investigator can face during the analysis are the YAFFS2, FAT, EXT4 or other proprietary systems such as Samsung's Robust FAT file system (RFS). Especially with Samsung's RFS, in order to work timely, Linux kernel does not use MTD devices but creates several Sector Translation Layers (STL) and Block Management Layers (BML) block devices (/dev/block/).

Previous work indicates that the most interesting partitions for a forensic investigator will be the user data and system partitions (Vidas et al. 2011). According to Lessard and Kessler (2010), a forensic examination of an Android based mobile device is mainly focused on the Libraries component of the architecture and specifically on the SQLite databases, where the majority of the data is stored. Moreover, in (Hobarth and Mayrhofer, 2011) we find that each application has its default home directory, which is located at /data/data/<package_name>/ and contains four directories: databases, libs, files and shared_prefs. Also, various manufactures may want to alter these directories to their own needs, for example Samsung might use in various devices the convention /dbdata/databases/<package_name>/ for preinstalled applications. Hence, an investigator must know that there are numerous default locations on Android devices for applications to be stored.

2.3 After-the-Fact Components

2.3.1 Network Forensics and NFAT

Network Forensics is an emerging field within Digital Forensics, although there is no official definition there have been several definitions proposed.

The following definition was proposed by DFRW at the 2001 workshop in New York.

"Network Forensics

The use of scientifically proven techniques to collect, fuse, identify, examine, correlate, analyse, and document digital evidence from multiple, actively processing and transmitting digital sources for the purpose of uncovering facts related to the planned intent, or measured success of unauthorized activities meant to disrupt, corrupt, and or compromise system components as well as providing information to assist in response to or recovery from these activities" (Palmer, 2001)

Network Forensics is not a protection product. It is not supposed to replace firewalls and intrusion detection systems. It is a process in which methodologies, tools and human intelligence combine for the purpose of a digital investigation.

Network forensics deals with the data found in a network or in a cloud environment covering both traffic from one host to another and data stored in a cloud environment. Network forensics analyse logs, network dumps and data gathered through firewalls or intrusion detection systems or at network devices like routers. The purpose for this is to find the intruder or the criminal so that they can be prosecuted.

To achieve this, the forensics analyst needs tools, so for the purpose for this project Network Forensics Tools (NFAT) are defined as:

"Forensically sound application that enables the interpretation (and possibly capture) of data contained in low level protocols and facilitates the high level application analysis required of many forensic investigations" (Dr. Huw Read, 2012).

These are tools used to support the capture and analysis of network data during a forensic investigation. Network Forensics Analysis Tool's bridge the low level protocol related activity with the high-level application analysis required of many forensics investigations. While it can be argued that these tools have a number of similarities with network security and network analysis tools, they have a number of additional key requirements.

NFATs synergize with IDSs and Firewalls and make possible the long term preservation of network traffic records for quick analysis (Peterman et al., 2002). The attack traffic can be replayed and attackers' moves can be analysed for malicious intent. NFATs facilitate organization of the captured network traffic packets to be viewed as individual transport layer connections between machines, which enable the user to analyse protocol layers, packet content, retransmitted data, and extract traffic patterns between various machines.

NFATs can also be used as an active component in a forensic framework. Active network forensics makes all past and present network data instantly visible and allows perfect fidelity through replaying past traffic, enabling organizations to detect and understand the full source and scope of any security event so they can protect against further attacks. Active network forensics also enables an organization to validate that the same attack doesn't work again after they have implemented appropriate counter-measures. Combining high-speed data capture, indexed storage, and comprehensive analysis tools, active network forensics is analogous to putting a security camera on a network. Doing so instantly exposes any specific network event, making even the most sophisticated and targeted attacks plainly visible both when they happen, and at any time in the future.

2.4 Steganography Detection

The use of several means of covert communication is appealing among individuals or groups that are interested in securing the content of an exchange concealing the act of their interactions. Steganography is one of the methods which have been introduced in order to hide information and covertly spread hidden data through public channels without causing suspicion. JPEG images constitute a widely used medium of secret communication, partially thanks to the fact that they can be produced by any camera, smartphone or image processing tool and can be easily exchanged between a variety of applications.

Steganography aims to transport a message in a hidden fashion by embedding it in a transport medium called a carrier (Fridrich et al., 2001). The grouping of the carrier with the secret message is known as a stego medium or stego cover. The detection of steganographic algorithms and techniques can be a hard task, even more so if the secret data are encrypted with a stego key. Steganalysis is the process of attacking and breaking steganographic methods, either by simply detecting the presence of a secret message or by extracting and potentially destroying it (Chandramouli et al., 2004).

The success of a steganalytic method can be quantified either by the accuracy of the prediction of a secret message's presence in a stego object or by the extraction of hidden information. Steganalysis methods can be further classified into two broad categories: *targeted* and *blind* (or *universal*). In *targeted* steganalysis the attack is mounted against an already known embedding technique. *Blind* steganalysis methods aim to determine whether an object is carrying a hidden message, without any a-priori knowledge.

When the stego carrier is a image steganalysis is prominently based on two approaches: *visual* and *statistical* attacks (Westfeld and Pfitzmann, 2000; Jolion, 2001). Visual attacks demand long training steps and a significant amount of resources. Statistical attacks are more resource-efficient and as a result, several can be found in the literature (Chandramouli and Subbalakshmi, 2004). These are based on the fact that the images' histograms or high order statistics get modified after the steganographic techniques take place. Modern blind steganalytic schemes engage supervised learning to differentiate between the plain media and stego images and also distinguish the data hiding algorithm used for steganography (Solanki et al., 2007).

Benford's empirical law of anomalous numbers (Benford, 1938) has been successfully used in the past for fraud detection in the accountancy sector. It has also been demonstrated that the law in a generalized form can be employed to perform a series of forensic tasks on images, such as the detection of double compression (Fu et al., 2007). This work was limited to greyscale images however. The generalized Benford's Law has been employed for steganalysis elsewhere (Zaharis et al., 2011), but there it was applied on raw byte values and not from an image analysis perspective.

Steganography detection can be used proactively as well as reactively and for that reason it is presented in this separate section.

2.4.1 Steganalysis of JPEG Images

The term JPEG comes from the consortium that created the standard (Joint Photographic Experts Group). It is one of the most common formats and it is widely used by all the manufacturers of digital consuming products such as digital cameras. It comes from the need to exchange images through different platforms and applications. The main goal of the compression is to discard information which is imperceptible to the human eye while leaving unchanged the aesthetic details of the image. Simultaneously, the compression reduces image data size. A detailed presentation of the procedure followed in order to compress a data stream with the standard can be found in Wallace (1992). Usually the Discrete Cosine Transform (DCT) encoding procedure consists of six basic steps: Conversion of the representation of colours from RGB (Red, Green, Blue) to YC_bC_r, downsampling of the chrominance values (usually by a factor of two), transformation of values to frequencies (using 8x8 pixel blocks), quantization process, zigzag ordering, lossless compression using a variant of Huffman encoding.

In more detail, an image consists of pixels and each pixel usually has three bytes that represent its three basic colour components: Red, Green and Blue. The first step to the encoding procedure is to convert these pixel values from RGB to YC_bC_r which is another colour space that has three components. Y represents the brightness of an image and is called luminance while C_b and C_r represent colours and they are called

chrominance. It is known that the human eye can recognize the difference in the luminance of an image more easily than the chrominance coefficients (Lee et al., 2006). The type II DCT is responsible for the quantization process. DCT is a mathematical transformation (uses cosine functions) that converts the pixel values of 8x8 blocks to blocks of 64 frequency coefficients. These numbers are critical for our method.

A digital image and especially a JPEG image can be a perfect cover medium because it usually has large amounts of space where one can embed information. There are numerous factors that result in a successful embedding procedure such as the embedding technique and the cover image characteristics (McBride et al., 2005). A general assumption is that the image should be busy, meaning that it should lack large areas of similarities. Popular techniques used to hide information in images are the Least Significant Bit (LSB) and the DCT encoding. Embedding techniques focus on the quantized DCT coefficients and they usually embed data by applying LSB encoding in those coefficients that are not equal to zero. In McBride et al. (2005) we can find a list of tools that use the quantized DCT coefficients to embed data in images. They rely on the fact that the procedures which follow the quantization phase are lossless and the hidden information can then be obtained. Indicative algorithms from this category are Jsteg, Outguess, JPHide and F5. Those techniques introduce irregularities in the statistics of the quantized DCT coefficients of a colour image.

Statistical attacks aim to determine whether the examined data comply with specific statistical rules that normal image files would follow. A very popular attack is the Chi-squared test which compares the statistical behaviour of a suspected image with the theoretically expected properties of its carrier (Westfeld and Pfitzmann, 2000). Histogram attacks, which can also be classified as statistical, depict disturbances in the distribution of the frequencies of DCT coefficients of a JPEG image. These figures can reveal the existence of a steganographic attempt. A comprehensive and well informed work on steganalysis trends was published by Chandramouli and Subbalakshmi (2004).

Nowadays, machine learning techniques are common in the field of steganalysis. These techniques are based on image features which get altered during the embedding process and machine learning is the de facto standard procedure that deals with them utilising support vector machines (SVM) and lately, ensemble classifiers (Zong et al., 2012; Kodovsky et al., 2012). The features constitute a model for natural, pure images which can be used against the suspected stego carriers. However, despite their accuracy, these techniques are time consuming, they introduce extensive training steps and their complexity is high.

3 Design Specification

3.1 Architecture Overview

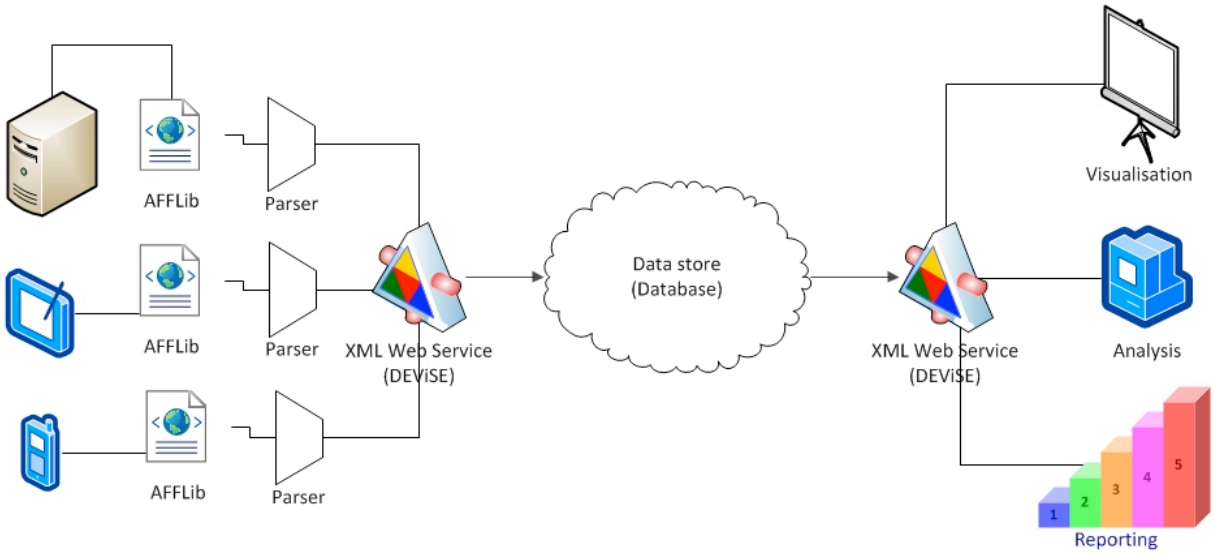


Fig. 2. - High Level System Architecture

3.1.1 DEViSE – Original Relational Database Design

The Glamorgan Cloud is based on earlier research work into relational databases, building a system capable of handling different Computer Network Defence (CND)/ network management data inputs. The new system will support multiple sources of digital forensic data, as further demonstrated in this document.

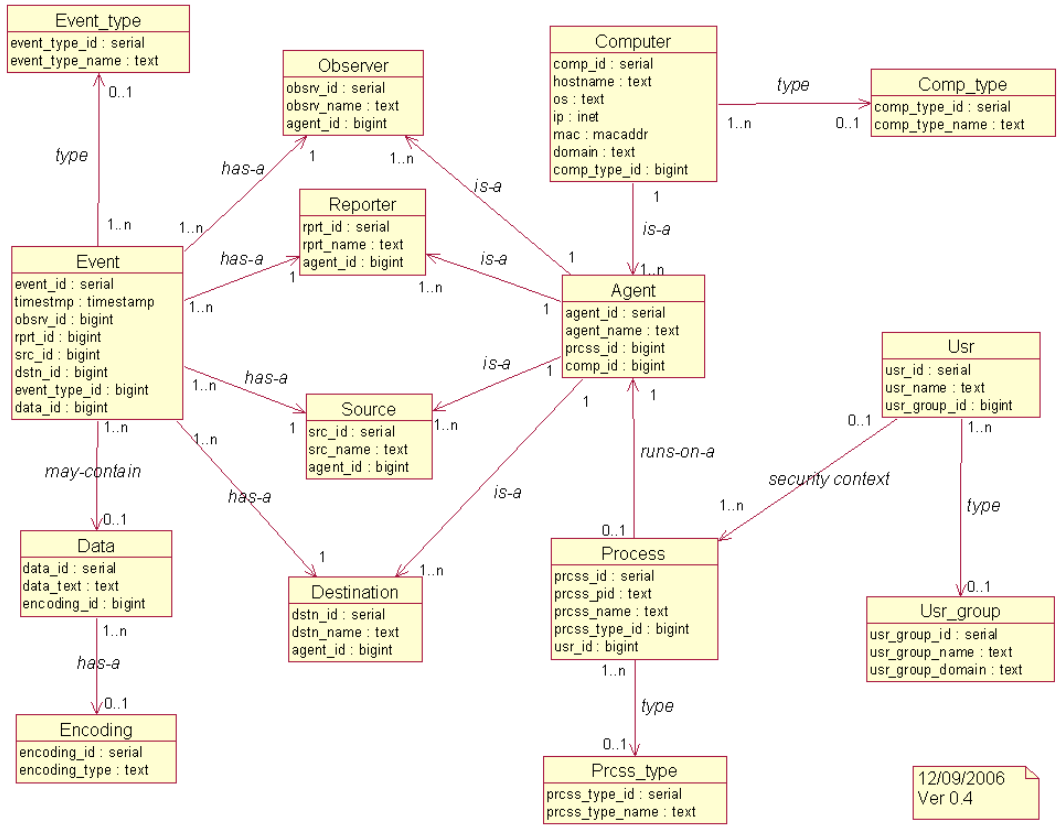


Fig. 3 - Relational Database Design of Core

Fig. 3 demonstrates the relational database that the system was initially built on. This is known as the core system and all other sources of information will expand upon this system. The system was originally built around a relational database, PostgreSQL, and additional work has been carried out to convert the underlying database into a cloud-based format using the Apache Project Cassandra. The existing DEViSE standard has been adhered to for compatibility purposes. Based upon the above schema example INSERT/SELECT functions (in XML) are shown in Appendix 2.A-2.G.

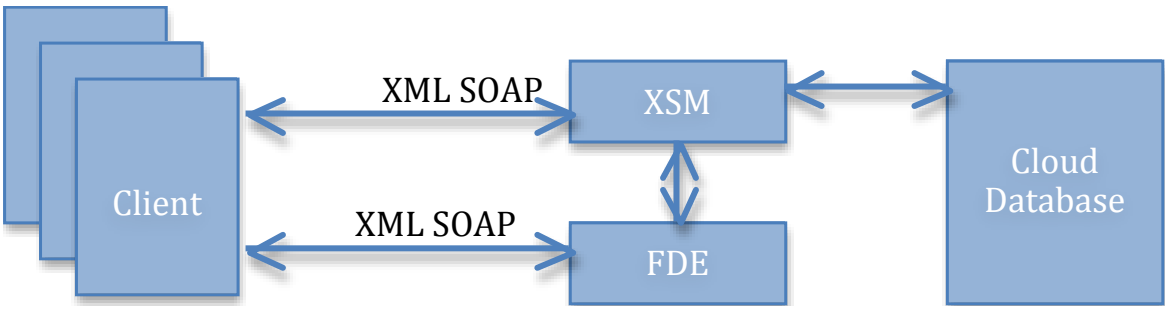


Fig. 4 - Cloud Database Architecture

The XSM XML system, in Fig. 4, is not necessarily tied down to the database design shown in Fig. 3. We have designed a core database which supports expansion based on the event_id. The XML returned by the system has an almost generic format to accommodate any new tables and columns. Within the Cassandra architecture it is possible to change the design without having to update the database schema every time. This is done on-the-fly by the system.

The FDE allows us to associate functional names (key words) to more complex queries (e.g.,joins) therefore eliminating the need to construct the joins every time. The functional names can be expanded to include a new vocabulary as new systems become integrated. Examples of the FDE are shown in Appendix 2.C.6., 2.C.7.

The integration of the application middleware and database tiers is also referred to as DEViSE.

The interfaces to read / write data from / to the Cloud are nearly identical to the original documentation (See Appendices 2.A-2.G). The benefit here is conceptually thinking of the cloud data store as a relational database, but having the benefit of cloud architecture.

To illustrate the interfaces, XML examples are provided in the Appendices as follows:

- Appendix 2.A demonstrates how a Snort log (in syslog format) can be inserted via XML
- Appendix 2.B demonstrates how to obtain the most recently inserted event (by querying the highest or “MAX” event value)
- Appendix 2.C demonstrates how to obtain the *n* most recent events in one query
- Appendix 2.D demonstrates how to obtain events within a specified time-range
- Appendix 2.E demonstrates how to obtain a list of events with data from a specific indexed column
- Appendix 2.F demonstrates how to query based on Event ID
- Appendix 2.G demonstrates how to send a request and obtain specifics from a previously inserted event (e.g. obtaining the source IP address of the event) using the FDE interface

3.1.2 Digital Evidence Storage Formats and Common Exchange Standard

The focus of how to acquire a forensically sound disk image has changed in the last few years due to the fact of the rapid growth in disk size and the diversity for digital evidences. The conventional RAW (dd) disk image takes up too much storage space and it does not contain any other information than the physical bit-by-bit copied image. The ideal format would contain the evidence or multiple instances of evidence from different sources, plus all the information (metadata) you would expect to follow a criminal case (Chain of Custody, type, size, investigator, etc.). The Digital Evidence Bag (DEB) and similar approaches can provide these features. The representation of data, in particular the metadata, differs from each storage format. The storage formats have different vocabulary and datasets for their metadata and these have a direct effect on how the data can be processed and/or exchanged with other parties.

A common storage format for digital evidence has been identified by the Common Digital Evidence Storage Format (CDESF) as a key requirement for sharing and

interoperability of digital evidence. The forensic storage formats discussed in this paper are the most commonly used formats for storing and processing digital evidence as of today.

"The CDESF working folded quite a few years ago… I think the problem was just too difficult to solve and get agreement on, a real shame in my opinion as the industry really needs a common format." (Turner, P., 2011).

The main focus for this group was defining a standardised Common Digital Evidence Storage Format for storing digital evidence and the associated metadata. The work done by the group did not at the time cover a common representation or a common vocabulary for the metadata.

The evidence format that has gained most ground of the formats discussed at CDESF is The Advanced Forensic Format (AFF). AFF has now been included into Access data's software suite FTK, this is one step closer to a common standard. AFF is an open standard so program developers can include it in their software. AFF supports xml metadata and would be a good candidate for merger with one of the exchange standards (Garfinkel et al., 2006).

The representation and exchange of data has been a key issue for the work done by the Department of Justice (DOJ) on their Global Justice XML Data Model (GJXDM) and the later National Information Exchange Model (NIEM)(NewDawn, 2011).

Dr. J. Philip Craiger continued to develop the standard that the DOJ proposed and did significant work in this area with the development of DEML (Digital Markup Language) (NIJ, 2005-2009). DEML is an extension of the GJXDM and later became a part of the NIEM standard. The NIEM standard is now close to a 3.0 Alpha release as the community pushes for a common exchange markup language.

Another approach to include digital evidence into standards like NIEM is FIDEX developed by the National Forensic Science Technology Center (NFSTC), this standard covers more of the conventional forensic science. Whereas, DEML is directed towards digital forensics.

There has been considerable work invested in developing exchange and storage formats for digital evidence and some success in standardising these formats, but there is still a lack of a unified standard for both commercial and open-source tools that is capable of addressing the full range of data captured during an investigation. The capabilities of these formats may have an impact on the investigations results. This paper discusses the problems that arise from the continuing trend in intelligent devices and the limitations of the current work on evidence formats. In particular it addresses the issues of the need to incorporate the demands of communication data, network traffic and cloud storage within these formats. Increasing data is no longer confined to a physical digital device for gathering of evidence, so there is an increasing need to exchange information between conventional forensic tools and network forensic tools.

3.2 Social Network Forensic Tools

In this section we are going to analyze the social forensic tool as well as describe in detail its core components.

Online Social Networks (OSNs) have become an integral part of our social life. An ever-growing amount of people, use them for communication and business purposes. Unfortunately, miscreants could not stay uninvolved. Incidents like cyber-bullying, insults, identity thefts and scams/phishing are currently present on some social networks.

Our goal is to develop a toolset capable of downloading every available information from an individual's social network account (providing that we have account's credentials). Information from different social networks accounts of an individual (Facebook, Twitter) can be combined with local data (e.g Skype chat logs) found on suspect's computer, thus providing forensic analysts with a clearer image of suspect's actions and intentions as well as an interface for querying data. Social network credentials can be obtained by browser's saved passwords file during forensic data analysis.

The social forensic tool has been designed to be modular. Each OSN crawler is a plug-in component that can be attached to the tool, and can communicate with it using a predefined API. Thus social networks crawlers can be easily integrated to the tool. Each plugin must inherit an abstract class and implement a method, called run(). This method is called by the tool for each available plugin to start the crawling process. Architecture of the social forensic tool is depicted on Fig. 5.

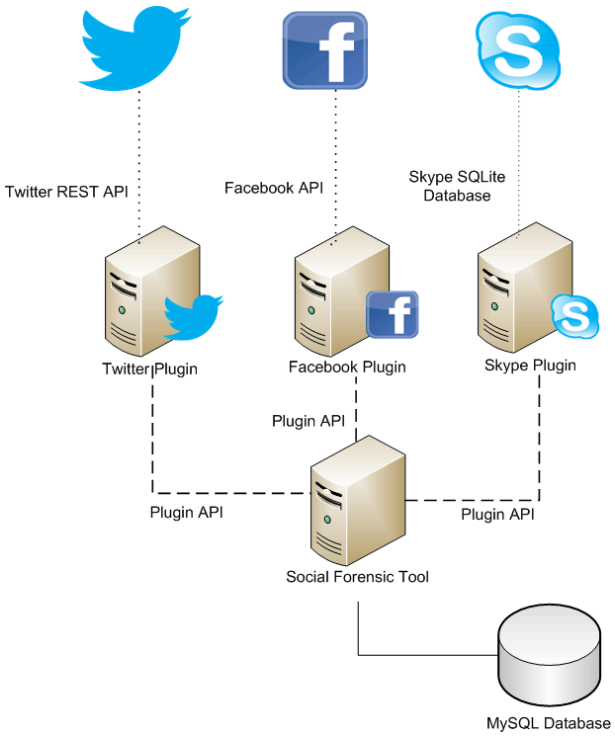


Fig. 5 - Social Media design

Each crawler instance is a plugin that communuicates with the core component of our tool through a common Plugin API. On the other hand, each crawler fetches data from sources, using either API provided (such as Facebook and Twitter) or interacts with local databases and files (like Skype)

Collected data are stored on a MySQL database. Each social network has its own database schema which is different from others. Moreover some notions present on a

social network are absent from others (a friend on Facebook versus a follower on Twitter). This imposes a challenge, in order to combine data from diverse sources under the same database and at the same time preserve the semantic correctness of data. Despite differences, any social network exhibits common structures, for example a social graph.

Using the above observation, we followed a more abstract approach concerning the database schema. Our schema is constructed as a graph, thus each data from different sources must be interpreted on this graph. Each node of the graph represents the notion of account (Facebook profile or page, Twitter account, Skype account etc). Graph edges represent the relationship that two existing accounts have on a social network (Facebook friends, Twitter followers etc). The relationship type can be specified during relationship creation. Moreover, each relationship has a number of interactions. Interactions could be anything, from Facebook comments, likes, posts to individual tweets, and Skype chats.

Thus our database schema consists of three tables, "accounts", "relationships" and "interactions" respectively. To preserve data semantics of structured data such as URLs, we added a fourth table called "resources". The database schema is depicted on Fig. 6. We used fields found commonly on Facebook and Twitter for our columns.

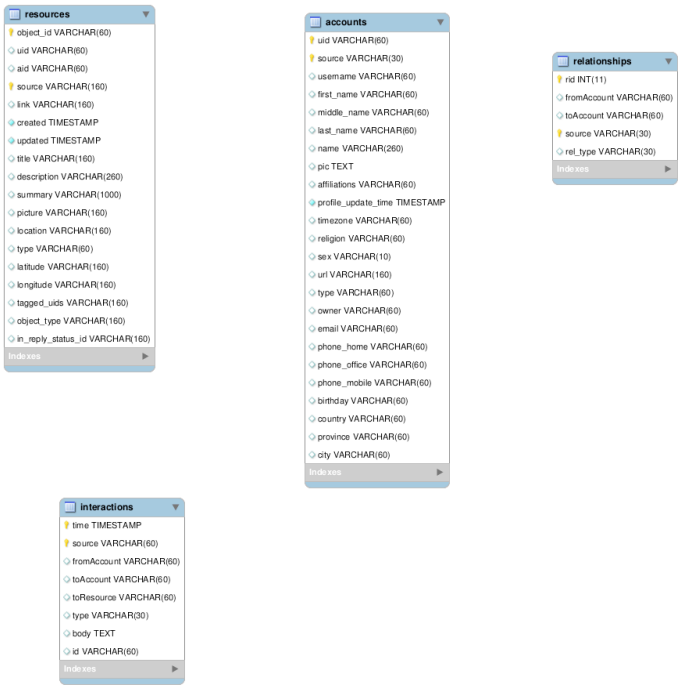


Fig. 6 - Social Media ER

Three plugins have been developed, for Facebook, Twitter and Skype respectively.

Facebook introduced a REST API, so as third-party developers could produce rich content applications for Facebook users. In particular, Facebook provides a Graph API (https://developers.facebook.com/docs/reference/api/) for accessing entities on Facebook social graph, as well as Facebook Query Language (FQL), similar to SQL syntax, to perform more advanced queries.

To protect end users from malicious applications, Facebook uses OAuth2.0 (http://oauth.net/2/) in order to prevent some application to access various aspects

of user's profile. Each application is bundled with a number of permissions to (read-only or read-writable) resources where the application can perform queries.

We leveraged fbconosle (https://github.com/facebook/fbconsole), a Python wrapper for Facebook API calls to perform requests. During authorization stage, we requested permission at any read-only resource, to prevent tampering of data during information gathering. Providing with the correct credentials (username/password) of an individual we have access to a variety of resources like friends, profile information, photo albums, direct messages (thus Facebook chat logs), likes, Wall posts, notifications etc of the account. Any available information on the individual or on its friends profile is downloaded. This gives us a broader scope of actions as well as information for other users that are unaccessible otherwise (some facebook friends restrict access to specific pieces of information e.g album to a limited scope). Our crawling mechanism has the ability to fetch data recursively at a deeper depth.

A similar strategy was followed for Twitter crawling. Like Facebook, Twitter has its own REST API (https://dev.twitter.com/docs/api) for developing custom applications. It uses OAuth2.0 for authorizing application access to a user's profile. Moreover, there exists a wide variety of Python wrappers for Twitter API. We chose tweepy (http://tweepy.github.com/) for our crawling mechanism. Despite common strategy with Facebook, Twitter implements a rate limiting mechanism on requests performed through its REST API.

This would limit us into 350 requests per hour. This may lead information gathered would not be "forensically sound". To overcome this obstacle, we requested resources that are private (such as direct messages) at the beginning. Moreover, during tweet collection from friends and followers, accounts with small amount of tweets are crawled first. When rate limiting occurs, the crawler pauses and sleeps for a pre-defined period of time.

Skype keeps various information like chat logs, and call stats on an SQLite database called main.db, locally, under pre-defined locations on each system. Many applications have been released in order to present data extracted from this database. Among tables exist on main.db, we can observe tables like "Conversations", "Accounts", "Contacts", "CallMembers", "Chats", "Transfers" and "Messages". These contain detailed information about Skype accounts that have logged in from this computer, many chat logs, file transfers between Skype contacts, call information and many others. We gather all information present on these tables and make the appropriate mappings on our database, so as they are semantically correct.

3.2.1 Social Media Management and Monitoring Hubs: an overview

Given that one significant component of ForToo involves monitoring social media to conduct pro-active forensics which will deter future cyber attacks, an overview of the many pre-existing social media management and monitoring hubs is warranted. Although the exact function of individual hubs depends largely on their target use – many hubs are designed to be effective marketing tools, for example – the hubs overall share many common functionalities, such as being able to monitor different social websites for specific content as well as interface with multiple social websites at the same time.

Social media hubs which manage content, monitor social websites and aid with marketing campaigns are so numerous that the marketing blog TopRank has

compiled a list of 22 social media marketing management tools (http://www.toprankblog.com/2010/09/22-social-media-marketing-management-tools/). Although the list is from 2010, it demonstrates not only how fast-growing the market is for social media management hubs but also how longstanding the market is. Companies such as HootSuite, HubSpot and Awareness interface with a number of social websites, including Facebook, Twitter and YouTube, and enable users to post content across multiple websites, monitor public posts containing keywords specified by the client, target content to specific customer bases by specifying criteria such as location and language, and create and co-ordinate marketing campaigns across multiple websites.

Monitoring services such as DataSift and Gnip are popular as well. These services focus on building specific queries to search through aggregated data from social websites. The end user is then able to monitor multiple websites for particular content. ForToo's pro-active forensics component most closely resembles such monitoring services and as such researchers may want to further examine their technical functionality.

3.2.2 Related Social Network Terms and Conditions

One mandate of the ForToo research project involves creating a toolset for proactive forensics – that is, a toolset that can be used to prevent future cyber attacks. As this toolset uses data gathered from several social media websites via their application program interface (API) or database, ForToo researchers will need to contact these companies to obtain permission to use their data for research purposes.

The toolset currently uses the Twitter REST API, Facebook API and Skype SQLite database to download available information from an individual's social network accounts, provided that the toolset has the accounts' credentials. Other social networks can be added and social network crawlers can also be easily integrated into the toolset. Data is fed to a common MySQL database, where the different schemas are combined into one overarching dataset.

This presents a challenge as the toolset uses social media APIs to gather private information about users. ForToo researchers will not only need to comply with general terms and conditions specified on each website, but will also need to comply with terms and conditions formulated specifically for application developers.

Term 2 in Section 1 of Facebook's Platform Policies states the following:

> *You must not include functionality that proxies, requests or collects Facebook usernames or passwords.*

Twitter takes a clear stance on the use of their data, which is that users must use the Twitter API when accessing Twitter content or services. Moreover, in Section 8 of the Twitter Terms of Service, Twitter states the following which directly affects the work of ForToo:

> *You may not do any of the following while accessing or using the Services: ... (ii) probe, scan, or test the vulnerability of any system or network or breach or circumvent any security or authentication measures...*

Additionally, Term 4 in Section 1 of Twitter's Developer Rules of the Road contains the following stipulation:

> *You will not attempt or encourage others to … use or access the Twitter API to aggregate, cache (except as part of a Tweet), or store place and other geographic location information contained in Twitter Content.*

Section 6.3 in the Skype Terms of Use is particularly relevant to, and problematic for, ForToo as it specifies prohibited use of the product:

> *You may not:*
> *(a) intercept or monitor, damage or modify any communication which is not intended for you;*
> *(b) use any type of spider, virus, worm, trojan-horse, time bomb or any other codes or instructions that are designed to distort, delete, damage, emulate or disassemble the Software, Products, Skype Websites, communication or protocols;*
> *… (e) use the Software, Products or Skype Websites to cause or intend to cause embarrassment or distress to, or to threaten, harass or invade the privacy of, any third party;*
> *… (g) collect or harvest any personally identifiable information, including account names, from the Software, Products or Skype Websites;*
> *… (i) use or launch any automated system, including without limitation, robots, spiders or offline readers that access the Software, Products or Skype Websites. …*

3.3 Data gathering from mobile devices

There is an increasing trend for using data stored on a mobile phone as evidence in civil or criminal cases (Rizwan and Dharaskar, 2009). Various types of information can be acquired from a mobile device using commercial forensic tools. The data could be phone books, logs of phone calls, SMS and MMS messages, Email, Instant Messaging content, URLs and content of visited Web sites, Audio, Videos, Images, SIM content (Hoog, 2011). Due to the fact that mobile devices can be found in many crime scenes, the capability to perform forensic analysis on them has become important to law enforcement.

Hoog (2011) introduced numerous forensics acquisition techniques for the Android Operating System, however their drawback is that the investigator must have root privileges to perform the analysis on the phone. SD card analysis, logical and physical data acquisition and the chip-off method related to the NAND memory are some of the most popular forensic methods. We are using the Android Debug Bridge (ADB) tool, which can be installed along with the Android Software Development Kit (SDK) and provides the connection between an Android smartphone device and a remote workstation, such as a personal computer. The ADB tools will help us to do Physical Imaging (with the utility dd (or nanddump). Due to its nature, Unix-like traditional forensic command-line tools such as dd, which allows a bit-by-bit physical imaging of Unix files, can work with the Android system conveniently (Distefano et al., 2010).

3.3.1 Using tools for physical acquisition of Android smartphone partitions

The investigation (Linux) machine must have the Android Debug Bridge (ADB) tool installed, which is provided with the Android's software development kit (SDK) from the official Android developers site. Especially for the 64-bit machines the investigator should first install some 32-bit libraries by typing:

sudo apt-get install ia32-libs

Before starting the investigation of the seized smartphone the forensics examiner is advised to read the documentation for the ADB tool which can be found online in the official Android developers web site. ADB commands can be executed from the command line on the development machine. Here we demonstrate the use of ADB's start-server, kill-server, shell, pull and logcat commands.

The ADB tool requires the USB debugging option of the phone to be enabled. At this point we should clarify that the method will be successful on rooted Android smartphones, meaning that the investigator has Super User privileges and also the BusyBox application (or similar) is installed and works properly as discussed in relevant bibliography (Sylve et al., 2012). A popular way to root an Android phone is by using programs such as SuperOneClick provided by shortfuse.org. The Super User and BusyBox Apps are provided by the Play Store (Android Market) at no cost. The BusyBox application contains Unix utilities for the phone and we will use the df command with the option -h to see the partitions relevant to our investigation. If the phone uses MTD devices we can see the partitions by inspecting /proc/mtd. Moreover, the mount command provides information about the file system of the partitions.

The method for evidence acquisition from the smartphone consists of two major parts. First, we keep the phone's log files stored in the main and the events ring buffers. (We can use the logs from the system and radio buffers but here we obtain only the events and main buffer logs). Afterwards we take a physical image of the device's critical partitions (data and system). Having performed those steps, we are able to safely examine the images by mounting them in our examination machine as a virtual disk. The use of an SD card is vital for the specific method. The SD card must be formatted and wiped and its capacity must be bigger than the phone's internal memory.

In details, the investigator should:

- Make sure all the actions taken during the investigation are recorded. Date and time must be clearly marked on the record.
- Obtain the mobile device and check whether the smartphone device has any SD card.
- Activate the 'airplane mode', which turns off all the wireless connections of the smartphone device.
- Activate the USB Debugging mode (usually located in Settings, Applications, Development, USB Debugging).
- Connect the USB Cable from the developing machine to the smartphone device.
- Start the ADB server by typing the command: sudo adb start-server
    - (We assume that the invoked tool is stored in the path)
- Use the logcat tool to dump (-d option) the logs of the Android smartphone device which are formatted to show the time (-v option) using the commands:
    - adb logcat -d -v -time -b main > file
    - adb logcat -d -v -time -b events > file

- Use the md5sum utility of the developing machine to produce a checksum for each and every file and then type: sudo adb kill-server
- Turn off the device, remove any SIM card or SD card and send them to specialized laboratories for further examination.
- Install a new formatted and wiped SD card in the smartphone device and boot up the device. It will automatically mount.
- Gain root access on the smartphone device and repeat steps 5 - 6 in order to start the daemon.
- Use the command su to gain root access when the shell has been established (adb shell). The hash sign (#) on the shell indicates the super user state.
- Type the command cat/proc/mtd to find out the partitions structure if the device is using MTD partitions. If not, type (busybox) df -h to identify the partitions. (Command in parenthesis is not compulsory.)
- Type mount in order to recognize the file system of the partitions.
- Use the dd command to image the data partition and store it in the SD card: (busybox) dd if=/x/y of=/SDCARD/NAME.img bs=4096 where /x/y represents the partition that will be copied from the smartphone device (e.g. /dev/stl13 for Samsung's data partition), SDCARD represents the path of the SD card's partition on the smartphone device and NAME represents the name of the image file that will be created. We are interested in the partitions that refer to the user data or data and to the system. If the device uses the YAFFS2 file system, then the investigator should use nanddump instead of dd (Vidas et al., 2011).
- Type exit twice after all images have been retrieved successfully.
- Use the command adb pull /SDCARD/NAME.img /PATH_DOWNLOAD to download the image that has been created. PATH_DOWNLOAD is the desirable path to store the image file on the development machine.
- Use the command sudo adb kill-server to stop the server.
- Use the md5sum utility of the developing machine to produce a checksum for each image. md5sum -b NAME.img
- Remove the USB cable and store the mobile device in a secure place according to the local authorities' standards.
- Browse to the folder where the image file was downloaded from the device and use the mount command to mount the image on your development machine, e.g. for a Samsung Galaxy Europa the examiner should type: sudo mount -o loop NAME.img path (path is the full path where the image file will be mounted).
- Perform the investigation of the proper files.
- Unmount the image file at the end of the examination using the command sudo umount path (path is where the image file was mounted).

The log files are stored on the device in circular buffers and the storage ability of those buffers differs between Android smartphones. For instance, the main buffer on a Samsung Galaxy Europa is 256 Kb, while HTC Desire and LG Optimus E400 can only store up to 64 Kb. Because of the nature of circular buffers, time becomes a crucial enemy for the forensic examiner. Thus, it is highly possible to find important information, if the investigator seizes the phone right after a crime has been committed. The most interesting findings will be derived investigating the relevant

databases to the specific case. For example, when an analyst seeks evidence of the wireless facilities of an Android Smartphone (version 2) the files that could be of interest are shown in the table below (Andriotis et al., 2012).

Path	File name
/misc/bluetoothd/MAC (MAC: device dependant)	classes, config, lastseen, linkkeys, names, profiles
/data/com.android.bluetooth/databases	btopp.db
/misc/wifi	wpa_supplicant.conf
/data/com.google.android.server.checkin/databases	checkin.db
/data/com.google.android.gsf/databases	htcCheckin.db
/data/com.google.android.location/files	cache.wifi, cache.cell
/data/com.android.browser/databases	browser.db, webview.db, webviewCache.db
logcat files and application related databases e.g. Dropbox	main, events buffer and Dropbox's db.db

Fig. 7 – Sources of Evidence

3.4 Steganography Detection in JPEG Images

This section describes the design of tools StegBennie and CompBennie, which together make a novel approach to the problem of steganography detection in images by applying a statistical attack. The method is based on the empirical Benford's Law and, more specifically, on its generalised form. We have devised a blind steganographic method which can flag a file as a suspicious stego-carrier. The proposed method achieves very high accuracy and speed and is based on the distributions of the first digits of the quantised Discrete Cosine Transform coefficients present in JPEGs. In order to validate and evaluate our algorithm, we developed steganographic tools which are able to analyse image files and we subsequently applied them on the popular Uncompressed Colour Image Database. In addition to steganography detection, if certain criteria are met the method can also reveal which steganographic algorithm was used to embed data in a file.

3.4.1 Benford's Empirical Law of Anomalous Numbers

The first attempt to decode the behaviour of the first digits in a set of natural numbers was conducted towards the end of the 19th century by Newcomb (1881). This note presents a table which lists the probabilities of occurrence of the first digits of a set of natural numbers. Numbers cannot be zero and they have more than one digit. Fifty years later Benford (1938) rediscovered and restated the law. He investigated large groups of natural numbers and observed that, in all selected groups, the probability of the first digit x of a number being 1 is higher than that of the first digit being 9. Furthermore, the distribution of the appearance of the first (or significant) digits in a set of natural numbers follows a logarithmic rule. Therefore:

$$P(x = 1) > P(x = 2) > \ldots > P(x = 9).$$

The mathematical equation which describes the first digits law is:

$$p(n) = \log_{10}\left(1 + \frac{1}{n}\right),\ n = 1, \ldots, 9.$$

$p(n)$ represents the probability of n being the first digit of a number in a set of natural numbers. Sets should contain as many numbers as possible in a random fashion. This

empirical law is applicable to different groups of natural numbers such as population, addresses, drainage and death rates.

According to this empirical view we are able to predict that, in a set of natural numbers, it is more probable to find numbers with the significant digit to be 1 than 8 or 9. This law looks like it fights against common sense but it is now widely used in the area of expenses and accounting fraud detection and was also introduced in various social occasions. For instance, detected fraud and fake survey results using the Benford's Law (Schäfer et al., 2004). The basic principle behind all examples is that natural data generally follow the first digit law quite well in contrast to maliciously changed or randomly guessed data. Some attempts to utilize the results of the findings of the logarithmic law can also be encountered in literature related to image processing and digital forensics (Jolion, 2001; Fu et al., 2007; Pérez-González et al., 2007).

3.4.2 Generalized Benford's Law

In 2007, Fu et al. presented a new approach to image forensic analysis using the law of anomalous numbers and studied in depth the behaviour of the JPEG image block coefficients (Fu et al., 2007). In this work there are some conclusions about the validity of Benford's Law in the most significant digits of DCT coefficients (before quantization) of the 8x8 pixel blocks of any grey scale JPEG image; the DC coefficients are excluded from the research. Experiments were conducted considering only 8 bit grey scale pictures, using as main reference a widely used dataset of TIFF images called the Uncompressed Colour Image Database (UCID) (Schaefer and Stich, 2004). The use of such a database guaranteed that those images have never before been JPEG compressed. They also examine the distribution of the first digits of the quantized DCT coefficients that emerge after the quantization process. After completing the calculation of the appearance of significant digits of the DCT coefficients in this set of images, their mean distribution was obtained. The significant digits of DCT coefficients conform quite well to the Benford's Law, with goodness of fit results confirmed by using x^2 divergence.

By conducting thorough experiments on the same set of images, the authors also calculated the mean distribution of the first digits of the quantized DCT coefficients under different quality compression factors (QF = 100, 90, 80, 70, 60, 50). The results show that the distributions of those coefficients also follow a logarithmic trend. A comparison between the mean distributions that they obtained for each compression quality and the expected Benford's Law distributions revealed that the quantized coefficients do not follow the rule in a very strict way as the DCT coefficients do. However, there is also a logarithmic law behind the distribution of the first digits of the quantized DCT coefficients. The model they proposed is described by the following equation:

$$p(n) = N \cdot \log_{10}\left(1 + \frac{1}{s + n^q}\right),\ x = 1, 2, \ldots, 9$$

N, s and q are parameters which describe precisely those distributions under different compression quality factors. In the special occasion where N=1, s=0 and q=1 we can conclude that the equation, which is called the generalized Benford's Law (gBL) (Fu et al., 2007), is equal to the Benford's Law Equation.

3.4.3 Method and Algorithm

The attack designed as part of this project is based on an analysis of the quantized coefficients of a large amount of colour images. Our method indicates that it is possible to predict the behaviour of the distributions of their significant digits and any disturbances of these distributions can then be considered an indication of the presence of steganography. By studying the deviations of their distributions, we propose a decision making model based on our findings related to the behavior of digit 2. Moreover, we developed a set of automated tools which implement the attack and can be used to conduct blind steganalysis and thus help forensic analysts to identify suspicious colour images. In order to validate the method and assess its performance, we used it to analyze files taken from a widely-used database of approximately 1340 images, enriched by our own set created by the use of a smartphone.

Our method focuses on the distributions of the significant digits which can be extracted from the quantized coefficients of colour images. The decompression of a JPEG image is exactly the inverted process of what we presented in a previous section. We have previously underlined that the gBL was proposed by investigating 8 bit grey scale images only. Thus, only the luminance of each image was taken into consideration.

For this reason we decided to investigate the behaviour of the quantized DCT coefficients of all the components of a image; both luminance and chrominance. The investigation contributes to previous work by extending the results and by creating a new reference as a basis to describe the expected distributions of the quantized DCT coefficients of a colour image. The knowledge of the compression quality is critical at this phase. The compression quality factor can be revealed by looking at the image's metadata.

The basic steps of our method include the calculation of the appearance of the significant digits of the quantized DCT coefficients of all the components of a colour image. For example, if the first row of an 8x8 block of coefficients is [154 32 1 19 2 0 0 0], the first digits are [x 3 1 1 2 x x x] (154 is the DC coefficient and it is excluded and also the zeros are not taken into consideration). Then we estimate their expected distribution (given by the gBL equation) and finally compare the deviations between the expected and the calculated distributions. We use this information to decide if the image is suspicious or not. In some cases, the same data can be used to determine exactly which steganography algorithm was used to embed the hidden object.

In order to achieve this, we need a model to represent the distributions of the quantized DCT coefficients of any colour image. This can be feasible if we prove that gBL is still a reliable model that describes the probability of appearance of the first digits of the quantized DCT coefficients of a image, even if these were collected from all the components of the image; luminance as well as chrominance. We used the second version of the UCID for this experiment which contains 1338 uncompressed TIFF images. A Matlab script was written to compress them with different quality factors. The script used Matlab's functions *imread* and *imwrite* and compressed the images within seconds. As a result, we accumulated seven groups of 1338 images that had never been compressed before. This step was vital in terms of accuracy because we were able to know the compression history of each image. Secondly, we calculated

the distributions of the first digits of the quantized DCT coefficients. After this step the mean distributions for each digit were calculated by Matlab. The algorithm that was used can be described by the following pseudo code.

```
decompress_image();
for all components {
  for each DCT block {
    consider only AC coefficients;
    extract_first_digits();
    distribute_first_digits();
  }
}
calculate_percentage_of_appearance();
```

We estimated the goodness-of-fit of the generalized Benford's Law using Matlab's Curve Fitting Toolbox. To avoid the calculation of any complex values from Matlab we had to define the boundaries of parameters N, s, q. The use of the curve fitting toolbox for all quality factors resulted in the conclusion that gBL can describe the distributions of the appearance of quantized DCT coefficient first digits of a colour image in a very satisfactory manner. As a matter of fact, the statistics that Matlab provides to estimate the fitting results, show that the gBL describes the mean distributions perfectly (R-Square=1, Adjusted R-square=1). Fig. 8 presents the values of parameters N, q, s for each quality factor. There is also a column which represents the Sum of Squares Due to Error (SSE). SSE is another fitting statistic that Matlab provides and the figure shows that in our case this error is infinitely minor.

We are now able to calculate the expected distributions of the appearance of the quantized DCT coefficients. The idea behind this concept is that given the quantization table of the luminance of a JPEG image, we can obtain the compression quality that was used during encoding. Afterwards, we can calculate the distributions of appearance of the coefficients and compare them with the expected distributions. We will be able to estimate the deviations between the distributions (current and expected) and decide if the image is suspicious or not. In our research we used the percentage of the deviations because it makes the comparison between first digit distributions more reasonable. For example, digit 9's distribution is always between 1 - 2% and digit 1's distribution can vary from 55 - 60%. Their deviations should be measurable and comparable and this is why we should use the % of deviations as a common measurement system.

Quality factor	*N*	*q*	*s*	Goodness-of-fit (SSE)
100	1.608	1.605	0.0702	5.129e-06
90	1.25	1.585	−0.405	7.235e-07
80	1.344	1.685	−0.376	3.007e-06
75	1.396	1.731	−0.3549	3.986e-06
70	1.434	1.766	−0.339	4.455e-06
60	1.514	1.843	−0.3114	5.464e-06
50	1.584	1.909	−0.2875	5.119e-06

Fig. 8 - Goodness of fit for the gBL model for luminance and chrominance.

Subsequently, we measured the impact of steganography on these distributions. We chose random images from our seven sets (each set contained 1338 files) and we embedded data with JPHide, Outguess and Vsl. JPHide and Outguess hide data in

the quantized DCT coefficients. We then calculated the deviations of the distributions of the first digits of the quantized DCT coefficients for each potential stego carrier. By doing this, we gained a clear picture of the consequences that these algorithms cause to the distributions of the first digits. Fig. 9 shows the % deviations caused to an image when we embedded a text file with JPHSWIN and their (absolute) difference.

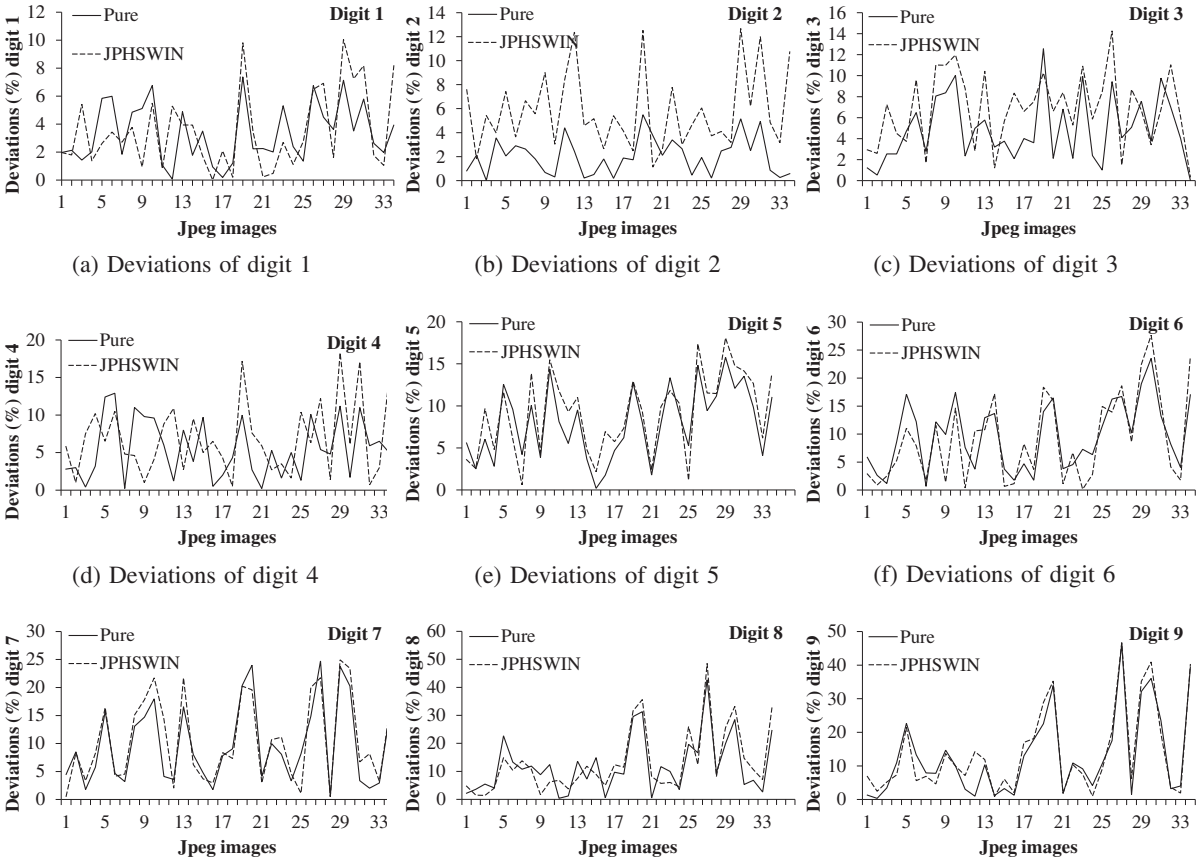
First digits	Deviations (pure)	Deviations (stego)	|Difference|
1	5.117569	0.947102	4.170467
2	0.678150	9.001642	8.323492
3	8.373005	10.988395	2.61539
4	9.832138	1.039585	8.792553
5	3.874760	4.447051	0.572291
6	9.937180	1.376626	8.560554
7	14.700152	17.820417	3.120265
8	8.818516	1.664183	7.154333
9	14.713667	13.687573	1.026094

Fig. 9 - Difference between deviations of distributions in a pure image and a stego carrier.

At this stage we tried to verge on the issue of finding a reliable indicator that could safely reveal the suspicious image. We focused our interest on the deviations of the distributions of the first digits of the quantized coefficients of pure images and stego carriers. The stego carriers were created by the same pure images but they also contained messages (in .txt format) which were embedded by JPHSWIN. We carefully examined about 480 images compressed with different quality factors. Fig. 10 illustrates deviations of first digit distributions for images compressed with quality factor 75. The solid line indicates deviations that emerged from the inspection of pure images and the dashed line indicates the deviations for the same images after applying steganography on them. The horizontal axis of the preceding figures represents the images we examined and the vertical axis states the percentage (%) of the deviations of the distributions of the examined digit. The overview we got by examining the figures we formed from images that were compressed by various quality factors was similar to what we can see in the figure.

The analysis of the difference in deviations of the distributions of pure images and their respective stego carries reveals that the differences are in most cases larger than 5%. Furthermore, we observe that differences in deviations are more extreme for digits 2, 4, 6 and 8. Subfigure Fig. 10.b further reveals a characteristic of digit 2 that no other digit seems to have. When we examined pure images the deviations of digit 2 were very stable. The range of these deviations was quite convenient and usually varied from 0 to 3 or 4%. Except from that, the deviations of digit 2 after the embedding of a message on the same images behaved in a similar fashion, but this time the deviations exceed the 4% threshold. We cannot see the same attitude from the digit 1 for example (Fig. 10.a). Here, the deviations are within a small range but we can see that the two lines do not have the same behaviour compared to the two lines of Fig. 10.b. In Fig. 10.b we can see that the solid line is almost always below the dashed line. Thus, in most of the cases, we expect that an image which contains a hidden message will present deviations which are higher than a certain threshold T. In the specific example, a suitable threshold would be $T = 3$. It becomes more interesting if we underline that the thresholds for all examined quality factors vary between 3 and 4. Taking these findings into consideration we concluded that the most stable and reliable indicator for a suspicious image to be revealed is the

deviation of digit 2. If this deviation exceeds a specific threshold, which depends on the quality factor of the compression of the examined image, we can conclude that the image is suspicious. We approximated these values statistically for each compression quality and present them in Fig. 11.



(a) Deviations of digit 1 (b) Deviations of digit 2 (c) Deviations of digit 3

(d) Deviations of digit 4 (e) Deviations of digit 5 (f) Deviations of digit 6

(g) Deviations of digit 7 (h) Deviations of digit 8 (i) Deviations of digit 9

Fig. 10 - Deviations of first digits for quality factor 75

We should underline at this point that we did not manage to verify the accordance of the previous results with images that were compressed with QF = 100. As a matter of fact, both deviations and differences between the pure images and stego carriers seem like they do not follow any rule that complies with our findings for the other quality factors. This phenomenon occurs because when compressing with quality factor 100, the quantization tables have a very weak effect on the first digits.

QF	Threshold
50	4.00
60	4.00
70	4.35
75	3.00
80	3.11
90	2.90

Fig. 11 - Threshold for quality factors

We repeated the same tests to images using Outguess and Vsl as the embedding algorithms. We analyzed the data using the same methodology and discovered that

the impact of steganography on the distributions of the first digits was significant. Often the difference between the expected and the given deviations was more than 70%. We also confirmed that the deviations of digit 2 were smooth and the thresholds discussed above were sufficient and capable to detect a suspicious image. A closer look at the effects of the application of steganography with Outguess and Vsl revealed that both algorithms change the image quantization tables when they embed a message into their internal structure. Outguess always uses the quality factor of 75 and Vsl always quantizes with QF = 100. Consequently, the expected distributions of the inspected images are significantly different than the observed ones. Our research revealed that Outguess leaves the quantization table of quality 75 as a fingerprint or signature. The same goes for Vsl which turns the quality factor of the stego carriers to 100. In other words, the metadata of a stego carrier created by Outguess or Vsl will always indicate that the quality factor used to quantize the block coefficients is QF = 75 or QF = 100, respectively. We used these fingerprints when we built the decision making module of our programs. If we try to investigate an image which has a quality factor of 75 or 100 and the deviation of digit 2 is really large, we can deduce that Outguess or Vsl was used, respectively.

The research on the behaviour of the first digits of the quantized DCT coefficients of colour images and the analysis of the data we gathered from their distributions and deviations resulted in the development of a new universal steganalytic tool which we called *stegBennie*. This tool uses the characteristics of the distributions of digit 2 and it is a new approach to the problem of steganalysis of colour images. It applies a statistical attack on a image using the generalized Benford's Law and estimates whether it is a suspicious image or not. It is an extension of the first tool we developed which was responsible to collect data from the images and calculate the distributions and their deviations from the expected. We call the latter tool *compBennie*.

3.5 Visualisation Architecture

The visualisation component of the work will look at event correlation techniques in order to understand what has been happening. Rather than developing a single "one size fits all" monolithic application, a different approach will be taken when creating visualisations for the ForToo project. A system whereby "visualisation tools can talk to other visualisation tools" will be developed.

The Unix philosophy is particularly well suited to the process of designing visualisation tools. Analysts should be able to choose visualisation tools that they, as individuals best respond to and be able to rapidly deploy new tools should a need arise. The philosophy states individual tools should "do one thing and do it well", that work together, and can handle a standard input / output.

Visualisation tools that adhere to this philosophy will benefit greatly. Applications naturally become more interoperable with others as they more readily accept data from a multitude of sources (not just from another visualisation within the same application) and will encourage data exchange as tools will be programmed with one specific visualisation paradigm, unlike larger multi-faceted applications which tend to retrieve and retain a great deal more data.

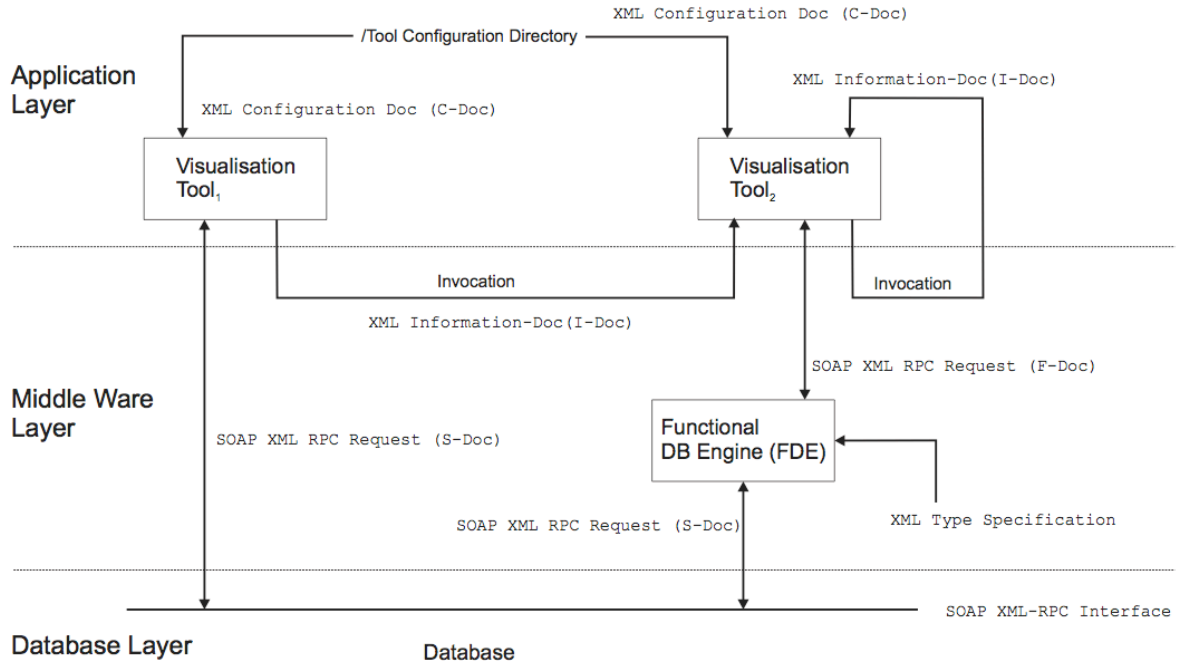


Fig. 12 - Visualisation Architecture

The exchange of data from one tool to another makes use of three components. The Configuration Document or C-DOC, is an XML file which stores the configuration of a visualisation tool. It is used within the context of data exchange between tools to identify the input a tool can accept. The Information Document or I-DOC is an XML file generated by one tool containing the data required from the underlying database. It is passed as a command-line argument to another tool, which uses the id / function combination to query the database, see Fig. 12.

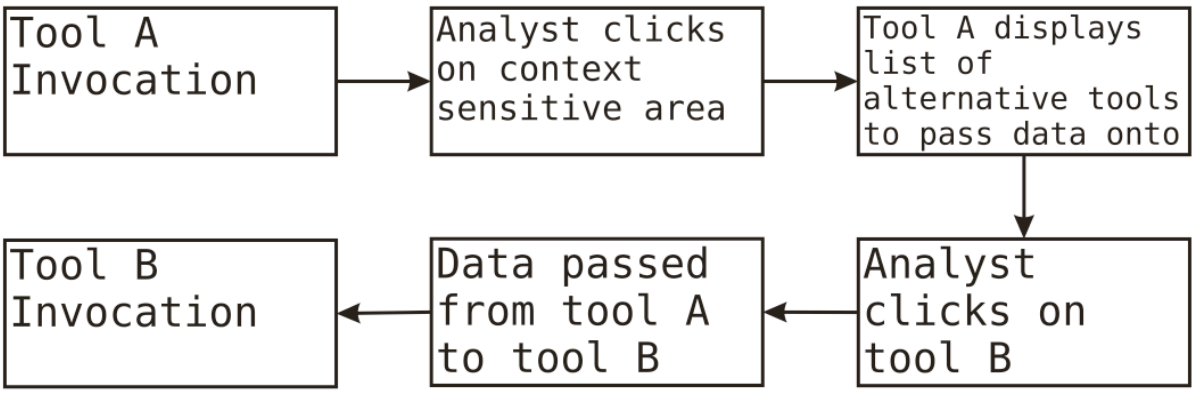


Fig. 13 - Dataflow when invoking visualisation tools

Objects are context sensitive areas programmed into an application. An object represents one or more function types and holds one or more events. When clicked the object will find compatible applications that can visualise the data it holds. This is summarised in Fig. 14.

Document Type	Purpose	Usage
C-DOC	Stores application configuration data	By other tools when identifying applications that accept their data types
I-DOC	Contains function reference and primary key of an event	This is the file passed between tools
Objects	To provide locations where analysts may select data for transfer within an application	Generated right-click menu presenting analyst with compatible tools

Fig. 14 - Summary of components permitting data-sharing between tools

3.5.1 Objects

An object can be better described as a context sensitive area within a visualisation tool. It is the pre-defined region of an application that enables users to interact with the visualisation by performing gestures, such as clicking a particular button on the mouse. The object denotes a logical and convenient location within the tool where it is possible to obtain a subset of data derived from the current dataset being visualised. By clicking on the object the typical operation is to create a menu displaying the other tools the current visualisation may pass data onto. Objects are the initiators of the data exchange between tools; they provide a convenient way of interacting with a visualisation.

Tools are inherently different, visualising different types and quantities of data. Objects in question can show the quantity of data being provided. Through the objects, a standard method is used to pass data seamlessly between tools.

Using an object is a seamless affair. Upon clicking on an object in one visualisation, the tool then carries out a number of steps, ultimately resulting in the generation of an I-DOC and it being passed to another tool.

In terms of design, there are no formal requirements as to where objects should be placed. Objects should however be intuitive and provide the most logical data types based on what the object represents. For example, clicking on the hand of a clock visualisation could make use of a “time & date” type, clicking on a link between two mobile phones could open up options relating to social media received from such devices. The most appropriate areas of a visualisation should have objects associated with them, as this is where users’ attention is naturally drawn.

3.5.2 Configuration Document (C-DOC)

The C-DOC holds all the configuration data about a tool, such as acceptable input, location on the hard disk and a brief description of the tools' function. There may be more than one C-DOC associated with any one tool (i.e. a one to many relationship). For any given tool, its numerous C-DOC's differ by the acceptable input each provides. If a tool is versatile enough to accept several forms of input, it would need several C-DOC's each specifying the input There are several reasons for this approach, opposed to one large configuration file.

- Avoids the creation of monolithic configuration files which are difficult to maintain.
- Reduces the problems associated with incorrectly modified C-DOC's. If a tool is updated to work with different data types, separating out these changes into different C-DOC's allow existing stable configurations to run alongside new functions.
- This approach is more in-line with the Unix philosophy. Individual C-DOC's are created, each with a different function name as valid input.

3.5.3 Information Document (I-DOC) Specification

An I-Doc is an XML file automatically generated by an application after a user clicks an object and selects a tool to pass data onto, stored with a unique name so that it is not overwritten. Data persistence is important in the architecture as tools must have the ability to be re-invoked with previously viewed data to generate the same visualisation. It is passed as input to a tool via a command line argument.

The I-DOC is used as an interoperable container to pass data between different visualisation tools. For the set of data it represents, the I-DOC carries the event id and a function name (data-type) for each event. There is a one-to-one relation between function names and event id's; there will always be the same number of each in an I-DOC.

3.6 Visualisation Tools

A number of visualisation tools will be developed and/or modified to provide the following functionality:

- Address the geographic nature of the data sources
- Overlay multiple evidence types to get a fuller picture of events
- Organise data by date/time irrespective of data source

The following are examples of how this may be achieved.

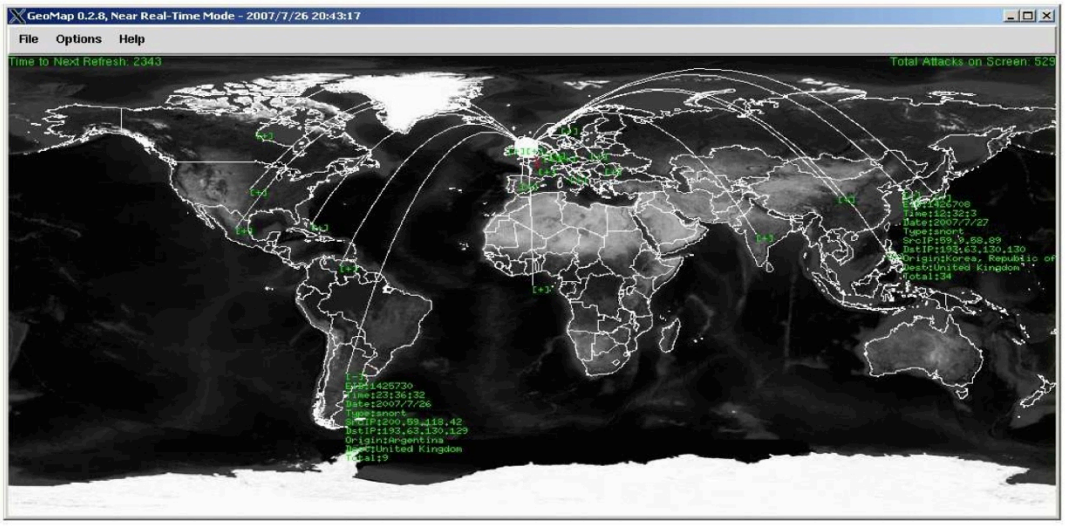


Fig. 15 - Geographical Tool

3.6.1 Geomap Tool

Geomap can query the database to retrieve events that contain geographic data. If there are complimentary events that have identifiable links to each other, lines may be drawn between them to better understand the relationships between different evidential sources.

Green boxes are always positioned at the source end of the line, providing detailed summary information of the event.

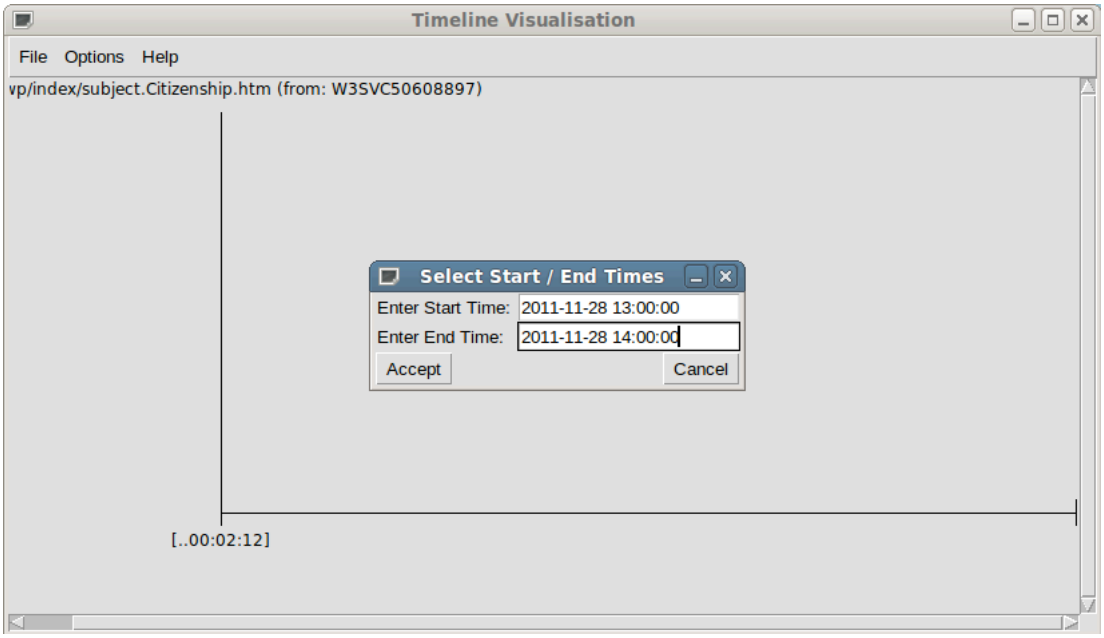


Fig. 16 - Timeline Tool

3.6.2 Timeline Tool

Timeline uses the date/time of an event to produce a diagram where the earliest event to occur appears on the left, the last event to occur appears on the right, and all other events in between are offset accordingly.

Overlays of different evidence sources will help to arrange an evidence trail in terms of sequence of events. Data from social media and local PCs can be overlaid and toggled.

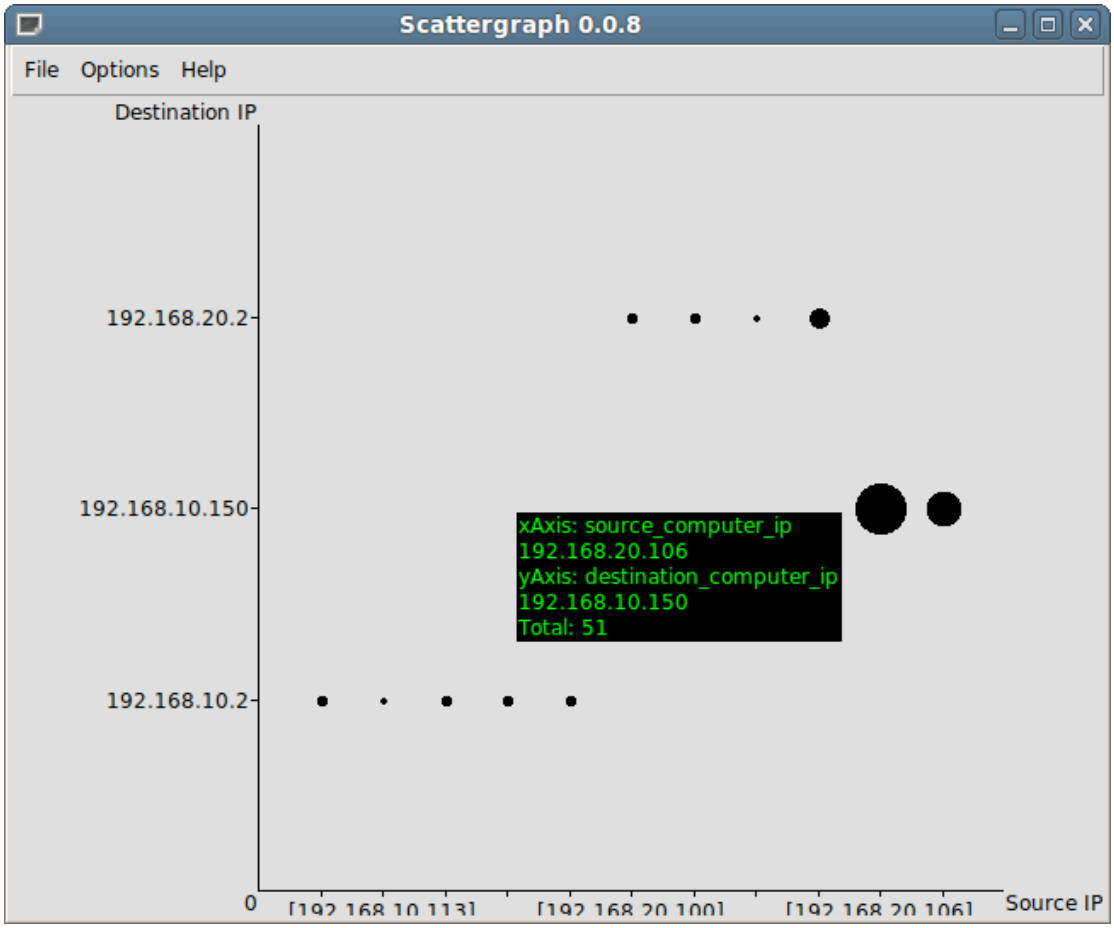


Fig. 17 - Scattergraph

3.6.3 Scattergraph Tool

Scattergraph provides a horizontal and vertical axis and takes any form of data available in the database with which is put on the axis. Data on the axis is, by default, sorted alphabetically, however special consideration is given to date/time and ip address types. Date/time is converted into a single integer value before sorting to ensure correct time order is kept. IP addresses are converted into integer representations to ensure they are sorted correctly. Dots are drawn at the position between horizontal and vertical when the respective data intersects. If there are several intersections at one point, the additional events are appended to the existing dot.

References

F. Adelstien, B. Carrier, E. Casey, S. Garfinkel, C. Hosmer, J. Kornblum, J. Lyle, M. Rogers, and P. Turner, "Standardizing digital evidence storage", Communications of the ACM, vol. 49, no. 2, Feb. 2006, pp. 67-68.

Ahmed, R., and Dharaskar, R. V. (2009) Mobile forensics: An introduction from indian law enforcement perspective. In Proc. Third International Conference on Information Systems, Technology and Management (ICISTM 2009), pages 173–184.

Andriotis, P., Tryfonas, T., Oikonomou, G., and Yildiz, C. (April 2013) A pilot study on the security of pattern screen-lock methods and soft side channel attacks. In Proceedings of the sixth ACM conference on Security and privacy in wireless and mobile networks, WiSec '13, pages 1–6. ACM.

Andriotis, P., Oikonomou, G., and Tryfonas, T. (2012) Forensic analysis of wireless networking evidence of android smartphones. In Information Forensics and Security (WIFS), 2012 IEEE International Workshop on, pages 109–114. IEEE.

Antonatos, S., Polychronakis, M., Akritidis, P., Anagnostakis, K. G., & Markatos, E. P. (2005). Piranha: Fast and memory-efficient pattern matching for intrusion detection. In Security and Privacy in the Age of Ubiquitous Computing (pp. 393-408). Springer US. Available at: http://link.springer.com/chapter/10.1007/0-387-25660-1_26 (Accessed: 26 May 2013)

Aviv A. J., Gibson, K., Mossop, E., Blaze, M., and Smith J. M. (August 2010) Smudge attacks on smartphone touch screens. In Proceedings of the 4th USENIX conference on Offensive technologies, pages 1–7. USENIX Association.

Bello, A.D. (2011) 'EnCase Automates Response to Security Incidents', *Threat Response*, 18 October. Available at: http://threat-response.guidancesoftware.com/2012/01/encase-automates-response-to-security.html (Accessed: 13 June 2012)

Benford F. The law of anomalous numbers. In: Proceedings of the American philosophical society; 1938. p. 551–72.

Bradford, P.G. (2004) 'Towards proactive computer-system forensics', *Proceedings of the International Conference on Information Technology: Coding and Computing*, Alabama, 5-7 April. IEEE Computer Society. [Online] Available at: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1286727&isnumber=28683 (Accessed: 19 April 2012)

Bradford, G. and Hu, N. (2005) 'A Layered Approach to Insider Threat Detection and Proactive Forensics' Annual Computer Security Applications Conference, Tuscon, AZ, December. Publisher Unknown. [Online] Available at: http://www.acsa-admin.org/2005/techblitz/hu.pdf (Accessed: 26 May 2012)

Bro – The Bro Network Security Monitor, http://www.bro.org

Chandramouli R, Kharrazi M, Memon N. Image steganography and steganalysis concepts and practice. In: Kalker T, editor. 2nd International Workshop on Digital

Watermarking (IWDW 2003). In: Cox I, Ro Y, editors. Lecture Notes in Computer Science, Vol. 2939; 2004. p. 35–49.

Chandramouli R, Subbalakshmi K. Current trends in steganalysis: a critical survey. In: Proc. 8th International Conference on Control, Automation, Robotics and Vision (ICARCV 2004), Vols. 1–3; 2004. p. 964–7.

Corey, V. Peterman, C. Shearin, S. Greenberg, M.S. and Bokkelen, J.V. 2002. Network forensics analysis. IEEE Internet Computing, 6 (6), 60-66.

Common Digital Evidence Storage Format Working Group. 2006. *Standardizing digital evidence storage*. *Commun. ACM* 49, 2 (February 2006), 67-68. *http://doi.acm.org/10.1145/1113034.1113071*

Common Digital Evidence Storage Format Working Group, *Survey of Disk Image Storage Formats*. Survey, New Oreleans: DFRWS.org, 2006, 18.

Desmond, P. (2002) 'Guidance Software Pushes Proactive Forensics', *eSecurity Planet*, 10 July. Available at: http://www.esecurityplanet.com/prodser/article.php/1383281/Guidance-Software-Pushes-Proactive-Forensics.htm (Accessed: 13 June 2012)

DFRWS, (2004), Common Digital Evidence Storage Format, http://www.dfrws.org/CDESF/index.html, Accessed 21 December 2005.

Distefano, A., Me, G. and Pace, F. (2010) Android anti-forensics through a local paradigm. Digital Investigation, 7(1): S83–S94, August 2010. The Proceedings of the Tenth Annual DFRWS Conference.

Fafinski, S. (2007) 'The security ramifications of the Police and Justice Act 2006', Network Security, 2007(2), pp8-11. Available at: http://www.sciencedirect.com.ergo.glam.ac.uk/ (Accessed 25 November 2011).

Forte, D. (2004) 'The importance of text search in forensics', *Network Security*, 2004(4), pp.13-15. *ScienceDirect* [Online]. Available at: http://www.sciencedirect.com.ergo.glam.ac.uk/ (Accessed: 19 April 2012).

Fridrich J, Goljanb M, Du R. Steganalysis based on JPEG compatibility. In: Proc. conference on multimedia systems and applications IV. Proceedings of the Society of Photo-Optical Instrumentation Engineers (SPIE); 2001. p. 275–80.

Fu D, Shi YQ, Wei S. A generalized Benford's law for JPEG coefficients and its applications in image forensics. In: Delp E, editor. Proc. 9th conference on security, steganography, and watermarking of multimedia contents. In: Wong P, editor. Proceedings of the Society of Photo-Optical Instrumentation Engineers (SPIE), Vol. 6505; 2007. pp. 65051L–65051L–11.

Garfinkel Simson L, Malan David J, Dubec Karl-Alexander, Stevens Christopher C, Pham Cecile. Disk imaging with the advanced forensic format, library and tools. In: Research advances in digital forensics (second annual IFIP WG 11.9 international conference on digital forensics). Springer; January 2006

Grobler, C.P., Louwrens, C.P., von Solms, S.H. (2010) 'A Framework to Guide the Implementation of Proactive Digital Forensics in Organisations', *2010 International Conference on Availability, Reliability, and Security*, Krakow, 15-18 February. IEEE Computer Society. [Online] Available at:

http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5438018&isnumber=5437988 (Accessed: 19 April 2012)

Hobarth, S., and Mayrhofer, R. (2011) A framework for on-device privilege escalation exploit execution on android. In Proc. IWSSI/SPMU 2011: 3rd International Workshop on Security and Privacy in Spontaneous Interaction and Mobile Phone Use, June.

Hoog, A. (2011) Android Forensics. Syngress-Elsevier, Waltham, MA.

Jensen, E. (2012) 'Comparison of SQL Server Compact, SQL Server Express 2012 and LocalDB', *Everything SQL Server Compact*, 15 April. Available at: http://erikej.blogspot.co.uk/2011/01/comparison-of-sql-server-compact-4-and.html (Accessed: 1 August 2012)

Jolion J. Images and Benford's law. Journal of Mathematical Imaging and Vision 2001;14(1):73–81.

Kodovsky J, Fridrich J, Holub V. Ensemble classifiers for steganalysis of digital media. IEEE Transactions on Information Forensics and Security 2012;7(2):432–44.

Kornexl, S., Paxson, V., Dreger, H., Feldmann, A., Sommer, R. (2005) 'Building a time machine for efficient recording and retrieval of high-volume network traffic', 5th ACM SIGCOMM conference on Internet Measurement (IMC '05), Berkeley, 19-21 October. [Online] Available at: http://www.icir.org/robin/papers/imc05.pdf (Accessed: 12 September 2012)

Lessing, M and Von Solms, B. (2008) 'Live forensic acquisition as alternative to traditional forensic processes', IT Incident Management & IT Forensics (IMF 2008), Mannheim, 23 - 25 September. [Online] Available at: http://researchspace.csir.co.za/dspace/bitstream/10204/3141/1/Lessing5_2008.pdf (Accessed: 20 June 2012)

Lee K, Westfeld A, Lee. Category attack for LSB steganalysis of JPEG images. In: Digital watermarking (5th international workshop) IWDW 2006 Jeju Island, Korea, November 8–10, 2006. LNCS, Vol. 4283. Springer-Verlag; 2006. p. 35–48. Revised Papers.

Lessard, J., and Kessler, G. C. (September 2010) Android forensics: Simplifying cell phone examinations. Small Scale Digital Device Forensic Journal (SSDDFJ), 4(1).

McBride B, Peterson G, Gustafson S. A new blind method for detecting novel steganography. Digital Investigation 2005;2(1):50–70.

McQueen, T. (2009) *Attack Graphs for Proactive Digital Forensics*. Unpublished research paper. Delaware State University.

Mitrou, L. and Karyda, M. (2005) 'Employees' privacy v.s. employers' security: Can they be balanced?', *Telematics and Informatics*, 23, pp.164-178. *ScienceDirect* [Online]. Available at: http://www.sciencedirect.com.ergo.glam.ac.uk/ (Accessed: 20 August 2012).

Mouhtaropoulos, A., Grobler, M., Li, C.T. (2011) 'Digital Forensic Readiness: An Insight into Governmental and Academic Initiatives', *Intelligence and Security Informatics Conference (EISIC)*, Athens, 12-14 September. IEEE Computer Society. [Online] Available at:

http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6061177&isnumber=6061109 (Accessed: 19 April 2012)

Newcomb S. Note on the frequency of use of the different digits in natural numbers. American Journal of Mathematics 1881;4(1):39–40.

NewDawn, (2011), Global JXDM & NIEM moving towards seamless information Sharing, http://www.newdawntech.com/CJIS/WP_GJXDM_NIEM.pdf, Accessed 21 June 1 2011.

NIEM – National Information Exchange Model, http://www.niem.gov/

NIJ Support of NCFS Research and Activities (2005-2009) Digital Evidence Markup Language: An Object-Oriented, XML-based Model for Sharing Computer Crime-related Information. Award Number: 2005-MU-MU-K044 2005-MU-MU-K044 Supplement 1

Papadogiannakis, A., Antoniades, D., Polychronakis, M., & Markatos, E. P. (2007, October). Improving the performance of passive network monitoring applications using locality buffering. In Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 2007. MASCOTS'07. 15th International Symposium on (pp. 151-157). IEEE. Available at: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4674410 (Accessed: 26 May 2013)

Papadogiannakis, A., Polychronakis, M., & Markatos, E. P. (2010, April). Improving the accuracy of network intrusion detection systems under load using selective packet discarding. In Proceedings of the Third European Workshop on System Security (pp. 15-21). ACM. Available at: http://dl.acm.org/citation.cfm?id=1752049 (Accessed: 26 May 2013)

Pérez-González F, Heileman GL, Abdallah CT. Benford's law in image processing. In: Proc. IEEE International Conference on Image Processing, (ICIP 2007); 2007. p. 405–8.

Rouse, M. (2007) Intrusion Detection (ID). Available at: http://searchmidmarketsecurity.techtarget.com/definition/intrusion-detection (Accessed: 20 June 2012)

Santry, D.J., Feeley, M.J., Hutchinson, N.C. (1999) 'Elephant: The File System that Never Forgets' Seventh Workshop on Hot Topics in Operating Systems, Rio Rico, AZ, 29-30 March. Publisher Unknown. [Online] Available at: http://ieeexplore.ieee.org.ergo.glam.ac.uk/stamp/stamp.jsp?tp=&arnumber=798369 (Accessed: 23 May 2012)

Schaefer G, Stich M. UCID – an uncompressed colour image database. In: Yeung M, editor. Proc. conference on storage and retrieval methods and applications for multimedia. In: Lienhart R, Li C, editors. Proceedings of the Society of Photo-Optical Instrumentation Engineers (SPIE), Vol. 5307; 2004. p. 472–80.

Schäfer C, Schräpler JP, Müller KR, Wagner GG. Automatic identification of faked and fraudulent interviews in surveys by two different methods. DIW-Diskussionspapiere 441. Deutsches Institut für Wirtschaftsforschung (DIW); 2004.

Shields, C. (2010) 'Towards Proactive Forensic Evidentiary Collection', *Proceedings of the 43rd Hawaii International Conference on System Sciences*, Honolulu, 5-8 Jan. IEEE Computer Society. [Online] Available at: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5428489&isnumber=5428274 (Accessed: 19 April 2012)

Shileds, C., Frieder, O., Maloof, M. (2011) 'A system for the proactive, continuous, and efficient collection of evidence', *Digital Investigation*, 8(Supplement), pp.s3-s13. *ScienceDirect* [Online]. Available at: http://www.sciencedirect.com.ergo.glam.ac.uk/ (Accessed: 19 April 2012)

Snort – http://www.snort.org

Solanki K, Sarkar A, Manjunath BS. YASS: yet another steganographic scheme that resists blind steganalysis. In: Furon T, editor. Information hiding. Lecture Notes in Computer Science, Vol. 4567; 2007. p. 16–31.

Sylve, J., Case, A., Marziale, L., and Golden G. R. Acquisition and analysis of volatile memory from android devices. Digital Investigation, 8(34): 175–184, 2012.

Turner, P., Unification of evidence from disparate sources (digital evidence bags), Digital Forensic Research Workshop (DFRWS), 2005

Vasiliadis, G., Antonatos, S., Polychronakis, M., Markatos, E. P., & Ioannidis, S. (2008, January). Gnort: High performance network intrusion detection using graphics processors. In Recent Advances in Intrusion Detection (pp. 116-134). Springer Berlin Heidelberg.3.6.4 Available at: http://link.springer.com/chapter/10.1007/978-3-540-87403-4_7 (Accessed: 26 May 2013)

Vasiliadis, G., Polychronakis, M., Antonatos, S., Markatos, E. P., & Ioannidis, S. (2009, January). Regular expression matching on graphics hardware for intrusion detection. In Recent Advances in Intrusion Detection (pp. 265-283). Springer Berlin Heidelberg. Available at: http://link.springer.com/chapter/10.1007/978-3-642-04342-0_14 (Accessed: 26 May 2013)

Vasiliadis, G., Polychronakis, M., & Ioannidis, S. (2011, October). MIDeA: a multi-parallel intrusion detection architecture. In Proceedings of the 18th ACM conference on Computer and communications security (pp. 297-308). ACM. Available at: http://dl.acm.org/citation.cfm?id=2046741 (Accessed: 26 May 2013)

Vasiliadis, G., Polychronakis, M., & Ioannidis, S. (2011, November). Parallelization and characterization of pattern matching using GPUs. In Workload Characterization (IISWC), 2011 IEEE International Symposium on (pp. 216-225). IEEE. Available at: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=6114181 (Accessed: 26 May 2013)

Vidas, T., Zhang, C., and Christin, N. (2011) Toward a general collection methodology for android devices. Digital Investigation, 8(1): S14–S24, 2011. 11th Annual DFRWS Conference, New Orleans, LA, Aug 01-03,.

Wallace G. The JPEG still picture compression standard. IEEE Transactions on Consumer Electronics 1992;38(1):R18–34.

WebWire (2006) *Paraben Corporation Provides Proactive Forensic Solutions*. Available at: http://www.webwire.com/ViewPressRel.asp?aId=8269 (Accessed: 13 June 2012)

Westfeld A, Pfitzmann A. Attacks on steganographic systems – breaking the steganographic utilities EzStego, Jsteg, Steganos, and S-Tools-and some lessons learned. In: Pfitzmann A, editor. Proc. 3rd international workshop on Information Hiding (IH 99). Lecture Notes in Computer Science, Vol. 1768. Springer Berlin/Heidelberg; 2000. p. 61–76.

Williams, P.A.H. (2008) 'In a 'trusting' environment, everyone is responsible for information security', *Information Security Technical Report*, 13(4), pp. 207-215. ScienceDirect [Online]. Available at: http://www.sciencedirect.com.ergo.glam.ac.uk/ (Accessed: 23 May 2012)

Yusoff, Y., Ismail, R., Hassan, Z. (2011) 'Common Phases of Computer Forensics Investigation Models', *International Journal of Computer Science & Information Technology (IJCSIT)*, 3(3). Academy & Industry Research Collaboration Center [Online]. Available at: http://airccse.org/journal/jcsit/0611csit02.pdf (Accessed: 23 May 2012)

Zaharis A, Martini A, Tryfonas T, Illioudis C, Pangalos G. Lightweight steganalysis based on image reconstruction and lead digit distribution analysis. International Journal of Digital Crime and Forensics 2011;3(4):29–41.

Zong H, Liu FL, Luo XY. Blind image steganalysis based on wavelet coefficient correlation. Digital Investigation 2012;9(1):58–68.

Appendix 1 – Interview with the Avon and Somerset Constabulary, UK

On Friday 9 March 2012, University of Bristol ForToo members met with Jon Niccolls, Network Investigator at Avon and Somerset Constabulary. During this comprehensive interview, which took place at the Avon and Somerset Constabulary offices in Bristol, Mr Niccolls and the team talked extensively about the following:

- The force's current forensic capability and procedures.
- Examples of typical investigation scenarios, methodology and common problems.
- User requirements for a network forensic toolset.

3.1 Current Capability

Constabulary officers and forensic analysts currently have at their disposal commercial software for data triage and analysis from hardware.

To analyse personal computers and notebooks, analysts use the EnCase[1] software suite. According to the company's website as well as the interviewee, the software generates an exact binary duplicate of the data on the disk and memory of the equipment under investigation and generates MD5 hash values for files, in order to reveal attempts to tamper with evidence. It can also recover deleted files and disk partitions, reconstruct HTML pages and instant messaging chat sessions, detect file deletion from log files and other events which might be important for a forensic investigation. The developing company states that the most recent version of the toolset can also analyse data from tablets and smartphones.

Investigators also have at their disposal a version of Micro Systemation's XRY[2]. This software is designed to retrieve data from a very large host of mobile handsets, smartphones and tablet PCs.

When it comes to the analysis of server log files, investigators do not have any automated tools. Searching through logs to detect suspicious or interesting events is done manually. When the investigation involves log files from different hosts, temporal correlation between events is also done manually, in order to counter potential clock skews and offsets between the hosts. Lack of automation means that investigations of this nature are very time consuming.

3.2 Desirable Capability

As discussed in the previous section, lack of automated log file analysis is causing delays to investigations and evidence collection. During the interview, the following desirable characteristics were identified for automated log file analyzers:

[1] http://www.guidancesoftware.com/

[2] http://www.msab.com/

- **Temporal analysis and event correlation**: The software, when provided with multiple log files as input, should be able to process them and highlight event correlations across different hosts. The raw output would be a set of relations between events. For instance, "At time T_0, Host H_1 initiated an SSH connection to Host H_2. The connection was accepted by H_2 at time T_1".
- **Automated detection of events of interest**: Currently, the investigation often involves manual browsing through server log files in an attempt to find events indicative of common attacks (e.g. SQL injection artifacts and cross-site scripts). An automated analysis tool would make it possible to conduct this search unattended. Extensible design could also describe new attacks with a descriptive language or a graphical interface, without having to adjust the tool's implementation. Additionally, it should be possible to detect and highlight traffic anomalies in an automated fashion, such as a sudden spike of incoming connection attempts which could indicate the start of a Denial of Service attack. Algorithms to achieve that can potentially be adopted from the state-of-the-art advances in the field of intrusion detection and prevention systems.
- **Visualization**: The outputs of the event correlation above can then be used to feed a visualization tool, which will highlight events of interest to the investigator. It will also make it possible for the analyst to navigate between events that occurred on different hosts.
- **Forensic Standards**: The tool must comply with current standards and good practices around forensic investigations. Specifically, it must keep digital evidence forensically sound and comply with all rules governing seizure and preservation of evidence. Good analytical skills need to come from the investigators themselves.

Appendix 2.A

2.A.1 Example of Snort through syslog

Nov 22 19:52:38 192.168.204.21 snort: [1:0:1] TCP Connect Scan {TCP} 66.90.69.150:2734 -> 192.168.3.25:8080

2.A.2 Example of XML insert of A.1 alert

```
<RELATIONS command="INSERT" datatype="Cassandra">
 <REL name="event">
  <ATT name="timestmp" type="timestamp">2010-11-22 19:52:38</ATT>
  <REL name="data">
   <ATT name="data_text" type="text">54435020436f6e6e656374205363616e207b5443507d</ATT>
   <REL name="encoding">
    <ATT name="encoding_type" type="varchar(8)">HEX</ATT>
   </REL>
  </REL>
  <REL name="event_type">
   <ATT name="event_type_name" type="text">snort</ATT>
  </REL>
  <REL name="observer">
   <ATT name="obsrv_name" type="text">observer-192.168.204.21</ATT>
   <REL name="agent">
    <ATT name="agent_name" type="text">agent-192.168.204.21</ATT>
    <REL name="computer">
     <ATT name="hostname" type="text">192.168.204.21</ATT>
     <ATT name="ip" type="inet">192.168.204.21</ATT>
     <ATT name="os" type="text">NULL</ATT>
     <ATT name="mac" type="macaddr">00:00:00:00:00:00</ATT>
     <ATT name="domain" type="text">unknown</ATT>
     <REL name="comp_type">
      <ATT name="comp_type_name" type="text">Not-Known</ATT>
     </REL>
    </REL>
    <REL name="process">
     <ATT name="prcss_name" type="text">snort</ATT>
```

```
      <ATT name="prcss_pid" type="bigint">0</ATT>
      <REL name="prcss_type">
       <ATT name="prcss_type_name" type="text">snort</ATT>
      </REL>
     </REL>
    </REL>
   </REL>
   <REL name="reporter">
    <ATT name="rprt_name" type="text">reporter-syslog-server</ATT>
    <REL name="agent">
     <ATT name="agent_name" type="text">agent-syslog-server</ATT>
     <REL name="computer">
      <ATT name="hostname" type="text">syslog-server</ATT>
      <ATT name="ip" type="inet">127.0.0.1</ATT>
      <ATT name="os" type="text">NULL</ATT>
      <ATT name="mac" type="macaddr">00:00:00:00:00:00</ATT>
      <ATT name="domain" type="text">unknown</ATT>
      <REL name="comp_type">
       <ATT name="comp_type_name" type="text">Not-Known</ATT>
      </REL>
     </REL>
     <REL name="process">
      <ATT name="prcss_name" type="text">syslog</ATT>
      <ATT name="prcss_pid" type="bigint">0</ATT>
      <REL name="prcss_type">
       <ATT name="prcss_type_name" type="text">syslog</ATT>
      </REL>
     </REL>
    </REL>
   </REL>
   <REL name="source">
    <ATT name="src_name" type="text">source-66.90.69.150</ATT>
    <REL name="agent">
     <ATT name="agent_name" type="text">agent-66.90.69.150</ATT>
     <REL name="computer">
      <ATT name="hostname" type="text">PTR</ATT>
      <ATT name="ip" type="inet">66.90.69.150</ATT>
      <ATT name="os" type="text">NULL</ATT>
      <ATT name="mac" type="macaddr">00:00:00:00:00:00</ATT>
      <ATT name="domain" type="text">unknown</ATT>
      <REL name="comp_type">
       <ATT name="comp_type_name" type="text">Not-Known</ATT>
```

```
      </REL>
     </REL>
     <REL name="process">
      <ATT name="prcss_name" type="text">unknown</ATT>
      <ATT name="prcss_pid" type="bigint">0</ATT>
      <REL name="prcss_type">
       <ATT name="prcss_type_name" type="text">unknown</ATT>
      </REL>
     </REL>
    </REL>
   </REL>
   <REL name="destination">
    <ATT name="dstn_name" type="text">destination-192.168.3.25</ATT>
    <REL name="agent">
     <ATT name="agent_name" type="text">agent-66.90.69.150</ATT>
     <REL name="computer">
      <ATT name="hostname" type="text">192.168.3.25</ATT>
      <ATT name="ip" type="inet">192.168.3.25</ATT>
      <ATT name="os" type="text">NULL</ATT>
      <ATT name="mac" type="macaddr">00:00:00:00:00:00</ATT>
      <ATT name="domain" type="text">unknown</ATT>
      <REL name="comp_type">
       <ATT name="comp_type_name" type="text">Not-Known</ATT>
      </REL>
     </REL>
     <REL name="process">
      <ATT name="prcss_name" type="text">unknown</ATT>
      <ATT name="prcss_pid" type="bigint">0</ATT>
      <REL name="prcss_type">
       <ATT name="prcss_type_name" type="text">unknown</ATT>
      </REL>
     </REL>
    </REL>
   </REL>
  </REL>
</RELATIONS>
```

Appendix 2.B

2.B.1 Value of a Name in a Key

2.B.1.1 XSM Query

```
<RELATIONS command="SELECT">
<REL name="max_event"/>
</RELATIONS>
```

2.B.1.2 XSM Response

```
<RELATIONS command="SELECT_RESULTS">
<REL name="RESULTS_ID" value="1">
        <ATT name="max_event">1285586792.80569910000522330684 07</ATT>
        <REL name="TOTAL_RECORDS">1</REL>
</REL>
<REL name="TOTAL_RESULTS">1</REL>
</RELATIONS>
```

Appendix 2.C

2.C.1 Get Newest Keys

2.C.1.1 XSM Query

```
<RELATIONS command="SELECT">
        <REL name="event" limit="5" >
        </REL>
</RELATIONS>
```

2.C.1.2 XSM Response

```
<RELATIONS command="SELECT_RESULTS">
 <REL name="RESULTS_ID" value="1">
  <REL name="event">
   <ATT name="event_id">1317746544.1657851000052239042154</ATT>
  </REL>
  <REL name="TOTAL_RECORDS">2</REL>
 </REL>
 <REL name="RESULTS_ID" value="2">
  <REL name="event">
   <ATT name="event_id">1326370273.2976201000052239042154</ATT>
  </REL>
  <REL name="TOTAL_RECORDS">2</REL>
 </REL>
 <REL name="RESULTS_ID" value="3">
  <REL name="event">
   <ATT name="event_id">1326434876.6067240000052239042154</ATT>
  </REL>
  <REL name="TOTAL_RECORDS">2</REL>
 </REL>
 <REL name="RESULTS_ID" value="4">
  <REL name="event">
   <ATT name="event_id">1326468643.2295351000052239042154</ATT>
  </REL>
  <REL name="TOTAL_RECORDS">2</REL>
 </REL>
 <REL name="RESULTS_ID" value="5">
  <REL name="event">
   <ATT name="event_id">1329227240.5903831000052239042154</ATT>
```

```
  </REL>
  <REL name="TOTAL_RECORDS">2</REL>
 </REL>
 <REL name="TOTAL_RESULTS">5</REL>
</RELATIONS>
```

Appendix 2.D

2.D.1 List of Keys by Time Range

2.D.1.1 XSM Query

```
<RELATIONS command="SELECT">
  <REL name="event">
        <ATT name="timestmp" op="gtq">2010-09-27 11:26:00</ATT>
        <ATT name="timestmp" op="ltq">2010-09-27 11:27:00</ATT>
    </REL>
</RELATIONS>
```

2.D.1.2 XSM Response

```
<RELATIONS command="SELECT_RESULTS">
    <REL name="RESULTS_ID" value="1">
        <REL name="event">
            <ATT name="event_id">1285586761.4762199000052233068407</ATT>
            <ATT name="timestmp">2010-09-27 11:26:01</ATT>
        </REL>
        <REL name="TOTAL_RECORDS">2</REL>
    </REL>
    <REL name="RESULTS_ID" value="2">
        <REL name="event">
            <ATT name="event_id">1285586765.4343281000052233068407</ATT>
            <ATT name="timestmp">2010-09-27 11:26:05</ATT>
        </REL>
        <REL name="TOTAL_RECORDS">2</REL>
    </REL>
    <REL name="RESULTS_ID" value="3">
        <REL name="event">
            <ATT name="event_id">1285586770.4119999000052233068407</ATT>
            <ATT name="timestmp">2010-09-27 11:26:10</ATT>
        </REL>
        <REL name="TOTAL_RECORDS">2</REL>
    </REL>
    <REL name="RESULTS_ID" value="4">
        <REL name="event">
            <ATT name="event_id">1285586792.8056991000052233068407</ATT>
            <ATT name="timestmp">2010-09-27 11:26:32</ATT>
```

```
            </REL>
            <REL name="TOTAL_RECORDS">2</REL>
        </REL>
        <REL name="TOTAL_RESULTS">4</REL>
</RELATIONS>
```

Appendix 2.E

2.E.1 List of Keys by Data

2.E.1.1 XSM Query

```
<RELATIONS command="SELECT">
  <REL name="event">
        <REL name="data">
        <ATT name="data_text" type="text">54435020436f6e6e656374205363616e207b5443507d</ATT>
      </REL>
  </REL>
</RELATIONS>
```

2.E.1.2 XSM Response

```
<RELATIONS command="SELECT_RESULTS">
    <REL name="RESULTS_ID" value="1">
        <REL name="event">
            <ATT name="event_id">1285586761.4762199000052233068407</ATT>
            <REL name="data">
            <ATT name="data_text" type="text">54435020436f6e6e656374205363616e207b5443507d</ATT>
            </REL>

        </REL>
        <REL name="TOTAL_RECORDS">2</REL>
    </REL>
    <REL name="RESULTS_ID" value="2">
        <REL name="event">
            <ATT name="event_id">1285586765.4343281000052233068407</ATT>
            <REL name="data">
            <ATT name="data_text" type="text">54435020436f6e6e656374205363616e207b5443507d</ATT>
            </REL>

        </REL>
        <REL name="TOTAL_RECORDS">2</REL>
    </REL>
    <REL name="TOTAL_RESULTS">2</REL>
</RELATIONS>
```

Appendix 2.F

2.F.1 Query on event_id

2.F.1.1 XSM Query

```
<RELATIONS command="SELECT">
    <REL name="event" >
   <ATT name="event_id">1326370273.2976201000052239042154</ATT>
    </REL>
</RELATIONS>
```

2.F.1.2 XSM Response

```
<RELATIONS command="SELECT_RESULTS">
 <REL name="RESULTS_ID" value="1">
  <REL name="event">
   <ATT name="event_id">1326370273.2976201000052239042154</ATT>
   <REL name="data">
    <ATT
name="data_text">4d495343204d53205465726d696e616c207365727665722072657175657374205b436c617373696669636174
696f6e3a2047656e657269632050726f746f636f6c20436f6d6d616e64204465636f64655d205b5072696f726974793a20335d3a2
07b5443507d</ATT>
    <REL name="encoding">
     <ATT name="encoding_type">HEX</ATT>
    </REL>
   </REL>
   <REL name="destination">
    <REL name="agent">
     <ATT name="agent_name">agent-193.63.130.136</ATT>
     <REL name="computer">
      <REL name="comp_type">
       <ATT name="comp_type_name">Not-Known</ATT>
      </REL>
      <ATT name="domain">unknown</ATT>
      <ATT name="hostname">j4-be08.fat.glam.ac.uk.</ATT>
      <ATT name="ip">193.63.130.136</ATT>
      <ATT name="mac">00:00:00:00:00:00</ATT>
      <ATT name="os">NULL</ATT>
     </REL>
     <REL name="process">
      <ATT name="prcss_name">unknown</ATT>
      <ATT name="prcss_pid">0</ATT>
```

```
    <REL name="prcss_type">
     <ATT name="prcss_type_name">unknown</ATT>
    </REL>
   </REL>
  </REL>
  <ATT name="dstn_name">destination-193.63.130.136</ATT>
 </REL>
 <REL name="event_type">
  <ATT name="event_type_name">snort</ATT>
 </REL>
 <REL name="observer">
  <REL name="agent">
   <ATT name="agent_name">agent-J4-BE08</ATT>
   <REL name="computer">
    <REL name="comp_type">
     <ATT name="comp_type_name">Not-Known</ATT>
    </REL>
    <ATT name="domain">unknown</ATT>
    <ATT name="hostname">J4-BE08</ATT>
    <ATT name="ip">193.63.130.136</ATT>
    <ATT name="mac">00:00:00:00:00:00</ATT>
    <ATT name="os">NULL</ATT>
   </REL>
   <REL name="process">
    <ATT name="prcss_name">snort</ATT>
    <ATT name="prcss_pid">0</ATT>
    <REL name="prcss_type">
     <ATT name="prcss_type_name">snort</ATT>
    </REL>
   </REL>
  </REL>
  <ATT name="obsrv_name">observer-J4-BE08</ATT>
 </REL>
 <REL name="reporter">
  <REL name="agent">
   <ATT name="agent_name">syslog-server-agent</ATT>
   <REL name="computer">
    <REL name="comp_type">
     <ATT name="comp_type_name">Not-Known</ATT>
    </REL>
    <ATT name="domain">unknown</ATT>
    <ATT name="hostname">syslog-server</ATT>
    <ATT name="ip">193.63.129.199</ATT>
    <ATT name="mac">00:00:00:00:00:00</ATT>
    <ATT name="os">NULL</ATT>
```

```
      </REL>
      <REL name="process">
       <ATT name="prcss_name">syslog</ATT>
       <ATT name="prcss_pid">0</ATT>
       <REL name="prcss_type">
        <ATT name="prcss_type_name">syslog</ATT>
       </REL>
      </REL>
     </REL>
     <ATT name="rprt_name">reporter-syslog-server</ATT>
    </REL>
    <REL name="source">
     <REL name="agent">
      <ATT name="agent_name">agent-212.7.7.206</ATT>
      <REL name="computer">
       <REL name="comp_type">
        <ATT name="comp_type_name">Not-Known</ATT>
       </REL>
       <ATT name="domain">unknown</ATT>
       <ATT name="hostname">212.7.7.206</ATT>
       <ATT name="ip">212.7.7.206</ATT>
       <ATT name="mac">00:00:00:00:00:00</ATT>
       <ATT name="os">NULL</ATT>
      </REL>
      <REL name="process">
       <ATT name="prcss_name">unknown</ATT>
       <ATT name="prcss_pid">0</ATT>
       <REL name="prcss_type">
        <ATT name="prcss_type_name">unknown</ATT>
       </REL>
      </REL>
     </REL>
     <ATT name="src_name">source-212.7.7.206</ATT>
    </REL>
    <ATT name="timestmp">2011-05-04 18:33:45</ATT>
   </REL>
   <REL name="TOTAL_RECORDS">28</REL>
  </REL>
  <REL name="TOTAL_RESULTS">1</REL>
</RELATIONS>
```

Appendix 2.G

2.G.1 Value of a Name in a Key

2.G.1.1 FDE Query

```
<function>
        <name>source_computer_ip</name>
        <version>1.7</version>
        <eventid>1285586792.8056991000052233068407</eventid>
</function>
```

2.G.1.2 FDE Response

```
<result>
        <relation eventid="1285586792.8056991000052233068407" version="1.7" name="source_computer_ip">
                <row>
                        <data name="ip">69.140.129.95</data>
                </row>
        </relation>
</result>
```

2.G.2 max event

2.G.2.1 FDE Query

```
<function>
        <name>max_event</name>
        <version>1.7</version>
        <eventid>0</eventid>
</function>
```

2.G.2.2 FDE Response

```
<result>
        <relation version="1.7" name="max_event">
                <row>
```

```
                <data name="max_event">1285586792.80569910000522330684 07</data>
            </row>
        </relation>
</result>
```

Appendix 3 – Graphical Passwords and the Android Pattern Lock Screen

Very often people tend to use passwords to keep their personal data safe and modern forensic tools and exploits are able to bypass the locking mechanism of the devices. Text-based passwords and PIN codes are normally coupled with bank accounts, computational devices etc. Individuals possess several accounts and numerous passwords that need to remember. Thus, the users often have to balance usability with security. Graphical passwords may come in much more variety compared to text-based solutions. They can include procedures such as clicking some points on an image, drawing a line or a shape. The most important advantage they provide is the possibility to define a password in a way that is memorisable by the user and yet, still hard to guess by the attacker. However, graphical passwords can also have their weaknesses, if we take into consideration the fact that users may select their graphical passwords with respect to some meaningful process. Thus, human psychology can be associated with the choice of a graphical password.

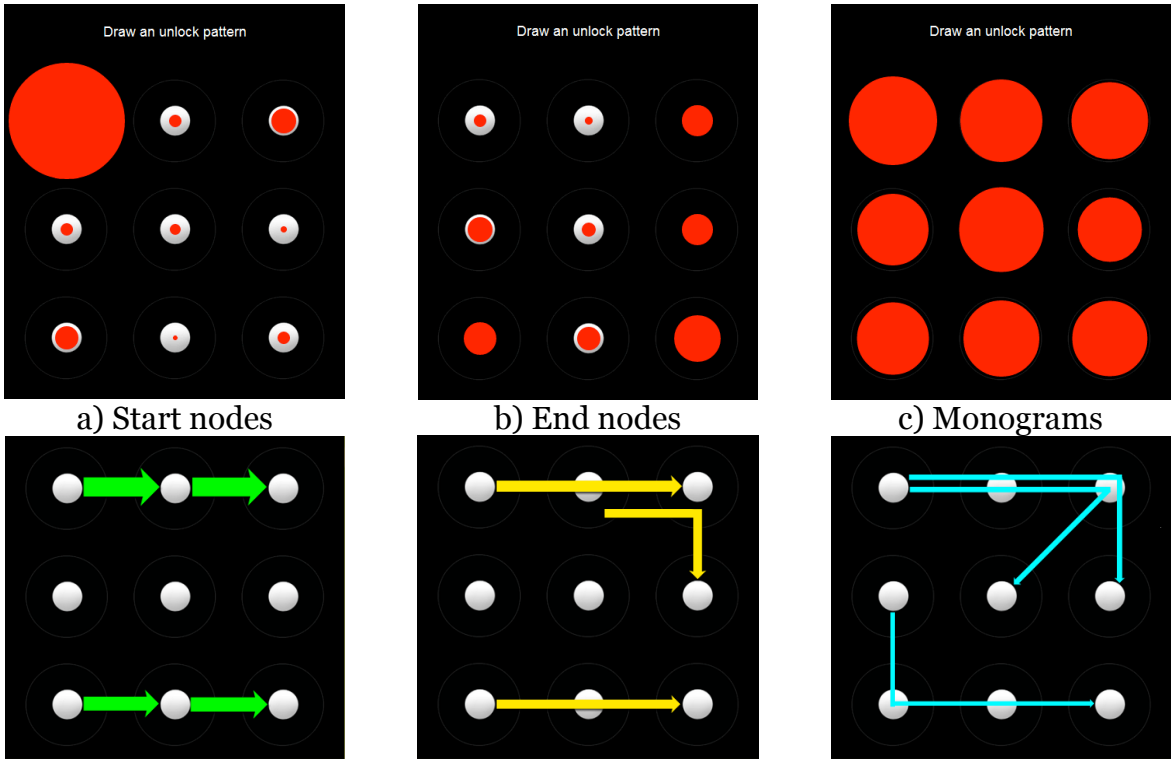
Android's pattern lock mechanism relies on users swiping their finger to unlock the device. This action leaves behind an oily residue or smudges. The first idea that comes to mind for retrieving a lock pattern is to observe the touchscreen for smudges using standard optics. Relevant research on retrieving lock patterns using standard optics is conducted by Aviv et al. (2010). In their work, they demonstrate how recovering smudges using a light source and a digital camera is possible due to the fact that touchscreen surfaces are reflective rather than diffusive. Apart from the ideal photograph capturing angles to retrieve smudges, the experiments focus on various states of a touchscreen. We conducted a pilot study aimed to investigate trends and biases that might exist while users are forming their lock screen patterns (Andriotis et al., 2013). The outcome of the research showed that it is possible to break efficiently the Android pattern lock, using physical attacks in combination with behavioural attacks.

In order to study the effect of psychological or behavioural factors on pattern setting we conducted a web-based survey. This method was chosen because the participant does not need to own a specific smartphone. The results presented in our work (Andriotis et al., 2013) are calculated after filtering the database from irrelevant entries (144 unique participants). The survey started with basic demographics, continued with questions about participants' smartphone experience and their opinion on the notion of device locking and finalised after two pattern entries. The first was a pattern the user thought would be easy to remember and the second was a pattern that the user thought would be a secure password.

Summarising the findings of the survey we deduce that 65.97% were males and 34.03% females. The majority of the users were aged between 18 and 29 inclusive (81.25%) and the next more frequent age bracket was 30-49 (15.28%). This figure was expected because the survey was promoted through social media and through a university mailing list. 79.86% of the participants have owned a smartphone at least once, 92.17% of which still own a smartphone. Among them, 48.11% currently use iOS and 40.57% use Android. Symbian and Blackberry follow with 5.67% and 3.78% respectively. The people who ever owned an Android smartphone had at least one

year of experience with it. 47.22% of the participants with a smartphone use any type of screen lock whereas 52.78% do not. The basic reasons they use a screen lock is to protect personal data and prevent others fiddling with the device. 65.98% of the participants believe that the highest risk that would compromise a lock is shoulder surfing. Smudges left on the screen and cameras in the room have the same rating of being the highest risk with 15.97%, yet the former has been selected more times as the second highest risk compared to the latter, rendering it the second highest risk after the shoulder surfing. Furthermore, 57.64% of the participants thought the secure pattern they entered is usable in everyday life, while 42.36% did not. Finally, 35.42% of the participants thought that the easy pattern they entered was secure enough, while 64.58% did not.

Using entropy calculations we have a clear view of users' behavioural trends when they form their patterns. The next figure shows the most frequent and popular starting and end points and also, the most visited sub-patterns drawn by our sample.



a) Start nodes b) End nodes c) Monograms

d) Bigrams e) Trigrams f) Fourgrams

Fig. 18 - Popular sub-patterns and nodes

To evaluate our proposed attack scheme (combining a physical and a behavioural attack on a pattern) we conducted a new experiment and derived data from a new set of 22 participants, which were not among those who took part on the web-survey. 15 of them (68.2%) were males and 7 (31.8%) females. 86.4% were aged between 21-30 and the age of the rest was 31-40 years old. The participants came from Europe (59.1%), Asia (31.8%) and America (9.1%). The experiment took place at a laboratory. They were asked to think of a secure pattern that they would probably use on their smartphones and then apply it on a real device. We copied and drew their patterns on paper and took photographs of the smartphone screen for further analysis. We marked the drawings with serial numbers before taking the photographs to ensure anonymity. The scenario we investigated assumes light usage of the phone after the

pattern was entered, thus, before the photograph was taken we rubbed the screen gently on a cotton surface. We used an HTC Desire smartphone and a Nikon D40x DSLR camera for this experiment.

The use of camera revealed, either fully or partially, 18/22 (81.81%) patterns. The psychological attack confirmed that 81.81% of the participants started their passwords using the most common start points and half of them ended their patterns at the expected end points. The average direction changes for males were 3.19, for females 2.83 and average pattern length for males was 7 and for females 6.33. We also saw some of the most common bigrams, trigrams and four-grams presented previously (popular bigrams were more frequent: 54.5%). Finally, the combination of the two attacks resulted in full or partial retrieval of 20/22 patterns and 100% of the patterns contained at least one of the reference standards. This statistic shows that combining our web-survey results with well-known physical attacks we can increase the possibility to recover a pattern. Thus, a tool that would be able to estimate the pattern if it uses as an input the traces left on the screen in conjunction to our pilot study would be a reasonable outcome of our on-going research.