

Abstract

Eventhough there are several tools developed by companies in a commercial setting, mobile forensics, especially for smartphone devices, are lacking methods and tools. This project introduces a new method for acquiring and examining forensic-grade evidence from a smartphone, regarding bluetooth and Wi-Fi connections. This method is an outcome of a thoroughly investigation of four scenarios that were determined for the needs of this project. Furthermore, this method is evaluated with the use of guidelines for good practice for computer-based, electronic evidence.

A smartphone running Android Operating System (OS) and specifically Samsung Galaxy Europa was used for the implementation of the four scenarios. The investigation of the scenarios was done in order to identify evidences found on the smartphone device which can be seen as potential evidence that a court can review. After the investigations of the scenarios, an analysis of the results took place.

This project is considered to be an investigatory project (type II), since it is motivated by a real world issue and deals with a practical use of various tools and techniques, in order to create a methodical and open way of acquiring and investigating forensic-grade evidence from a smartphone device which runs Android OS.

A research on the guidelines for forensic investigations and tools and methods which other investigators have been using was a necessity in order to understand the state of the art technology of the area. After this research was done, it was vital to determine methods and techniques for acquiring files and folders, which may have evidences stored inside them, and how an investigator can locate these evidences on a smartphone device which runs Android OS.

Apart from the literature review which introduced the term smartphone and the Android Operating System, determined the term of mobile forensics, examined various forensic tools that were used by other forensic investigators and investigated various guidelines for forensic investigations this project involved:

1. A research on how to acquire a physical image from an Android smartphone device
2. An investigation of the Android's logging system
3. A determination and a thoroughly examination of four scenarios which helped to determine evidences related to bluetooth and Wi-Fi connections
4. An analysis of the evidences that were found
5. A determination of a methodology for acquiring evidences from an Android smartphone device regarding its bluetooth and Wi-Fi connection
6. An evaluation of the above methodology with the use of guidelines for good forensic practice for computer-based electronic evidence

Acknowledgements

First and foremost I offer my sincerest gratitude to my supervisor, Dr Theodore Tryfonas, who has supported me throughout my thesis with his valuable assistance and determinant advise.

Thanks also to Christina Sarri for her help, feedback and support during this project.

I would like to greatly thank my family for their constant support and encouragement.

Finally, I thank Seyton Bradford for providing me with the Samsung Galaxy Europa which was used in this project and for his valuable help.

Table of Contents

Declaration.....	
Abstract.....	
Acknowledgements.....	
Table of Contents.....	
1. Introduction.....	1
1.1 Aims and objectives.....	1
2. Background.....	2
2.1 Smartphones.....	2
2.2 Introduction to Android.....	4
2.3 Digital Forensics.....	9
2.3.1 Introduction to Digital Forensics.....	9
2.3.2 Tools for mobile forensics.....	11
2.3.3 Guidelines of good forensic practice.....	15
2.4 Related work.....	18
3. Project's context and its relation with mobile forensics.....	24
3.1 Samsung Galaxy Europa	24
3.2 Android developers tools.....	24
3.3 Definition of the scenarios.....	26
3.4 Project's relation with background.....	28
4. Examination and analysis of the scenarios.....	29
4.1 Research prior to the examination.....	29
4.1.1 ADB Tool.....	29
4.1.2 Partitions of Samsung Galaxy Europa.....	31
4.1.3 Mounting image files with dd command.....	32
4.1.4 Logcat.....	33
4.2 Scenario 1.....	34
4.2.1 Implementation.....	34
4.2.2 Investigation and Results.....	36
4.3 Scenario 2.....	39
4.3.1 Implementation.....	39
4.3.2 Investigation and Results.....	39
4.4 Scenario 3.....	40

4.4.1 Implementation.....	40
4.4.2 Investigation and Results.....	41
4.5 Scenario 4.....	44
4.5.1 Implementation.....	44
4.5.2 Investigation and Results.....	44
4.6 Analysis of the results.....	46
5. Methodology for acquiring evidences regarding bluetooth and Wi-Fi connections.....	50
5.1 Definition.....	50
5.2 Evaluation of the methodology.....	53
6. Conclusion.....	58
6.1 Critical Evaluation.....	58
6.2 Future work.....	59
References.....	60

1. Introduction

1.1 Aims and objectives

This project has two main aims. Firstly, it aims to investigate and examine what information regarding bluetooth and Wi-Fi connections is stored on a smartphone device which runs Android operating system (OS). Secondly, this project aims to investigate how can this kind of information be extracted from the Android smartphone during a forensic examination, under the principles of good forensic practice, Association of Chief Police Officers' (ACPO) guidelines, assess and document the intervention as well as the implications from the smartphone's operation. The two research questions which will guide this research are:

1. What information regarding Wi-Fi and bluetooth connections is stored on an Android smartphone device?
2. How can this kind of information be obtained from the smartphone, under the principles of ACPO guidelines for good forensic practice, during a forensic examination?

A method for investigating and acquiring forensic-grade evidences which are related to bluetooth and Wi-Fi is going to be the outcome of this investigation.

In order to answer the research questions four scenarios will be created, two concerning bluetooth connections and two concerning Wi-Fi connections. It is noteworthy to state that these scenarios will be based on real life cases. The objectives of this study are the following:

1. Make a research on how to obtain a physical image from an Android smartphone device
2. Make a research concerning the logging system of the Android operating system.
3. Implementation of four scenarios
4. Investigation of the four scenarios and recording of all findings
5. Evaluation of findings
6. Establishment of a method for acquiring evidences regarding Wi-Fi and bluetooth connection
7. Evaluation of method with the use of ACPO's guidelines for good forensic practice for computer-based electronic evidence

2. Background

The rapidly development of mobile devices in recent years have seen them outselling personal computers three to one. They are considered to be one of the most disruptive technologies of the last years due to the fact that they are very successful in the everyday life of people from all social classes (Kalba K, 2008). Despite this development, mobile phone forensics are at an early stage when compared to computer forensics (Ahmed R. and Dharaskar RV, 2008). Obviously, this increase and proliferation of mobile phones has noted the urgent need for the creation of new analysis tools and techniques. As stated by Punja S. G. and Mislán, R. (2008) the area of digital forensics has grown out of the practice of computer forensics.

Information on mobile phones is located inside the mobile phone's internal memory but not with the use of a defined standard. Nevertheless, sometimes the cable which provide us the capability to access the internal memory of the mobile phone varies from model to model, company to company etc. Hence, getting the data straight from the mobile phone's memory is more inefficient than personal computers since there are no standard storage or document formats like personal computers (Moore T. 2006).

As Punja S. G. and Mislán, R (2008) advocate, forensics investigators have to cope with different types of mobile phones generation technologies, proprietary embedded firmware systems and with a staggering amount of unique cable connectors for different models of phones within the same manufacturer brand.

2.1 Smartphones

Ahmed R. and Dharaskar RV (2008) note that the sales of smart mobile phone devices which are capable of handling some Personal Digital Assistant (PDA) functions are growing whereas actual PDA devices sales are declining. In addition Distefano A, Me G. and Pace F. (2010) state that there are 2.6 billion subscribers in the world and growing – China Mobile as stated is adding approximately 6 million new subscribers per month. Kalba's K. (2008) study indicated that there are more than three billion subscribers worldwide and that mobile phones have overwhelmed every prior technology such as TV sets, radios, wrist watches, wireline phones, etc.

These numbers, the rapidly growing development and sale rate are the main reasons why mobile phones are gaining this boost in the market and illustrate that they merit attention as a case in global technology diffusion. In addition mobile phones capabilities, including computing power of the devices, hardware etc are increasing leading to various new functions available for mobile devices. Furthermore, because of this increase of computing power and hardware, mobile phones have become portable data carriers (Ahmed R and Dharaskar R, 2008). All these are achieved by simultaneously keeping the mobile devices size small enough, to fit in a pocket. There are various phone models in the market which can be categorized according to their hardware characteristics. Table 1 (Jansen, W. and Ayers R, 2006) divides the phone models in three categories: basic advanced and smart. The characteristics that are used in this classification scheme are illustrative. Characteristics of real mobile phone devices can have features from one or more categories which are defined here. In addition, new pioneering and revolutionary features may be added to smartphones leading advanced features to move from advanced and smart to basic. As noted by Jansen, W. and Ayers R (2006) this classification scheme is considered to be fuzzy and dynamic but it can be referred as a general guide.

As someone can easily understand from Table 1, smartphones are the most well hardware designed mobile phones in the market. This hardware advantage that they have, provides them with advanced functionalities which, as stated by Distefano A, Me G. and Pace F (2010), are ranged from user interface and computational resources, to connectivity and application development.

	<i>Basic</i>	<i>Advanced</i>	<i>Smart</i>
<i>Processor</i>	Limited Speed	Improved Speed	Superior Speed
<i>Memory</i>	Limited Capacity	Improved Capacity	Superior Capacity, Built-in Hard Drive Possibility
<i>Display</i>	Grayscale	Colour	Large Size, 16-bt Color (65,536 colours) or Higher
<i>Card Slots</i>	None	MiniSD or MMCmobile	MiniSDIO or MMCmobile
<i>Camera</i>	None	Still	Still, Video
<i>Text input</i>	Numeric Keypad	Numeric Keypad, Soft keyboard	Touch Screen, Handwriting Recognition, Built- in Qwerty-style keyboard
<i>Cell interface</i>	Voice and Limited Data	Voice and High Speed Data	Voice and Very High Speed Data
<i>Wireless</i>	IrDA	IrDA, Bluetooth	IrDA, Bluetooth, WiFi
<i>Battery</i>	Fixed, Rechargeable Lithium Ion Polymer	Removable, Rechargeable Lithium Ion Polymer	Removable, Rechargeable Lithium Ion

Table 1: Mobile phones categories according to hardware specifications. This table was copied from Jansen, W. and Ayers R, (2006)

Moreover, Table 2 shows the International Data Corporation (2011) findings regarding the worldwide smartphone operating system marketshare in 2011 and the prediction of the market share in 2015. It can be easily seen that Android is the leading smartphone operating system with 39.5% and a difference of 18.6% from the second operating system, Symbian, which holds the 20.9% of the market. iOS, BlackBerry and Windows 7/Windows Phone follows, with marketshare of 15.7%, 14.9% and 5.5% respectively. The predictions of IDC (2011) for the year 2015 shows that not only, Android will increase its share to 45.4% but will also increase the difference from the following operating system from 18.6% to 24.5, which in this case is Windows 7/Windows Phone.

<i>Operating system</i>	<i>2011 Market Share</i>	<i>2015 Market Share</i>
Android	39.5%	45.4%
BlackBerry	14.9%	13.7%
iOS	15.7%	15.3%
Symbian	20.9%	0.2%
Windows 7/Windows Phone	5.5%	20.9%
Others	2.5%	4.6%
Total	100.0%	100.0%

Table 2: Worldwide smartphone operating system marketshare for 2011 and prediction for 2015. This table was copied from IDC (2011).

In addition, Spreitzenbarth M., (2011) illustrates today's top smartphone platforms on a three month

average. Android is the most popular smartphone operating system in the U.S.A with a share of 31.2% followed by Apple's iOS with 24.7% share. In Europe, Symbian is the most popular smartphone operating system with a share reaching 27.0% followed by Android 24.3%.

The leading smartphone operating system, Android, is an open source mobile operating system licensed mainly under Apache Software License 2.0 (Android.com 2010). Moreover, as stated by Spreitzenbarth M., (2011) Android has the biggest growth rate in sector, it is supported by many different manufacturers, it has many different fields of its applications (smartphone, tablet etc) and is considered to be the mobile operating system of the future.

2.2 Introduction to Android

Android is an operating system which is developed by the Open Handset Alliance (OHA). The OHA as determinate on their website is a group of 84 mobile and technology companies who have come together to speed up the innovation in mobile and offer consumers a richer, less expensive, and better mobile experience (OHA, 2011). T-Mobile, Vodafone, HTC, Dell, nVidia, Acer, LG, Samsung, Motorola, Sharp, Sony Ericsson, eBay, Google and many more are members of the OHA.

According to Google Official Blog (2011) there are now 100 million activated Android devices, whereas 400,000 new Android devices are being activated every day. Moreover, 200,000 free and paid applications are available in Android Market and 4.5 billion applications have been installed from the Android Market.

In addition as we can see from Table 3, Android versions codenamed Froyo and Gingerbread are the most widely used platforms of the Android OS, as of September 2011, with a distribution of 82.5% (51.2% for Froyo and 31.3% for Gingerbread). Older versions of Android (Cupcake, Donut and Eclair) have a distribution of 1.0%, 1.8% and 13.3% respectively and the newest version of Android (Honeycomb) has a distribution of 1.4%.

<i>Platform</i>	<i>Codename</i>	<i>Distribution</i>
Android 1.5	Cupcake	1.0%
Android 1.6	Donut	1.8%
Android 2.1	Eclair	13.3%
Android 2.2	Froyo	51.2%
Android 2.3-2.3.2	Gingerbread	0.6%
Android 2.3.3-2.3.4		30.7%
Android 3.0	Honeycomb	0.2%
Android 3.1		0.7%
Android 3.2		0.5%

Table 3: Android devices that have accessed Android Market within a 14-day period ending September 2, 2011. This table was copied from Android Developers (2011).

As it can be concluded by Figure 1, in the future, the distribution of older versions of Android, along with Froyo, will be continually decreasing while newer versions of Android such as Gingerbread is more likely to increase its distribution.

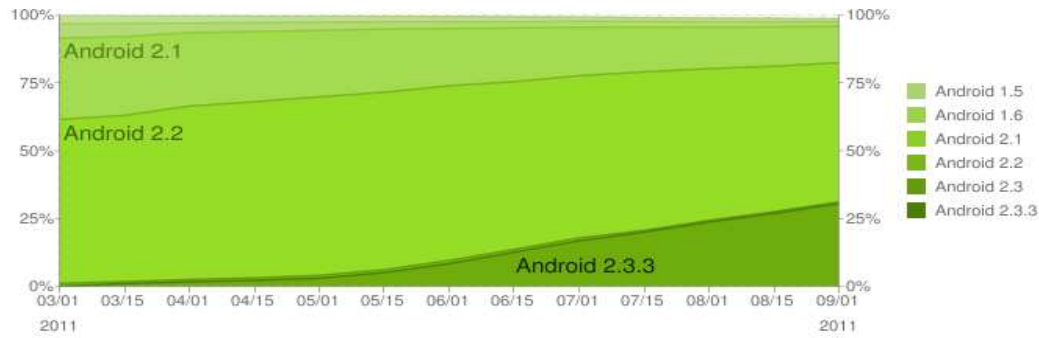


Figure 1: Historical dataset which was collected during two weeks period which ended on September 2, 2011. This figure was copied from Android Developers (2011)

Moreover, Android system architecture has four main levels which are demonstrated in Figure 2 (Xinfang Li et al, 2010).

1. Linux Kernel
2. Library and Android Runtime
3. Application framework
4. Application

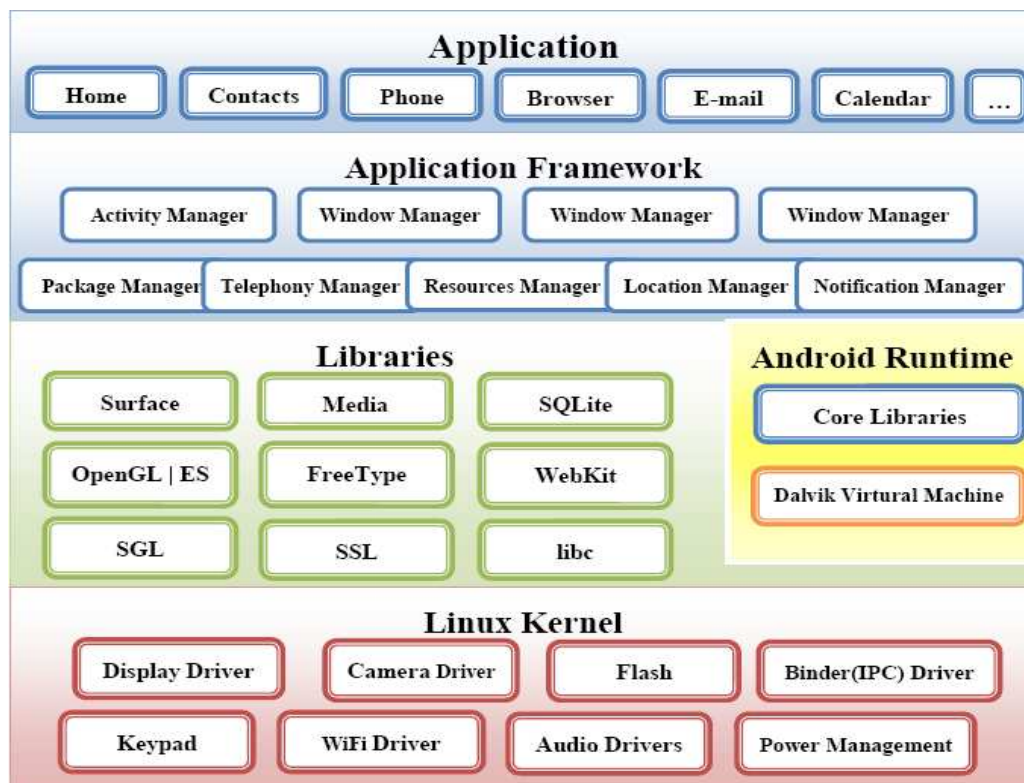


Figure 2: Android system architecture. This figure was copied from Lessard J. and Kessler C G. (2010)

Distefano A, Me G. and Pace F. (2010) provide a more detailed description of each component:

1. Linux Kernel: It is the lowest level of the pyramid and implements the use of Linux 2.6 kernel. It is used in order to support core services of the mobile phone such as memory and

- process management, network stack, drivers and security.
2. Libraries: Various libraries are used by the system through the Android Application Framework such as Standard C System Library, Media Libraries, 3D Libraries.
Android Runtime: This component is constituted by a group of Core Libraries and Dalvik Virtual Machine (DVM). Each application which is running has its own instance of the DVM and executes in its own process.
 3. Application Framework: With the use of this framework which is constituted by a set of exploitable service, Android offers the opportunity to develop Java applications. The developers of such applications can consume and provide services using the Application Programming Interfaces (APIs), with the aim of reusing components but simultaneously complying with the security constraints of the framework. Developers can also have full access to the same framework as APIs used by the core applications.
 4. Applications: A group of some typical applications which are considered as core programs and are provided along with Android, such as e-mail client, text messaging management application, browser, contact management written with the use of Java.

Lessard J. and Kessler C G. (2010) state that the Linux system device defaults to the first physical hard drive or /dev/hd0 and is only capable of understanding character and block devices, like keyboards and disk drives. A use of a Flash Transition layer is needed in order to provide to the mobile phone its functionality, due to the fact that flash memory devices are not character or block devices. The connection between the Linux kernel and the physical flash device is accomplished using a Memory Technology Device.

As Vidas T. , Zhang C. and Christin N (2011) add many, but not all, android smartphone devices use Memory Technology Devices (MTD) which is an abstraction layer for raw devices which permits software to use an interface in order to use various flash technologies. In the occasion where NAND data must be gained from devices that utilise MTD, nand dump can be used. In order to gain this data on devices that are not using MTD other collection techniques must be used. In addition, Vidas T. , Zhang C. and Christin N (2011) add that information is not only stored on the onboard memory but can also be stored on one or more SD cards. Furthermore, user can choose to save some applications and data on the SD card and in some cases manufacturers store the data partition on the SD card.

Vidas T. , Zhang C. and Christin N (2011) illustrate that most android devices have various partitions which are the mostly mapped to MTD devices. The typical partitioning information on an Android device can be found in Table 4. This is not what it is implemented on all Android devices since each manufacturer may choose a different implementation but it is the typical partitioning system. Eight typical partitions are found on a typical Android device as it is demonstrated by Table 4. User data, cache, boot and recovery are some of the partitions. As it can be seen from Table 4, many Android devices use YAFFS2 as their filesystem. YAFFS was developed in 2002, and it was the first file system which was intended to be used along with NAND (Not-AND) flash memory devices. YAFFS2 was developed in 2004 due to the fact that larger size NAND flash devices were created (Lessard J. and Kessler C G. 2010). In addition, YAFFS2 is a fast log-structured file system, which is able to provide built in wear-leveling and error correction and it does not need huge amounts of RAM to run properly (Hoog, A., 2010).

Vidas T. , Zhang C. and Christin N (2011) also state that booting is a bootable image which contains a header, kernel ram disk and an optional second image each page aligned . The SD cards partitions are sometimes marked as internal or external and are recognised by /dev/block/mmcblkXpY where X is the card ID and Y the partition ID on the card and is normally mounted to /sdcard or /mnt/sdcard. They also add that the most interested partitions for a forensic investigator will be the user data and system partitions.

<i>Path</i>	<i>Name</i>	<i>File System</i>	<i>Mount Point</i>	<i>Description</i>
/dev/mtd/mtd0	pds	yaffs2	/config	Configuration data
/dev/mtd/mtd1	misc	-	N/A	Memory Partitioning data
/dev/mtd/mtd2	boot	booting	N/A	Bootable (typical boot)
/dev/mtd/mtd3	recovery	booting	N/A	Bootable (recovery mode)
/dev/mtd/mtd4	system	yaffs2	/system	System files, Applications, Vendor additions, Read- Only, Cache Files
/dev/mtd/mtd5	cache	yaffs2	/cache	
/dev/mtd/mtd6	system data	yaffs2	/data	User data (Applications)
/dev/mtd/mtd7	kpanic	-	N/A	Crash log

Table 4: Typical partition information on an Android device. This table was copied from Vidas T. , Zhang C. and Christin N (2011)

In addition with the typical filesystem (YAFFS2), in 2010 it was announced by Google that Android version 2.3, codenamed Gingerbread, was going to support ext4 filesystem (Android Developers, 2010). Furthermore, in contrast with the most android smartphones Samsung utilises a proprietary file system named Robust FAT File System and OneNAND memory (L. Flash Software Group, Samsung Electronics Co., 2008) . In order to work promptly, Android needs to load some kernel modules and this has an effect that the analysis becomes more difficult as there is no software at the moment that parses RFS related data. In this occasion of Samsung's android smartphones the Linux kernel does not use MTD devices but creates several Sector Translation Layers (STL) and Block Management Layers (BML) block devices (/dev/block/). Table 5 illustrates a typical partition with the use of BML devices.

<i>Device</i>	<i>Name</i>	<i>Mount Point</i>	<i>Description</i>
bml1	boot	-	Primary boot loader
bml2	pit	-	Partition map data
bml3	efs	/efs	Unknown
bml4	SBL	-	Secondary boot loader
bml5	download	-	Download Mode
bml6	param	/mnt/.lfs	Unknown (lfs)
bml7	kernel	N/A	Kernel + initramfs
bml8	recovery	N/A	Kernel + initramfs
bml9	system	/system	Typical /system data (RFS)
bml10	dbdata	/dbdata	Dbcache (RFS)
bml11	cache	/cache	Cacge (RFS)
bml12	modem	-	Modem software

Table 5: Partition information typical of a Samsung's android device. This table was copied from Vidas T. , Zhang C. and Christin N (2011)

According to Lessard J. and Kessler C G. (2010) a forensic examination of an Android based mobile device is mainly focused on the Libraries component of the architecture and specifically at the SQLite databases where the majority of the data is stored. Hoog A. (2011) states that SQLite is a widely used, lightweight database which is available in a single cross-platform file. It is used by many developers due to its lightweight and structured data storage and can be found in most

smartphones like iPhone, Android, Symbian, webOS and in major computer operating systems and applications like Apple's Mac OS X, Google Chrome, Chrome OS and Firefox. Hoog A (2011) also illustrates the five storage data types (sometimes called classes) that can be used with SQLite.

<i>Storage data (classes) types</i>	<i>Description</i>
NULL	The value is a NULL value
INTEGER	The value is a signed integer, stored in 1, 2, 3, 4, 6, or 8 bytes depending on the magnitude of the value.
REAL	The value is a floating point value, stored as an 8-byte IEEE floating point number.
TEXT	The value is a text string, stored using the database encoding (UTF-8, UTF-16BE or UTF-16LE).
BLOB	The value is a blob of data, stored exactly as it was input. Often used to store binary data

Table 6: SQLite storage classes and data types. This table was copied from Hoog A (2011)

Distefano A, Me G. and Pace F. (2010) note that Android relies on the Linux kernel and the functions and management that the Linux kernel provides for the security of the mobile phone. But in order for Android to support the reusing of components and the provisioning of services among various applications some security features are provided using the method of permissions.

- Applications and sandboxes: Android does not give access to any application to operate with aim to alter any other application, the OS or the end-user.
- User IDs and permissions: Android manages installed applications as a different Linux user and at its installation the application is given a unique Linux user ID. All the data owned by one application will have the same ID with it and in order to enable other applications to have access to this data, it is required to enable the access from the Others group of Users.

Security mechanisms of Android may impede forensics investigators from examining the device although there are some basic tools and techniques, which give the opportunity to investigators to recover data from the mobile phone (Lessard J. and Kessler C G. 2010).

Faisal A. et al (2011) state that in order to understand the security system of Android it is crucial to understand the structure of the Android OS. They also add that each application is related with a number of permissions according to its user ID and each one of these permissions is categorized as either normal, dangerous or signature level. Permissions which are considered dangerous are those which will probably have the most impact on the user, meaning that they permit access to private data and other system services like Internet access. These kind of dangerous permissions require the user's acceptance prior to the application installation. Finally, all applications must be signed by a developer's private key which is checked by the Android Package Manager during installation.

As stated before, Android is based on a modified linux kernel and Höebarth S. and Mayrhofer R. (2011) illustrated that each application has its default home directory which is located at /data/data/<package name>/ and contains four directories which can be seen in Table 7.

<i>Directory</i>	<i>Description</i>
/databases	Default location for SQLite databases
/libs	Contains all native libraries of the application, copied during the installation process
/files	Default directory for all files created by the application itself at run time
/shared_prefs	Contains the XML based shared preferences of the application

Table 7: Application default directories. This table was copied from Höebarth S. and Mayrhofer R.

(2011)

Höebarth S. and Mayrhofer R. (2011) continue by stating that pre-installed system applications that are found on the mobile and have higher privileges like the com.android.phone also use the same directories as shown in Table 7. Various manufactures, though, may want to alter these directories to their own needs, for example Samsung uses /dbdata/databases/<package name> for preinstalled applications (Höebarth S. and Mayrhofer R., 2011). Hence, an investigator must know that there are various default locations on android devices for applications to be stored.

2.3 Digital Forensics

2.3.1 Introduction to Digital Forensics

Palmer G. (2001: 16) gives the definition of the digital forensics science as it was given by the Digital Forensics Research Workshop:

“The use of scientifically derived and proven methods toward the preservation, collection, validation, identification, analysis, interpretation, documentation and presentation of digital evidence derived from digital sources for the purpose of facilitating or furthering the reconstruction of events found to be criminal, or helping to anticipate unauthorized actions shown to be disruptive to planned operations.”

Xinfang Li et al, (2010) note that crime issue is considerable when it comes to the exploitation of smartphones technologies and that digital forensics offer methods for acquiring evidences, so that the court can review. But with the rapidly growing development of the technology, digital devices such as computers, mobile phones, digital cameras, storage devices etc have integrated in people's everyday life. Hence, digital forensics are used in the areas of network forensics, mobile forensics, computer forensics etc.

Digital evidence as stated by SWGDE/SWGIT (2011) is a series of binary digit numbers on transmission. Xinfang Li et al. (2010) added that digital evidence can also be information files stored on an electronic device and that there are various formats such as audio, video, images etc. Moreover, digital evidences can be copied without having any differences, can be altered without facing difficulties, hard to be identified from the original resource, can be integrated on data verification and can not be understood directly without technical process.

Handheld device forensics are considered to be an emerging field within the wider area of the computer forensics field but there are some differences between them due to various reasons which are (Ayers R. et al 2005):

1. The mobile nature of the handheld devices (the size of the device, the battery used, different interfaces, media and hardware)
2. The filesystem varies from mobile devices and between mobile devices and computers.
3. The behaviour of such devices when hibernating
4. Various operating system are used
5. The product cycle for handheld devices is shortcomings

Jansen's, W. and Ayers's, R. (2006: ES-1) definition of Mobile phone forensic is:

“Mobile phone forensics is the science of recovering digital evidence from a mobile phone under forensically sound conditions using accepted methods. Mobile phones, especially those with advanced capabilities, are a relatively recent phenomenon, not usually covered in classical computer forensics.”

The field of Mobile Forensics, has been rapidly developing since the start of the 21st century and this is due to the fact that more mobile devices are found at crime scenes whether that are technical, non-technical or violent (Punja S. G. and Mislán, R., 2008). Lessard J. and Kessler C G. (2010) illustrate that mobile phones may have more important information which can be linked to a person per byte examined than most personal computers. But this kind of information is hard to be founded because the technology lacks of a proper forensic method.

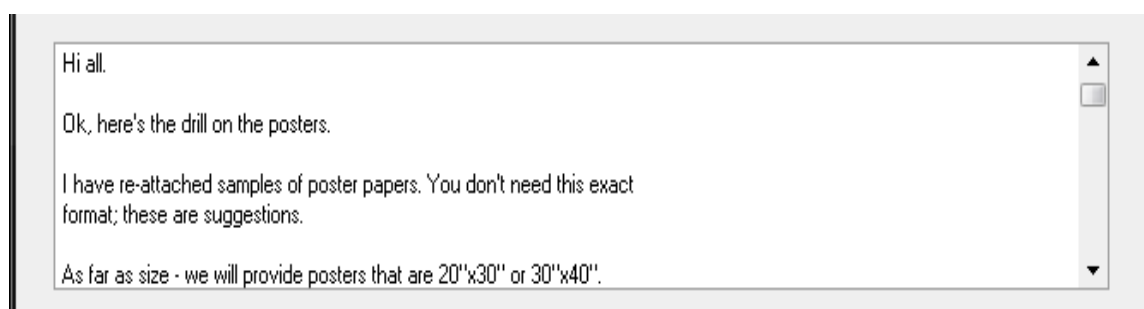
Distefano A, Me G. and Pace F. (2010) noted that the number of mobile devices which are found in crime scenes is growing whereas the ability of performing a forensic analysis on these devices is limited from a technological and a methodological aspect. Moreover, depending on the forensic environment, the amount of personal information which is stored by the mobile device vary, thus the importance of the forensic analysis of the device is depended on the amount of information it holds. Because of their ability to hold large amount of information, smartphones are considered to be the most interesting for forensic purposes (van der Knijff, 2007).

There is an increasing trend for using data which is stored on a mobile phone as evidence in civil or criminal cases (Ahmed R and Dharaskar R, 2008). As stated before, new opportunities for advanced functions that are provided to the holders of mobile phones, may have information which can be used as evidence in criminal cases. Punja S. G. and Mislán, R., (2008) consider smart mobile devices as a treasure trove of helpful information where crime investigators not only acquire call history, contacts and text messages but can also find other sources of evidence such as photos, videos etc. These kind of new evidences either are used to take investigation to the next step or either lead the crime scene investigators to more questions. Xinfang Li et al (2010) state that information security issue of smartphones is getting more important in today's society.

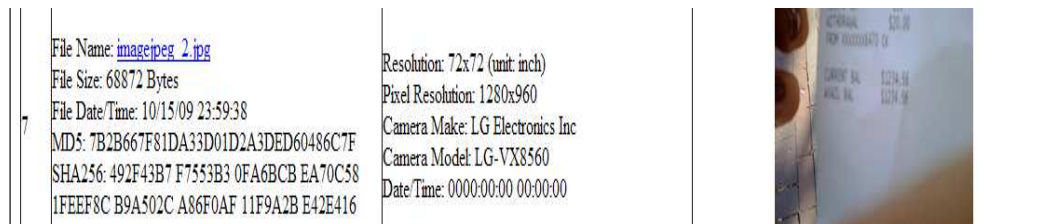
Ayers R. et al (2005) state that there are various types of information that can be acquired from a mobile device using forensic tools such as Personal Information Management (PIM) data such as phone book, logs of phone calls, SMS/EMS/MMS messages, Email, Instant Messaging content, , Audio, Videos, Images, SIM content and Uninterrupted image data. Figure 3 demonstrates some examples of acquired information from an investigation done by Lessard J. and Kessler C G. (2010).

	social_id	user_id	user_row_id	active_row_id	full_name
15	facebook	1328640088	76	1	Jay Weier
14	facebook	11018009	26	1	Andrew Cote
7	facebook	68901928	281	1	Asa Gillette
10	facebook	721811220	357	1	Brittany Mailhotte

a) Facebook status updates.



b) A complete email



c) Picture file extracted by the UFED

Figure 3: Examples of acquired information from mobile devices. These figures were copied from Lessard J. and Kessler C G. (2010)

Although its importance, the field of forensic analysis of smartphones is experiencing a lot of difficulties and problems which need to be solved. Due to the fact that mobile devices can be found in several crime scenes the forensic analysis of them has become important to law enforcement, tracking illegal activities and in providing evidence.

Mislan, R.P., Casey, E., and Kessler, G.C. (2010) state that the dynamic and rapidly evolving nature of mobile device forensics is forcing forensic investigators to acquire data the moment it is observed and available because of the mobile device forensics nature as a source of intelligence or evidence. Hence, mobile devices are considered challenging from a forensic investigator point of view. There is always the need for a quick and easy method – without the help of an expert – to acquire information stored in mobile phones, which is growing daily. A software application or standard method is the best way to acquire this kind of information.

2.3.2 Tools for mobile forensics

Jansen, W. and Ayers, R. (2004:1) define forensic toolkits as:

“Forensic toolkits are intended to facilitate the work of examiners, allowing them to perform the above steps in a timely and structured manner, and improve the quality of the results.”

Moreover, Jansen, W. and Ayers, R. (2006) state that this kind of tools are using logical acquisitions with the help of common protocols for synchronization, debugging and communications. In some occasions like the recovery of deleted data, more specialized tools are needed in order to accomplish the goals of the forensic examination. In addition, this kind of tools are mostly in their early stages of maturity and may have limitations, such as not working on all devices and can not acquire the same amount of evidence.

Ayers R. et al. (2005) claim that although there is a large number of toolkits used for mobile forensics, the majority of them is only operational with a small number of mobile devices because of the various operating systems used and the hardware architecture. Moreover, they continue by stating that there is a number of toolkits which are only able to acquire, examine and report evidence from a subset of evidences of the mobile phone.

They are various ways for connecting the tools with the mobile devices (Figure 4) such as wired connections and through Infrared (IrDA) and Bluetooth connections (although IrDA technology is becoming obsolete in mobile phones). In addition, phone software and/or hardware must be used in order for the examiner to acquire data from the mobile device:

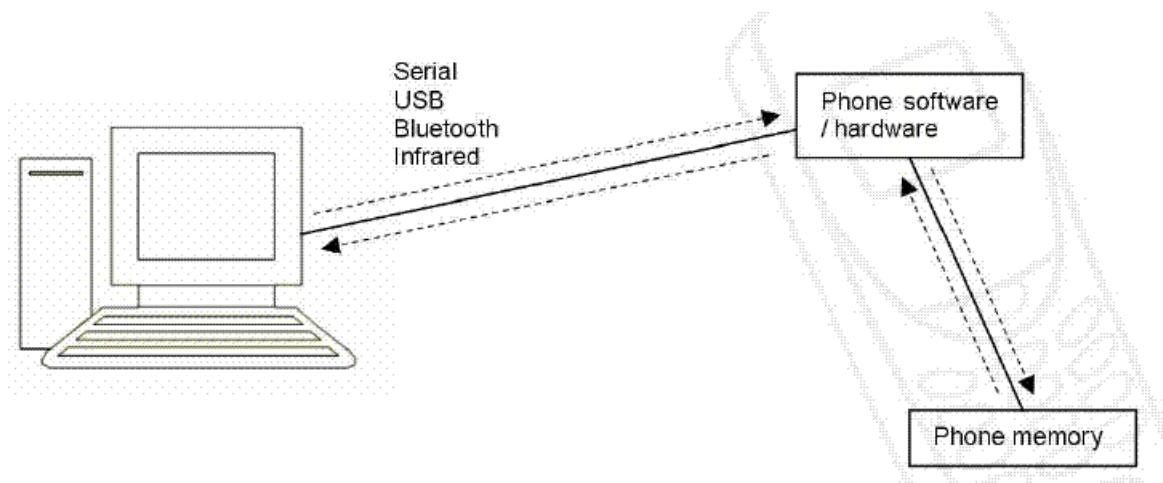


Figure 4: Access to mobile device. This figure was copied from McCarthy (2005)

Forensic tools for mobile devices are able to gain data from the device in two ways: Physical and logical acquisition. As stated by Ayers R. et al. (2005) Physical acquisition is about copying every single bit of the physical store whereas logical acquisition is about copying the logical storage objects of the logical store, such as directories and files. The advantage of physical acquisition over the logical is that it allows data which lay on the unallocated RAM or unused filesystem space to be examined by the investigator whereas the logical acquisition would have ignored this kind of data. The second advantage of the physical acquisition is that images of physical devices can easily be read by other tools for further examination if this is necessary. On the other hand, logical acquisition give the investigator a more understandable data organization. Hence, it is advised to use physical and logical acquisition if possible (Ayers R. et al, 2005).

The following table summarizes several forensics tools, open source and commercial, which can be used in an examination. One must consider that functionalities of tools are subject to change or add due to the continuous research of the tools . Table 8 illustrates some of the tools:

	<i>Function</i>	<i>Target Devices</i>
Forensic Card Reader	Acquisition, Reporting	SIMs
ForensicSIM	Acquisition, Examination, Reporting	SIMs and USIMs ¹
SIMCon	Acquisition, Examination, Reporting	SIMs and USIMs
SIMIS	Acquisition, Examination, Reporting	SIMs and USIMs
USIMdetective	Acquisition, Examination, Reporting	SIMs and USIMs
BitPIM	Acquisition, Examination, Reporting	Certain CDMA phones using Qualcomm chipsets
Oxygen PM (forensic version)	Acquisition, Examination, Reporting	Nokia phones
Oxygen PM for Symbian	Acquisition, Examination,	Symbian phones

1 (U)SIM is a smart card which contains a processor and an electronically erasable, programmable read only memory of about 16 to 128 KB (Jansen, W. and Ayers, R. 2006)

(forensic version) PDA Seizure	Reporting Acquisition, Examination, Reporting	Palm OS, Windows Mobile/Pocket PC, and Blackberry devices
Pilot-Link Cell Seizure	Acquisition Acquisition, Examination, Reporting	Palm OS devices TDMA, CDMA, and GSM phones
CellDEK	Acquisition, Examination, Reporting	SIMs and USIMs GSM and CDMA phones
GSM .XRY	Acquisition, Examination, Reporting	SIMs and USIMs GSM and CDMA phones
MobilEdit!	Acquisition, Examination, Reporting	SIMs and USIMs GSM phones
PhoneBase	Acquisition, Examination, Reporting	SIMs GSM phones
Secure View	Acquisition, Examination, Reporting	SIMs and USIMs TDMA, CDMA, and GSM phones
TULP 2G	Acquisition, Reporting	SIMs GSM phones

Table 8: Various Forensic Tools. This table was copied from Jansen, W. and Ayers, R. (2006)

Hoog , A. (2009) introduced some forensics acquisition techniques for the Android Operating System:

- Android Debug Bridge (ADB) is given along with Android Software Development Kit which provides the connection between an Android smartphone device and a remote workstation such as a personal computer. It is constituted of three main components: client, server and a daemon. Both client and the server are established on the remote workstation where the daemon is established on the mobile device (an Android emulator could work aswell). With ADB a remote execution of the commands on the Android phone is possible. Only if they are executed on an emulator or running under root² user these commands have root privileges. Some commands which might interest a forensic investigator and are available through ADB are dd, ls, pull (Distefano A, Me G. and Pace F, 2010).

- Nandroid backup: A group of tools which are able to back up and restore capabilities on rooted Android devices and are able to support NAND flash memory imaging with the help of a special boot mode. Its ability to acquire entries which were deleted from the internal memory File System is not supported in many forensic tools (Distefano A, Me G. and Pace F, 2010).

- Physical imaging by dd: Due to its nature, Unix-like traditional forensic command line tools such as dd, which allows a byte-level physical imaging of Unix files can work well enough not only to regular files but to device files as well (Distefano A, Me G. and Pace F, 2010). In addition, Hoog (2010) was able to develop a method for extracting the data from an Android device in the YAFFS2 but as stated by Liles, S., Kovacik, S., and O'Day, D. (2010) not any forensic tool have implement it yet. On another note, Lessard J. and Kessler C G. (2010) have managed to gain root access on Android through Windows and acquire an image of the SD card.

- Commercial tools: A group of commercial tools regarding the mobile forensics which

² Rooting a device means one gains access to the root directory / and has permissions to take root actions
Lessard J. and Kessler C G. (2010)

support or are willing to support Android, such as Paraben, Micro systemation, Celle Brite (Distefano A, Me G. and Pace F, 2010) and AFLogical (Liles, S., Kovacik, S., and O'Day, D., 2010). As stated by Distefano A. Me G. and Pace F. (2010) Open source mobile forensic tools are, for the time being, lacking competitiveness. Mobile Internal Acquisition Tool (MIAT) is an example of Open Source mobile forensic tools (Distefano A. and Me G. 2008).

- Serial commands over USB: This approach is considered to be controversial due to the fact that it was reported working on HTC Dream (aka Android G1) but it is not thoroughly tested on other devices so it may not be compatible. With this method the cabling is reverse engineered which allowed eavesdropping of the data transferred over the USB connection (Distefano A, Me G. and Pace F, 2010).

- Simulated SD card: Again this approach has not been tested and is only theoretical and is about modifying the update file in order to avoid any loss of the memory data and provide kernel-level tools to help to the acquisition of data (Distefano A, Me G. and Pace F, 2010).

- Software application: As almost all mobile operating systems today, Android provide the developers the ability to build their own applications. The Application Programming Interface (API) can provide to the developers some interesting functionalities such as providing the interface with the File System. The API by itself can support the realisation of applications which can explore, read and mirror the data managed by the File System not only from the SD card of the mobile device but for the internal memory storage volume as well (Distefano A, Me G. and Pace F, 2010).

Hoog, A. (2010) demonstrated that there are four primary ways for mobile forensics on an Android mobile device:

1. SD Card analysis: Most the Android mobile devices can have an external SD Card(FAT32 system) in order to store data. An investigator of an Android device must remove the SD Card and continue the examination with the standard way.

2. Logical acquisition: The investigator must first install a small Android Forensics application written by viaForensics, run it and then remove it from the mobile device. This application has the ability to acquire data related to:

1. Call Logs
2. Contact Methods
3. External Image Media (meta data)
4. External Image Thumbnail Media (meta data)
5. External Media, Audio, and Misc. (meta data)
6. External Videos (meta data)
7. MMS
8. MMSParts (includes full images sent via MMS)
9. Organizations
10. People
11. SMS
12. List of all applications installed and version
13. Contacts Extensions
14. Contacts Groups
15. Contacts Phones
16. Contacts Settings

All these data are written on an SD Card which the investigator must insert in the mobile device he is examining. New data sources are developed continually and added weekly (Hoog, A, 2010).

3. Physical acquisition: A technique was created in order to physically gain a “dd” image file from supported Android device but it is a prerequisite that the investigator should have root privileges on the device. In addition the investigator must have a notable amount of information.

4. Chip-off: If the investigator has access to forensics lab facilities he or she can use the chip-off techniques related to the NAND memory.

Ahmed R and Dharaskar R. (2008) claim that mobile forensic tools are too immature so they can not cope with the new technologies on mobile devices such as the wider use of applications and the large storage capacity. Most of the forensics tools are created by third party companies, thus they are not independently verified or tested for forensic soundness. Jansen, W. and Ayers, R. (2004) state that forensics tools which were not developed only for forensic examinations purposes are questionable and should be thoroughly evaluated before use.

Mokhonoana P. and Olivier M. (2007) demonstrate some requirements for forensic tools so that they can be more effective and are based upon their ability to acquire data and are not related to the presentation of the results, compression and integrity verification. The requirements are as follow:

- 1) The forensic tool or method should minimise changes to the device.
- 2) The forensic tool or method should retrieve as much data as possible.
- 3) The forensic tool or method should minimise user interaction with the device.

2.3.3 Guidelines of good forensic practice

Various guidelines of good forensics practice exists and are listed below along with a short discussion about every guideline and its integration on mobile phone forensics.

First there is the Association of Chief Police Officers (ACPO, 2008:4) guidelines which have four principles involved in computer-based electronic evidence:

- *“Principle 1: No action taken by law enforcement agencies or their agents should change data held on a computer or storage media which may subsequently be relied upon in court.*
- *Principle 2: In circumstances where a person finds it necessary to access original data held on a computer or on storage media, that person must be competent to do so and be able to give evidence explaining the relevance and the implications of their actions.*
- *Principle 3: An audit trail or other record of all processes applied to computer-based electronic evidence should be created and preserved. An independent third party should be able to examine those processes and achieve the same result.*
- *Principle 4: The person in charge of the investigation (the case officer) has overall responsibility for ensuring that the law and these principles are adhered to.”*

Ahmed R and Dharaskar R. (2008) illustrate that the Principle 1 of this guidance is slightly inaccurate when it comes to mobile phone forensics. In addition, Jansen, W. and Ayers, R. (2004) illustrate that this deflection of the principle 1 is due to the fact that mobile phone storage is continuously modified, hence the data can be altered without the user taking any actions. Jansen, W. and Ayers, R. (2004) are referring to the third version of the ACPO Guidelines, but Principle 1 has not changed in the fourth version, hence their statement can be applied for fourth version as well. When a mobile phone acquisition is taking place attention should be given on how to not alter the contents of the storage media of the phone. Moreover, Ahmed R and Dharaskar R. (2008) note that the investigator should have good knowledge not only on the hardware but also on the software of the mobile device he or she is examining. The investigator must also have very good knowledge of

the tool he or she is going to use in order to gain data from the phone. Jansen, W. and Ayers, R. (2004) advise investigators to use more than one tool when acquiring data from the device due to the fact that some tools do not inform you about error messages during the acquiring procedure.

Owen P. and Thomas P. (2011) state that ACPO guidelines do not give any recommendations for forensic software and hardware to use during a forensic investigation. They also add that it would be better if information and testing of forensic tools will be included in ACPO guidelines as this will help to the consistency of organisational liaison and will eventually speed the efficiency of forensic examinations. As stated above, Owen P. and Thomas P. also advocate that the forensic examination of mobile devices is correlated with the methods and tools that are provided by the manufactures and that ACPO guidelines do recognise this, due to the fact that it is crucial for the integrity of a device. In addition, ACPO guidelines provide legal considerations and principles that must be followed and ensure that the integrity of the evidence remains intact. On the other hand, as they state ACPO guidelines miss to guide how the law enforcement should handle mobile devices during examination.

International Organization on Computer Evidence (IOCE) published some guidelines for forensic examination of digital technology. These are the general principles of the IOCE (2002):

- *“The general rules of evidence should be applied to all digital evidence.*
- *Upon seizing digital evidence, actions taken should not change that evidence.*
- *When it is necessary for a person to access original digital evidence that person should be suitably trained for the purpose.*
- *All activity relating to the seizure, access, storage or transfer of digital evidence must be fully documented, preserved and available for review.*
- *An individual is responsible for all actions taken with respect to digital evidence whilst the digital evidence is in their possession”.*

Ahmed R and Dharaskar R. (2008) state that IOCE Guidelines can not be applied to evidences acquired from smartphone devices due to the fact of they have a dynamic nature. In addition mobile phone acquisition tools which are claiming to be forensically sound do not really have access to the device memory but instead they use various commands provided by the device's software or hardware interface. Thus, the forensic soundness of such tools its relying on the devices' software or hardware interfaces and the investigator can not know verify if such tools are altering the device's memory.

Several other guidelines for forensic examinations exist and among them are the Best Practices for Seizing Electronic Evidence published by the United States Secret Service and the Electronic Crime Scene Investigation: A guide for First Responders by the the National Institute of Justice (NIJ). As Ahmed R and Dharaskar R. (2008) stated both of these guidelines do not have any information on how to forensically examine a smartphone due to the fact that they are both considered to be outdated. On the other hand, these guidelines note that mobile phones can be useful since they might have some evidence but they do not expand on it and also do not take in consideration the storage capabilities and applications of smartphones. Moreover, both of the guidelines do not inform that applications of phones may have evidential significance. Symbian, Windows Mobile and Android based phones are found to be vulnerable to malicious applications (McCarthy P. 2005, Cannings R., 2011). Nevertheless, all applications which can be installed on mobile phones can be seen as evidence because they might have information regarding illegal activities or log files or data which can be considered as evidences (Ahmed R and Dharaskar R., 2008). Hence, all applications and data which are related to them such as web mail applications, browsers, logs related to Bluetooth, Infrared (IrDA), WiFi connections, browser history data, instant

messaging data etc must be examined by the investigator in a forensic examination of a mobile device.

Patzakis (2003) conveys that when forensic investigations are taking place, they must be done with proper forensic processes. He continues by stating that even the most widely known guidelines may allow mishandling of potential digital forensic evidences. The use of chain of custody, when it comes to forensic investigation, is not only to verify the data but also to demonstrate that the data exists and was used. Moreover, chain of custody will indicate that an evidence which was placed in a media is related somehow to unauthorized actions. Hence, it is important to have a report which proves and maintains a chain of custody. This report can be useful because, if an investigator does not use it, he or she will likely face difficulties to determine the exact location of evidence, despite the fact that he or she may be using proper forensic techniques. This kind of report, can be used as print outs of an electronic crime scene and may be used as key evidence in trials. In order for an investigator to correctly maintain a digital chain of custody, four basic steps must be followed:

- Physically control the scene. In the case of a remote network investigation, the investigator must log all access and connectivity through an integrated and secure reporting function
- The investigator must create a binary, forensic duplication of original data in a non-invasive manner.
- A digital fingerprint must be used continually in order to verify data authenticity
- Log all investigation details in a thorough report generated by an integrated computer forensics software application

In addition Bradford P G, Brown M, Perdue J, Self B. (2004) state that chain of custody, documents the details when various people can have access to potential evidence so it is vital to keep a chain of custody easy to understand and well documented.

Mokhonoana P. and Olivier M. (2007) note that when a forensic investigator examines a mobile device he or she must be careful to not alter the data acquired from the mobile device in order to be used in the court. This can be accomplished by a documentation of all the interactions made with the mobile device, which in the future can verify that all the interactions with the mobile device did not have any effect on the integrity of the evidence. The more the investigator interacts with the mobile device the possibilities for omitting the documentation or making mistakes increases. Hence, it is vital for the investigators to use a method for acquiring data from mobile devices which needs as little interaction as possible.

All of the guidelines lack some key points regarding a smartphone device forensic examination. As stated by Hoog, A (2010) in digital forensics it is vital to ensure that no modification of the target device have been made by the investigator. Due to the fact that mobile phones do not have hard drives which can be shutdown and imaged in forensically sound way and that Android forensic techniques do modify the mobile device, the investigator must use their discretion during the investigation and if by any chance the information on the device is altered, he or she must be able to explain how it was altered and why was it altered.

To sum up, I will be using the ACPO principles for evaluating the method that will be the outcome of this study due to the fact that ACPO principles play an active and crucial role in guiding and coordinating the direction and development of the police service in England, Wales and Northern Ireland (Induruwa A., 2009). Moreover, Induruwa A (2009) adds that ACPO are important and formulate guidelines to disseminate best practices and also guide forensic investigators to make examinations with in some standardised frameworks. In addition, the latest release of the ACPO Guidelines have a separate section regarding mobile phone seizure and examination. This section illustrates how the four principles for digital evidences are ported in to the field of mobile forensic

and provides an analysis of the the implications regarding the personnel involved in mobile phone forensics (Induruwa A, 2009). Last but not least Induruwa A. (2009) state that all UK investigators are expected to follow these guidelines and that investigators from other countries also found the ACPO guidelines useful.

2.4 Related work

Lessard J. and Kessler C G. (2010) have used various methods in order to acquire information from an Android mobile device, in this occasion Sprint HTC Hero was used, and concluded with some results on each method.

Firstly they used a dd analysis with Access Data Forensic Tool Kit (FTK) version 1.81 which was selected due to the fact it has data carving and searching functionalities. Hence, its ability to search for strings was considered paramount.

The advantages of this method were the fact that they were able to recover deleted text messages and contacts which in other circumstances it may have been difficult to find and the fact that they found passwords quite easily. On the other hand, in order to use this method, one must have root access of the phone and the results were fragmented. They spent, as claimed, a lot of hours to locating and setting up every data together, whereas another forensic tool may have handled better the data.

The second method they used was a logical analysis of specific databases. The advantage of this method was that they were able to recover almost everything, which is consider helpful for an investigator during a forensic examination. The type of information that they were able to acquire was: call history, pictures, MMS/SMS messages, e-mail data with complete messages, and even GPS data, voice mail and passwords. The disadvantage of this method, is that it also requires root access and they were not able to recover all deleted SMS messages, phone records and contact info.

The third and last method they used was the data extraction with the CelleBrite UFED. UFED, is a stand alone hardware device which is able to acquire data such as contact lists, address books, pictures, videos, music, ringtones, text messages, call history, and determining device information. Using the above method they were able to recover MMS/SMS messages, call logs, photos, video and contact information and found that this is a simple and stand-alone method. In contrast, they were not able to acquire any data regarding emails and browser information. Moreover, this method only uses Logical extraction due to the fact that physical acquisition is not yet supported.

When it comes to the question which one of the method is better, they add that when they were browsing the databases logically they were able to find biggest amount of data in a easily viewable way. On the other hand, a dd image is believed to be very valuable and could be considered as the best tool. So, from their point of view FTK is valuable when one wants to search for something specific, such as a string of text.

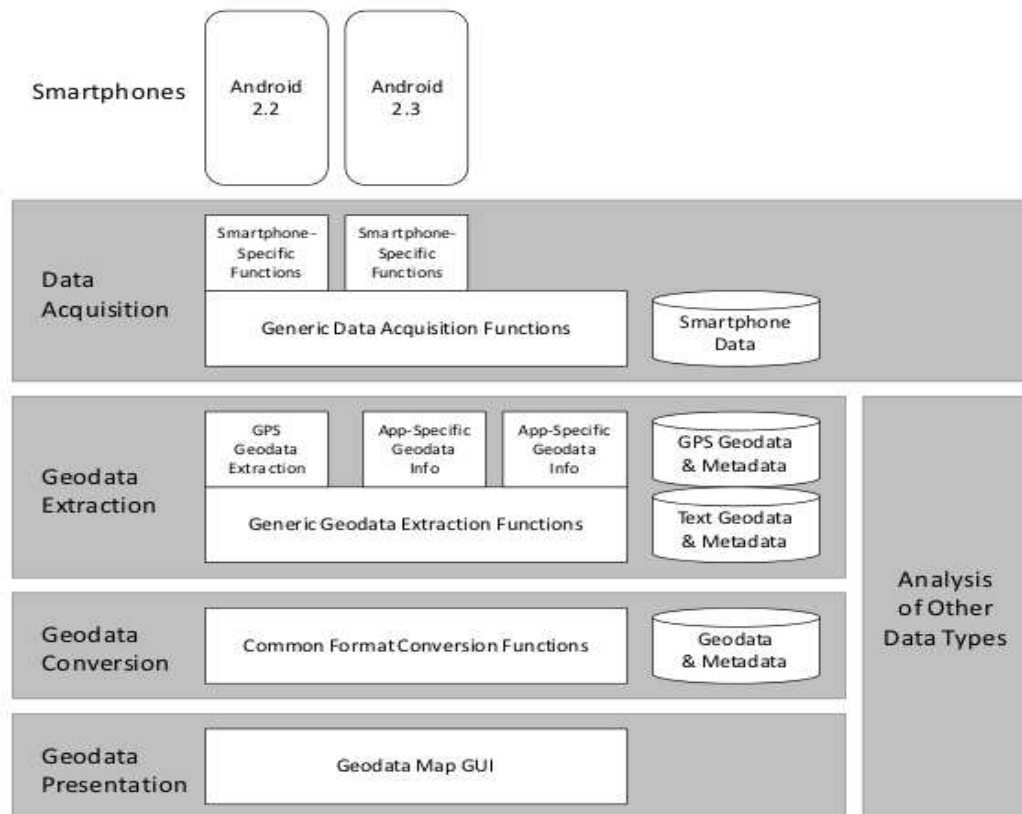


Figure 5: Modular Architecture for Geodata Forensic Analysis (This figure was copied from Maus S., Höfken H. and Schuba M. (2011))

Maus S., Höfken H. and Schuba M. (2011) in their study were focused on a forensic analysis of an Android smartphone and especially for crime investigations about geographical or spatial data (geodata). Their interest was derived from the fact that many apps in smartphones use geodata for additional capabilities like providing information regarding the smartphone's user location or store geodata to analyse in later stages. As they state, geodata can provide the forensic examiner a geographical location history of the smartphone's user which is vital in a crime investigation. Hence, the more geodata available to the investigator the more precise the location history of the user can be. In order to achieve this, they did not only use GPS data which were acquired from the smartphone, but they also used textual geodata. Figure 5, illustrates the method they used to make a comprehensive analysis of the possible location based apps which are available for smartphone's today. This approach was tested on an Android smartphone, but as they state, it can be theoretically ported to other devices which run other smartphone mobile operating systems, like iOS etc.

Vidas T., Zhang C. and Christin N (2011), in their study explored the special device boot modes of Android and the Android's partitioning schema. They also illustrated the composition of an Android bootable image and discuss whether a creation of such image will be used in a forensic analysis. They also demonstrated a general process for data collection on Android devices using these boot images and related results of investigations which they carried out on various Android devices. This was achieved with the use of a custom recovery boot image and concluded that on Android smartphones data can be acquired with a very little probability of corrupting user data.

Mislan, R.P., Casey, E., and Kessler, G.C. (2010) tried to formalize on scene triage process by introducing it to the wider forensic process and providing some guidelines. In addition they defined

some requirements for automated triage tools. These are the guidelines they introduced for scene triage process with chronological order:

1. Initiate chain of custody
2. Isolate device from network (if feasible and applicable)
3. Disable security features (if feasible and applicable)
4. Extract limited data
5. Review extracted data
6. Preview removable storage media

As stated, the on-scene triage inspection will help investigators to determine and take decisions faster and to not wait for official results from a Digital Forensic Lab. Moreover, this on triage process must be allow a fast acquisition of information before sending it to a Digital Forensic Lab. This can help the investigation by reducing time needed from the lab to return the results of the examination and may allow Digital Forensic Lab to focus on high level examination of the mobile devices where on-triage process can focus on low-level examination.

Thing L., Kian-Yong Ng and Ee-Chien Chang (2010) have used the Android platform in order to introduce a system which can analyse the mobile device's memory, which continually changes, and can also output real-time evidence. Their system could analyse the persistency of data which are created by interactive and communication based applications. Various scenarios of communication have been analysed in this research with parameters such as message lengths, messaging intervals, dump intervals, keypress interval changing values during the research.

With their research they were able to extract the information, that outgoing messages of the investigated phone had higher persistency than the incoming messages. Thus, they concluded that in a more real life scenario where the mobile device owner is less likely to send messages in a small period of time the rate for acquiring evidence for a forensic examination is acceptable. Moreover, they highlighted the need for a live memory forensic tool in order to cope with the volatile memory.

According to Arthur C. (2011), Android smartphone devices are collecting data regarding the user's movements and this discovery is credited to Magnus Eriksson, a Swedish researcher. Eriksson's study has shown that Android smartphone devices keep a record of the user's locations along with unique IDs of the last 50 mobile masts that it has communicated and with the last 200 Wi-Fi networks it has scanned. This kind of information is stored in a circular buffer which overwrites the oldest information when the list is full. It was not discovered if the lists are sent to Google or not (cited by Arthur C. 2011). In addition Arthur C (2011) provides the official website of the tool named android-locdump that it was created by Magnus Eriksson which parse the two lists.

As stated on the official website of the android-locdump tool, the tool is basically a dumper which parses the files from the Android location provider. These files are named cache.cell and cache.wifi and are located in the folder /data/data/com.google.android.location/files of any Android smartphone device. If someone wants to access these files he or she must have root access. As it is also stated, this functionality can be disabled by unchecking the option under Settings → Location and Security → Use wireless networks. When this option is unchecked, the smartphone devices that run Android version 2.3 delete the above files whereas smartphone devices running Android version 2.2 keep the files but stop updating them. Information regarding other version of Android and how they handle these files are still unknown (Android-Locdump, 2011).

According to the official website of the android-locdump the command that can be used to enable the tool is the following:

parse.py <cache file>

where the cache file is either the cache.wifi or the cache.cell file (Android-Locdump, 2011).

In Table 9a there is an example of the outcome of parsing the cache.wifi file and on Table 9b an example of parsing the cache.cell file. It can be easily understood that Android smartphone devices can keep information about the location of the smartphone and the time the phone was on that location.

<i>key</i>	<i>accuracy</i>	<i>conf.</i>	<i>latitude</i>	<i>longitude</i>	<i>time</i>
50:63:13:57:42:7e	80	92	57.689354	11.994763	04/11/11 10:03:51 +0200
e0:cb:4e:7e:cc:53	75	92	57.689340	11.994495	04/11/11 10:03:51 +0200
4c:54:99:14:47:68	57	92	57.708979	11.916581	04/11/11 01:14:53 +0200
00:26:18:0a:ad:cb	60	92	57.709699	1.917637	04/13/11 08:40:36 +0200
00:22:15:28:3f:7a	60	92	57.699467	11.979340	04/13/11 11:52:16 +0200
00:22:3f:a7:d9:fd	65	92	57.699442	11.979343	04/13/11 11:52:16 +0200

a) Results after parsing cache.wifi file. This table was copied from Android-Locdump (2011)

<i>key</i>	<i>accuracy</i>	<i>conf.</i>	<i>latitude</i>	<i>longitude</i>	<i>time</i>
240:5:15:983885	1186	75	57.704031	11.910801	04/11/11 20:03:14 +0200
240:5:15:983882	883	75	57.706322	11.911692	04/13/11 01:41:29 +0200
240:5:75:4915956	678	75	57.700175	11.976824	04/13/11 11:52:16 +0200
240:5:75:4915953	678	75	57.700064	11.976629	04/13/11 11:53:09 +0200
240:7:61954:5892	1406	75	57.710205	11.921849	04/15/11 19:46:31 +0200
9					
240:7:15:58929	-1	0	0.000000	0.000000	04/15/11 19:46:32 +0200
240:5:75:4915832	831	75	57.690024	11.998419	04/15/11 16:13:53 +02:00

b) Results after parsing cache.cell file. This table was copied from Android-Locdump (2011)

Table 9: Results when parsing cache.wifi and cache.cell files (Android-Locdump, 2011)

Hoog A and Strzempka K. (2010) made an overview of forensic tools and their abilities to gain information regarding to the Apple's iPhone. For the needs of their study, they have used a not jailbroken iPhone 3G running iOS 3.1.3 firmware and the tools were running either under Windows XP Professional Service Pack 3, either a Mac or Linux machine according to the tools needs.

Table 10 shows the outcomes of their research. Each tool have been tested and a value was given to each tool according to its functionality upon installation (weight 0.1), acquisition of data (weight 0.1), reporting (weight 0.1), and the accuracy of the data (weight 0.7). An overall average 3.3 was then computed according to the weights on each category. Everything below the average was listed as "Below", equal to the average as "Meet" and above the average as "Exceed".

<i>Software</i>	<i>Installation</i>	<i>Acquisition</i>	<i>Reporting</i>	<i>Accuracy</i>	<i>Overall</i>
<i>Cellebrite</i>	5.0	4.0	2.5	3.0	3.4 (Exceed)
<i>FTS iXAM</i>	3.0	3.0	3.0	4.2	3.9 (Exceed)
<i>Oxygen</i>	5.0	4.0	2.5	3.0	3.4 (Exceed)
<i>Forensic Suite</i>					
XRY	5.0	5.0	4.0	3.2	3.7 (Exceed)
Lantern	5.0	5.0	4.0	3.2	3.7 (Exceed)
MacLockPick	3.0	3.0	1.0	3.1	1.9 (Below)
Mobilyze	5.0	5.0	3.5	2.9	3.4 (Exceed)

Zdzriarski	5.0	3.5	2.0	4.2	4.1 (Exceed)
Paraben	3.0	2.5	2.5	2.9	3.0 (Below)
Mobile Sync	4.5	5.0	3.0	3.1	3.5 (Exceed)
Browser					
CellDEK	4.0	3.5	3.0	2.7	2.9 (Below)
EnCase	4.5	4.5	3.0	2.9	3.3 (Meet)
Neutrino					
iPhone	5.0	3.0	2.5	2.4	2.8 (Below)
Analyzer					

Table 10: Overall Rankings. This table was copied from Hoog A and Strzempka K. (2010)

Pooters , I. (2010) created a tool, Symbian Memory Imaging Tool (SMIT) which is able to run on Symbian OS and can provide linear bitwise copies of the internal flash memory of the phone. SMIT is able to work on Nokia phones models E65, E70 and N78 and can acquire an image of the user drive volume of the file system layer. Moreover Pooters I. (2010) used cryptographic hashes in order to be able to verify the integrity of the data acquired by SMIT.

On the other hand, the Symbian's security mechanisms has prevented the writer from using some functions provided by the Application Programming Interface. These security mechanisms were overwhelmed using some hacks, but these hacks can not work on all mobile devices which are using Symbian. Moreover, these hacks may alter important data so it is in the opinion of the investigator if he prefer a full image of the drive with probably some altered data from nothing. The research concludes with the location of data of some known applications such as the messages and phonebook contacts and how to find some information regarding deleted messages in log files, the index file, slack space and unallocated clusters.

Casey, E., Bann M. and Doyle J (2010) made a research regarding commercial and open source tools when it comes to windows mobile forensics. An introduction to Windows mobile is achieved with some indications on how a forensic investigator can retrieve data from these systems. The paper continues with information on where a forensic investigator should look for when searching for evidence. In addition, this article is about Windows mobile devices which are not password protected, due to the fact that password protected devices may have a configuration which lead them to delete all information regarding a user after some failed login tries.

Klaver C. (2010) examined Windows Mobile devices not in the scope of logical acquisition, but with the help of commercial forensics tools and talks about typical hardware of a Windows Mobile Device and where an investigator should look for user data. Moreover, Klaver C. (2010) examines some software components of Windows Mobile devices (the bootloader, heap, filesystem and databases), where information about the user can be found and might be interesting for an investigator.

Klaver C. (2010) continues by describing how to retrieve a forensic duplicate of information on these kind of devices and illustrates some ways for investigators to make a physical acquisition of the Windows mobile device. Finally, some tools and techniques along with their results of a physical acquisition is illustrated. Moreover, Klaver C. (2010) introduces a method for investigators to examine isolated Windows Mobile Device databases volume files for active and deleted data.

Williamson B et al (2006) made a research regarding the expectations of forensic analysis and the methodologies they used. They used 4 forensic tools, TULP 2G, MOBILedit! Forensic, Cell Seizure and Oxygen Phone Manager on seven mobile devices which were selected for this research, Nokia 6385, 5110/5110i among others. Their research, as stated, was a partial success due to the fact that the investigators were not able to acquire any data that were deleted from the mobile devices. They were able to acquire a portion of data using MOBILedit! and Cell Seizure but this amount of data can not be compared with the data they extracted from Oxygen Phone Manager and TULP 2G.

Moreover, they highlighted several issues that forensic investigators need to be aware when they are examining mobile devices. They concluded, by stating that investigators must guarantee the integrity of the data.

Mokhonoana P. and Olivier M. (2007) introduced a new method of installing a forensic tool on the mobile device to gain data. The advantage of this method is that it does not require any connection between the mobile device and the PC, since it access all of the data from the operating system. This method is compared with traditional methodologies of acquiring evidences. Their research concluded that using this method some data could not be acquired due to the operating system restrictions but it is an advantage that no connection to a computer is required. Hence, the investigation can take place on the crime scene avoiding in this way the probability of altering the mobile device's data.

3. Project's context and its relation with mobile forensics

3.1 Samsung Galaxy Europa

This project, as stated above, will investigate if information regarding Wi-Fi and bluetooth connections is stored on an Android smartphone and how can we acquire this kind of information during a forensic examination. In order to achieve the aims of the project an Android smartphone was needed. Hence, I used a Samsung Galaxy Europa (or Samsung GT-I5500). Table 11 illustrates some of the important specifications of the Samsung Galaxy Europa I used. In addition, the Samsung Galaxy Europa that was given to me, was rooted and as I was told this was done using the application called SuperOneClick version 1.9.5.

<i>Parts</i>	<i>Description</i>
CPU	600 MHz Clock Speed
Memory	Internal: 170 MB External: 2 GB
Operating System	Android OS v2.2.1 (Froyo)
Bluetooth	v2.1 with A2DP
WLAN	Wi-Fi 802.11 b/g/n, DLNA

Table 11: Specifications of the Samsung Galaxy Europa I used for the needs of this project

3.2 Android developers tools

In order to fulfill the aims of the current study, firstly it was needed to make a research concerning the filesystem that it is used on the Samsung Android smartphones. As stated above, Samsung uses a proprietary filesystem so I could not use the typical Android partition information regarding MTD as this was described by Vidas T., Zhang C. and Christin N. (2011). It was also needed to make sure how the Samsung Galaxy Europa that I had in my possession implements the Sector Translation Layers (STL) and Block Management Layers (BML). To achieve this the Linux's DF command was used and a verification took place with the use of Busybox, which according to its official website is a combination of tiny versions of various UNIX utilities into a single small executables. It replaces various GNU's utilities like fileutils and shellutils. These replacements have fewer options from the utilities provided by GNU but have and can provide an accepted functionality which can be considered equal to the GNU's utilities. Busybox also offers a complete environment for small or embedded systems since it was written with size-optimisation and limited resources in mind. Moreover, busybox is licensed under the GNU General Public License, version 2 (Busybox, 2008). In addition Schmidt A.D. et al (2011) state that busybox is compatible with the Android operating system. Details regarding how I managed and obtained information on the filesystem used on the Samsung Galaxy Europa will be provided in the next chapter at section 4.1.2 Partitions of Samsung Galaxy Europa.

The identification of the partitions of the smartphone was not the only issue I had to face. In order to accomplish the aims of the project, I had to found information regarding the logging system of the Android operating system. According to the official guide on the Android developers website the logging system of Android is maintained by a tool called logcat. The Android logging system, provides a way for obtaining and viewing various logs like system debugs outputs from different applications which are installed and outputs from various portions of the system. These outputs are all stored in circular buffers which can be viewed with the use of the logcat tool. This tool can be used from a terminal shell (Android Developers, Developers' Guide, 2011). Hence, in order to be

able to see the android logs, one must have access to ADB.

According to Android Developers, Developers' Guide (2011), Android Debug Bridge (ADB) is a command line tool which enables you to interact with an emulator or to any connected device which runs Android. ADB tool can be found in the platform-tools folder of the Android's software development's kit (SDK), it is based on the client-server model and consists of three components:

1. Client: This component executes on the development machine and can start by typing the ADB command. Other Android tools like the Android Developers Tools (ADT) plug-in and Dalvik Debug Monitor Server (DDMS) can also create ADB clients.
2. Server: This component runs on the development machine. It is responsible for the communication between the Client and Daemon which runs on a device or emulator that runs the Android operating system.
3. Daemon: This component runs in the background of a device or emulator that runs the Android operating system.

ADB commands can be executed from command line on the development machine and its format must be like this:

adb [-d | -e | -s <serialNumber>] <command>

Table 12 demonstrates a short description of the -d, -e and -s <serialNumber> arguments. All the commands which are available for the ADB tool can be found at the official Android Developers' Guide (Android Developers, Developers' Guide, 2011).

<i>Arguments</i>	<i>Description</i>	<i>Comment</i>
-d	Direct an adb command to the only attached USB device.	Returns an error if more than one USB device is attached.
-e	Direct an adb command to the only running emulator instance.	Returns an error if more than one emulator instance is running.
-s <serialNumber>	Direct an adb command a specific emulator/device instance, referred to by its adb-assigned serial number.	If not specified, adb generates an error.

Table 12: A short description of the -d, -e and -s <serialNumber> argument. This table was copied from Android Developers, Developers' Guide (2011).

As stated above, in order to obtain the logs of an Android smartphone device, logcat tool is needed. This is the syntax of the logcat tool command:

[adb] logcat [<option>] ... [<filter-spec>] ...

Table 13 illustrates information regarding the option and filter spec arguments:

<i>Argument</i>	<i>Description</i>
-b <buffer>	Loads an alternate log buffer for viewing, like <i>event</i> or <i>radio</i> . The <i>main</i> buffer is used by default.
-c	Clears (flushes) the entire log and exits.
-d	Dumps the log to the screen and exits.

-f <filename>	Writes log message output to <filename>. The default used is stdout
-g	Prints the size of the specified log buffer and exits.
-n <count>	Sets the maximum number of rotated logs to <count>. The default value is 4. Requires the -r option.
-r <kbytes>	Rotates the log file every <kbytes> of output. The default value is 16. Requires the -f option.
-s	Sets the default filter spec to silent.
-v <format>	Sets the output format for log messages. The default is brief format.

Table 13: information regarding the option and filter spec arguments of logcat command. This table was copied from Android Developers, Developers' Guide (2011).

Details regarding how I used the ADB tool and logcat for the needs of this study are illustrated in the next chapter at section 4.1.1 ADB Tool

3.3 Definition of the scenarios

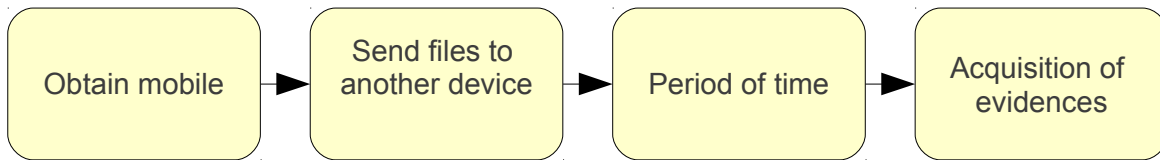
For the needs of this project four scenarios were created. This was done in order to implement the scenarios and then investigate them to find what kind of information is stored on the Android operating system and how can we obtain this information. All scenarios that were implemented were based, but were not identical, on real life scenarios.

The first two scenarios which were implemented were regarding bluetooth connections and the last two were related to Wi-Fi connections. The scenarios that were used are the following:

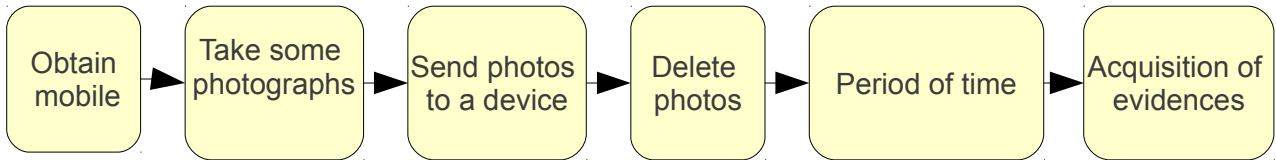
1. Child pornography photographs were founded on a company's laptop. The man who was in possession of the laptop claims that he has nothing to do about it. Police believes that the photographs were taken by the suspect's smartphone and were transferred on his laptop via bluetooth connection. Evidences that may lead to the confirmation of this accusation are needed (similar case was found at Daily Mail, 2009).
2. A man was arrested after some witnesses have seen him leaning over and holding what appeared to be a smartphone below the girl's skirt line. His smartphone was seized by the police after the accusations of the witnesses were verified from a video camera that was in the area. Evidences are needed to see if these images were send to any other device through bluetooth connection (similar case was found at San Gabriel Valley Tribune, 2011) .
3. A student has posted some questions from a test on an online forum and requested the answers from others. Evidences are needed to confirm that the smartphone was connected to the wireless network of the building and that the post was made by his smartphone (similar case was found at The Guardian, 2011).
4. A man is thought that he is uploading child pornography images using his neighbour's unsecured Wi-Fi network via his smartphone. He is using his dropbox application to upload these pictures (similar case was found at Huffington Post, 2011).

In Figure 6 there is a representation of each scenario as this was implemented in the current study. Figures a and b are the scenarios which are related with bluetooth connections and figures c and d are the scenarios related with Wi-Fi connections.

Bluetooth:

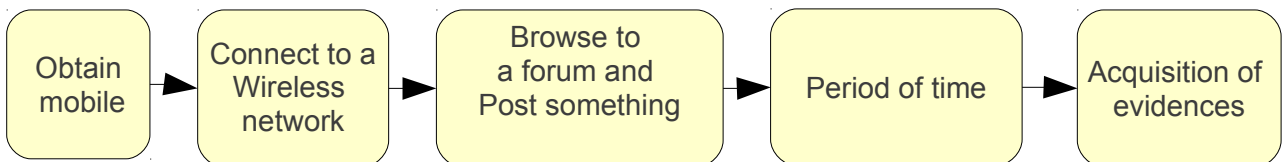


a) Scenario 1

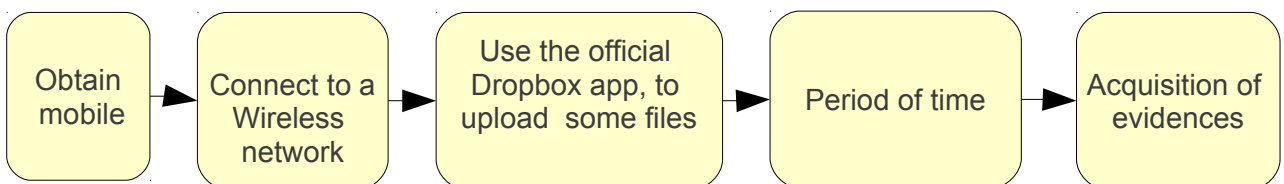


b) Scenario 2

Wi-Fi:



c) Scenario 3



d) Scenario 4

Figure 6: Representations of the four scenarios

As it can be seen from the representations, the first thing I did during the scenarios is to get the mobile in my possession and then make an action, regarding the scenario. After that I used the mobile device as my personal mobile phone for a specific period of time. This means that I was using the smartphone device regularly as a normal user, making phone calls, starting some applications and interacting with them etc. I repeated each scenario three times with different periods of time occurring between the supposed illegal activity and the acquisition of the evidence. I used 30 minutes 6 hour and 12 hours as periods of time due to the fact that the Android smartphone devices use circular buffer to hold their logging information. Hence, I suspected that these devices will not keep information for a long period of time.

Due to the fact that Wi-Fi is related with many applications, I decided to focus on two applications: the official browser of Android operating system and the official dropbox application. Dropbox, which began its service in 2008, according to Mulazzani M., et al (2011) is the leading cloud storage provider on the Internet with 10 million users and 100 billion files as of May 2011.

More details on how the scenarios were implemented and what results I obtained from each scenario can be found in the next chapter.

3.4 Project's relation with background

Due to the fact that the Android operating system was first released in 2008, not many studies have been done in the area of forensics regarding Android devices and specifically no studies have been done, at least to my knowledge, concerning bluetooth and Wi-Fi connections.

In order to make this kind of investigation it was needed to study and identify what a smartphone device is. In addition, Android's operating system history and the way it works needed to be known in order to fulfill the project's aims. As mobile forensics is a rather new section of digital forensics, attention was drawn in the digital forensics and especially mobile forensics, what it is, what tools exist and which are the most widely used guidelines. These guidelines will be used to evaluate the method that it is proposed towards the end of this study for acquiring evidences regarding bluetooth and Wi-Fi connections. These guidelines are important since they will provide a good evaluation of the method, hence, it is important to research what guidelines exist and what are the pros and cons of each guideline.

Since this project will be about forensic examination and investigation of a smartphone device, it is vital to note what other researchers have found. As stated in the previous chapter few studies have been made regarding the forensic analysis of a smartphone device. This project was based on the studies that have already been conducted in order to examine evidences found on the Android smartphone devices regarding bluetooth and Wi-Fi connections. In addition, in the context of this study, new methods will be used in order to analyse the filesystem that is used on Samsung Galaxy Europa which differs from the ordinary filesystem used in Android smartphones. Furthermore, the location of evidences regarding bluetooth and Wi-Fi connections on Android device and the way an investigator can obtain them will be presented. In order to achieve this, I had to study what others researchers have found in their studies, which were not related to bluetooth and Wi-Fi but may be vital during a forensic investigation.

4. Examination and analysis of the scenarios

Before starting the description of what it was done regarding the examination of the scenarios, it is noteworthy to state that during all the research prior to the examination and during examination, Ubuntu 11.04 was used as the main operating system on the PC I was using. Hence, from now on all command line tools or commands that are described or stated below were used under Ubuntu 11.04.

4.1 Research prior to the examination

4.1.1 ADB Tool

During the investigations of the four scenarios the use of the ADB tool is considered vital. Hoog, A. (2009) and Distefano A, Me G. and Pace F. (2010) consider the ADB tool important, but in order to work, it requires the USB debugging option which is located under Settings → Applications → Development to be enabled. This option enables the Debug mode when a USB cable is connected to the smartphone device.

When a smartphone is in the Debug Mode various commands can be executed by the ADB tool. It must be noted that all of the commands require the server component of the ADB tool to be started before executing any command. If the ADB server is not started when entering an ADB command for the first time, the ADB tool will automatically try to start the server. For the needs of this project I used a USB cable to connect the Samsung Galaxy Europa to my PC.

ADB tool provides two commands named pull and push which enable the ability to copy data files to and from a device or an emulator which runs Android. These commands let you copy directories and files to any location you like, if you have the privileges to do so. The syntax of the commands is as follows:

```
./adb pull <remote> <local>  
./adb push <local> <remote>
```

Local and remote are paths for the target files or directory on the development machine and on the device or the emulator which runs Android, respectively. Here's an example:

```
./adb pull /sdcard/1.png ./photos
```

This example copies the 1.png file which is located on the smartphone device in the folder /sdcard to the photos directory which is located in the directory where the adb is executed.

In order to use the shell to interact with the android smartphone device, I opened the platform-tools folder which is found in the Android's SDK folder and typed the command:

```
./adb shell
```

This provided me a remote shell where I could interact with the Samsung Galaxy Europa using various commands. This was vital for the forensic examination since it provided the ability to use the dd command. According to Chessman (1996) dd is a command found in the UNIX utilities and must be in everyone's knowledge. In addition, this command is able to strip headers, extract parts of binary files and write into the middle of floppy disks. In general, dd copies and if it is requested

converts data. It uses an input buffer, a conversion buffer (if a conversion is requested) and an output buffer. The input buffer size is determined by the input file or the device, then the conversions, if any, are applied and eventually writing in the output buffer is done. Finally it outputs the number of full and short blocks that have been read and written (Chessman, 1996).

Lessard J. and Kessler C G. (2010) have used dd command on their forensic analysis as stated in chapter two. In their study they used the following command (PARTITION was replaced with the partition they wanted to copy):

```
dd if=/dev/mtd/PARTITION of=/sdcard/mtd0.dd bs=1024
```

Due to the fact that they already knew the partitions and the filesystem that was used by the smartphone they had, they used this command to make a bit by bit image of the partition they liked. The command they used is using a block size of 1024 bytes and copies the image of the partition they chose to the SD card. As Lessard J. and Kessler C G. (2010) state it is very important to use a formatted and wiped SD card and the SD card that was used in the smartphone device that it is under examination should be put aside. It also crucial to not change the input file (if) and the output file (of) parameters because this will lead to deletion of any data. Since I was able to run a remote shell from the smartphone device and I was able to run the dd command on Android due to its Linux based kernel the partitions of the Samsung Galaxy Europa seemed to be the part that was missing.

There are some occasions where an error may occur during the dd procedure and can not be treated with the “conv=noerror” argument which informs the dd command to continue on read error. Android dd command does not support the “conv=error” argument and this can be overwhelmed using busybox (Linux Sleuthing, 2011).

A procedure which installs busybox on an Android device and obtains dd image files exists and it is described by Linux Sleuthing (2011):

1. Compile Busybox in order to be used in Android or obtain an existing busybox binary which was already compiled for Android by a trusted source.
2. Use the push command to transfer the binary of busybox to the SD card
3. Remount the SD card with executable permission using the commands below:

```
# mount -o remount, rw /PARTITION_OF_SDCARD /PARTITION_OF_SDCARD vfat
```

```
# mount /MOUNTING_POINT_OF_SDCARD /PARTITION_OF_SDCARD vfat rw, dirsync,  
relatime, uid=1000, gid=1015, fmask=0702, dmask=0702, allow_utime=0020,  
codepage=cp437, iocharset=iso8859-1, shortname=mixed, utf8, errors=remount-ro 0 0
```

where */PARTITION_OF_SDCARD* is the path the SD card partition and */MOUNTING_POINT_OF_SDCARD* is the mounting point of the partition of the SD card and can be found using the *mount* command. Examples are given below:

- */MOUNTING_POINT_OF_SDCARD = /dev/block/vold/179:1*
- */PARTITION_OF_SDCARD = /mnt/sdcard*

4. Browse to the folder where busybox is and use the command

```
./busybox dd if=/PARTITION_TO_COPY of=/DESIRED_PATH_OF_OUTPUT/NAME.img  
conv=noerror
```

PARTITION_TO_COPY is the path of the partition we would like to copy, *DESIRED_PATH_OF_OUTPUT* is the path where the image will be placed and *NAME* is

the desired name of the image. Examples are given below:

- */PARTITION_TO_COPY = /dev/block/mtdblock6*
- */DESIRED_PATH_OF_OUTPUT = /mnt/sdcard/*
- *NAME = mtd6*

As it is described by Linux Sleuthing (2011) the remounting and mounting procedure is done because the SD card is not mounted with executable permission. Hence remounting the SD card is vital. Moreover, busybox is placed on the SD card to avoid the modification of the smartphone's internal memory. In addition, in order to complete this procedure the investigator must have root access on the smartphone device.

4.1.2 Partitions of Samsung Galaxy Europa

In order to investigate the scenarios an identification of the partitions which are found in the Samsung Galaxy Europa was vital. In order to complete this I used the *df* command and *busybox* to verify its result. As Android is based on the Linux kernel a research was done in order to find a command in Linux which identifies the filesystems of the device. Hughes P. (1994) demonstrated that the *df* command is used to show the amount of disk space which is free on the filesystems of the device. The *df* command can be called using this format:

df [OPTION]... [FILE]...

When called with no arguments the *df* command displays used and free file space in blocks. The various columns are related with the name of the disk partition as it is in the */dev* directory, total space, blocks allocated, blocks available and the mount point of the file system (Hughes P., 1994). The *-h* option argument of the *DF* command returns the results which are related with blocks in sizes which are readable by humans for example 1KB, 234MB, 2GB.

As it can be read on the official busybox website (Busybox, 2008), *df* command is available and compatible with busybox hence it was needed to be tested on the Samsung Galaxy Europa to see the result of the command. The ADB tool provides the ability to run a remote shell but it requires the USB debugging to be enabled. This option enables the Debug mode when a USB cable is connected to the smartphone device.

Hence, I used a USB cable to connect the smartphone device to my computer and started a remote shell as described in section 4.1.1 ADB Tool. Afterwards, I could interact with the smartphone device through the ADB shell. Hence in order to test what results the *df* command I typed the command:

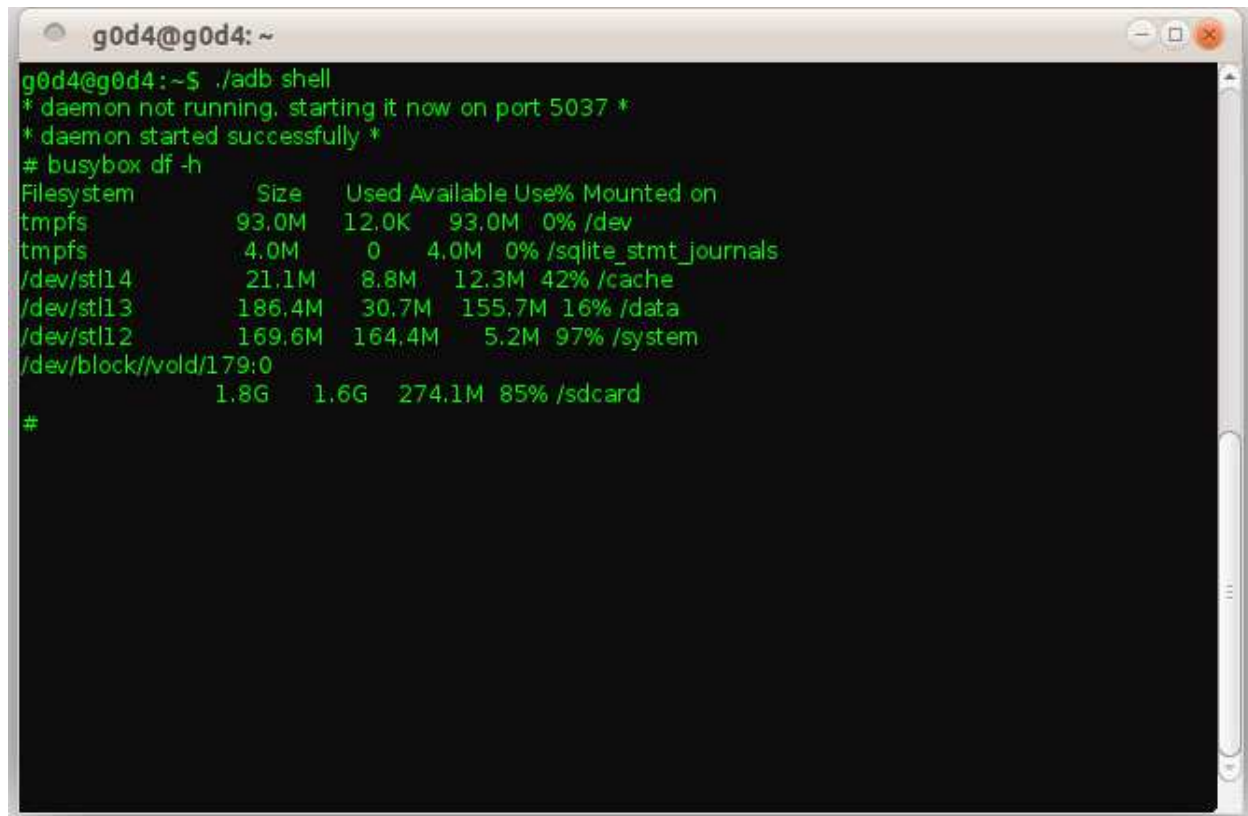
df

I then used busybox in order to verify the results:

busybox df -h

Indeed the results were the same but were shown in a different format. Figure 7 illustrates the results from the busybox command. As it can be seen from the figure Samsung Galaxy Europa has six partitions. Three partitions were related to the cache, system and data which were described in chapter two, section 2.2 and are related with system files, applications, vendor additions, cache files and applications stored on the device. The last partition concerns the external micro SD card that

was used in the phone. Concerning the first two partitions of tmpfs it is unknown how they are used by the Samsung Galaxy Europa. Hence, for the needs of the investigations, the four partitions – cache, data, system and the SD card – will be examined.



```
g0d4@g0d4: ~  
g0d4@g0d4:~$ ./adb shell  
* daemon not running. starting it now on port 5037 *  
* daemon started successfully *  
# busybox df -h  
Filesystem      Size  Used Available Use% Mounted on  
tmpfs           93.0M  12.0K   93.0M   0% /dev  
tmpfs           4.0M    0    4.0M   0% /sqlite_stmt_journals  
/dev/stl14       21.1M   8.8M   12.3M  42% /cache  
/dev/stl13      186.4M  30.7M  155.7M  16% /data  
/dev/stl12      169.6M  164.4M   5.2M  97% /system  
/dev/block/vold/179:0  
                1.8G   1.6G  274.1M  85% /sdcard  
#
```

Figure 7: The partitions of Samsung Galaxy Europa as these were found using busybox

4.1.3 Mounting image files with dd command

The partitions of the Samsung Galaxy Europa are now known and an exact image can be created using the above command. Hence, now a way for mounting these images needed to be found. According to the official Ubuntu documentation website the most common thing that can be mounted on Ubuntu is a hard drive partition. But the part that most interests us is the part where it is stated that mounting is not restricted only to physical drives but a filesystem image can be mounted using a fake device called the “loopback device” (Ubuntu Documentation, 2010).

In order to mount a filesystem which is contained into a file with the help of a loopback device this command must be used:

```
sudo mount Filesystem.img /home/user/MyFileSystem -o loop
```

where Filesystem.img is the filesystem contained in an image file we would like to mount and the /home/user/MyFileSystem is the path where we would like the filesystem to be mounted. The -o loop argument enables the use of the loopback device so that it will be known that we are mounting a filesystem image and not a physical drive. If -o loop is absent the mount command will lead to an error message. It is also noteworthy to state that the mount command needs super user privileges to work (Ubuntu Documentation, 2010).

In order to unmount the filesystem a command more simpler is used:

sudo umount /home/user/MyFileSystem

where the `/home/user/MyFileSystem` is the path where the filesystem image is mounted. For the unmount procedure to work correctly, the filesystem can not be in use when it is unmounted. If it is used it will lead to errors (Ubuntu Documentation, 2010).

4.1.4 Logcat

For the needs of this project it is crucial to look for information regarding the bluetooth and Wi-Fi connections inside the build-in circular logging buffer of Android. As stated above there are many ways for displaying the logs provided by Android and one must be careful when seeking for information. There are three logging buffers in android and can be seen in Table 14.

<i>Buffer</i>	<i>Description</i>
main (default)	The main application log
events	For system event information
radio	For radio and phone-related information

Table 14: Alternative log buffers of Android. This table was copied from Android Developers, Developers' Guide (2011)

Due to the fact that each android log message is associated with a tag and priority, logcat provides the ability to filter the log output. The tag of a log message consists of a short string which illustrates the system component which the log message belongs to, i.e. “View” from the view system. Priority is a character value which is ordered from lowest priority to highest. These are the available values of the priority (Android Developers, Developers' Guide, 2011):

- V: Verbose (lowest priority)
- D: Debug
- I: Info
- W: Warning
- E: Error
- F: Fatal
- S: Silent (highest priority where nothing is printed)

The log output can be reduced if filter expressions are used along with the rest of the command. These expressions provide you the ability to inform the system about the tags and priorities which are interested to you and the system automatically prints only what it was asked to print. The syntax of the expression is:

tag:priority

where tag is the tag of interest and priority holds the minimum level of priority to report for that tag (messages of that tag with priority above the given priority are printed out). There is no limitation on the number of tag:priority specification that can be given in a single filter expression, hence one can provide as many filter expressions as he or she wants as long as they are separated with a whitespace (Android Developers, Developers' Guide, 2011).

Here is an example of a filter expression as it is given by Android Developers, Developers' Guide (2011):

*adb logcat ActivityManager:I MyApp:D *:S*

In this example three filter expressions are used. The first filter expression prints messages that have tag equal to ActivityManager and have priority Info or above. The second filter expression prints messages that have tag equal to MyApp and have priority Debug or above. The last filter expression is used to avoid other log messages since it sets log messages from any tag to have priority Silent which is the highest priority. As stated by Android Developers, Developers' Guide (2011) the *:S filter expression is the best way to ensure that log messages are restricted to the filters that have been specified.

In addition, logcat can provide a formatting ability with the use of some variables. This is achieved with the use of some metadata fields, tags and priority (Android Developers, Developers' Guide, 2011). The variables that are used by logcat to format the logs can be seen in Table 15.

<i>Variable</i>	<i>Description</i>
brief	Display priority/tag and PID of originating process (the default format)
process	Display PID only
tag	Display the priority/tag only
thread	Display process:thread and priority/tag only
raw	Displays the raw log message, with no other metadata fields
time	Display the data, invocation time, priority/tag and PID of the originating process
long	Display all metadata fields and separate messages with blank lines

Table 15: Variables who control the log output format. This table was copied from Android Developers, Developers' Guide (2011)

It is noteworthy to state that only a specific output format can be provided each time using the -V option of the logcat command. In the four scenarios cases it was important to know the time which every process was doing something hence the time variable was used.

4.2 Scenario 1

4.2.1 Implementation

The first scenario as it was described before is about some child pornography photographs that it is believed that they were sent by the suspect's mobile on a laptop PC via bluetooth connection. In order to simulate this scenario I followed the steps which are shown in Figure 6a (page 27). This, as it was stated before, was repeated three times, where three different period of time (30 minutes, 6 hours and 12 hours) were used prior to the investigation of the device.

Six photographs were stored on the smartphone's device and were named:

- 2011-08-12 22.29.31.jpg
- 2011-08-12 22.29.16.jpg
- 2011-08-12 22.29.12.jpg
- 2011-08-12 22.29.07.jpg
- 2011-08-12 22.28.58.jpg
- 2011-08-12 22.28.55.jpg

These photographs were supposed to be sent by the suspect's smartphone device to a laptop. Hence, the photographs were sent over a bluetooth connection to my laptop. Then I used the smartphone device as a regular smartphone device, for example I used some applications, received and answered text messages, accepted phone calls etc for the specified period of time. When the time had passed I followed this procedure:

1. Activate the USB Debugging mode by enabling the option Settings → Applications → Development → USB Debugging
2. Connect a USB cable with the Samsung Galaxy Europa
3. Browse to the folder of the ADB tool and start the ADB server by typing the command:
./adb start-server
4. Copy the logs message of the Samsung Galaxy Europa using the follow commands:

```
./adb logcat -v time -b main > logcat_main.txt  
./adb logcat -v time -b events > logcat_events.txt  
./adb logcat -v time -b radio > logcat_radio.txt.
```

5. Safely remove the SD Card of the smartphone device by going to Settings → SD card and phone storage → Unmount SD card.
6. Insert a new SD card of size at least as big as the Samsung's Galaxy Europa internal memory (Android automatically mounts it)
7. Open a new terminal, browse to the ADB tool's folder and start an ADB shell with the command: *./adb shell*
8. Make an image of the partitions regarding cache, data and system of the Samsung Galaxy Europa with the use of the command:

```
dd if=/dev/stlXX of=/sdcard/NAME.img bs=1024
```

XX represents the numbers 12, 13 and 14 which the partitions belongs to and the NAME represents the name that was given to each image file.

9. Exit from the ADB shell by typing the command *exit*
10. Use the command *./adb pull /sdcard/NAME.img ./PATH* to download the images. NAME represents the name of the image file PATH the path where the image will be downloaded to.
11. Use the command *./adb kill-server* to stop the server and then remove the USB cable
12. Browse to the folder where the image file was downloaded from the device and use the command

```
sudo mount NAME.img /PATH -o loop
```

13.

to mount the image. NAME is the name of the image file that will be mounted and PATH is the path where the image file will be mounted

14. When finished use the command *sudo umount /PATH* where PATH is the path that the image file was mounted.
15. Make a thoroughly investigation of the image file and note every information related to the scenario
16. Repeat steps 12-15 for all the images acquired.

It is noteworthy to state that this procedure, from sending the files to the laptop until step 11 where I unplug the USB cable and use `./adb kill-server` to kill the adb server was repeated three times. One for each occasion of the scenario where the period of time of connecting the smartphone device with the laptop via bluetooth until the investigation was either 30 minutes or 6 hours or 12 hours. Before the repetition of the procedure for each occasion, I used the option Factory data reset, which is located under Settings → Privacy → Factory data reset, to bring the Samsung Galaxy Europa to its original condition. Moreover I formatted the SD card of the smartphone device and the SD card used for the acquisition of the images, using the option located under Settings → SD card and phone storage → Format SD card. Distefano A et al (2010) state that the factory reset is an operation which is available on many mobile devices and provides the ability to restore the device to its factory state. Eventually this operation will erase all the installed applications and all the user data as well.

4.2.2 Investigation and Results

After the simulation of the three occasions of the first scenario there were three log files and three filesystem images for each occasion that needed to be investigated. Firstly, I investigated the three log files that were taken from the logcat tool. Information that was found in the logcat varies for each occasion. In Table 16 it is illustrated if information were found and for what file they were found.

It is vital to state that the information that was found, it was stored in the main buffer of the android logging system whereas the other two buffers radio and events did not hold any information regarding the bluetooth connection. As it can be seen from the Table 16 on the first occasion where I used the smartphone device for 30 minutes before making the examination of the device, the main buffer of the Android's logging system had information for all the files that were sent by the smartphone. In the occasion where 6 hours had passed before making the examination, information was found only for the last file that was sent (2011-08-12 22.28.55.jpg). On the other hand, regarding the occasion where 12 hours had passed before the examination, information was not found for any file that it was sent.

<i>Filename</i>	<i>30 minutes</i>	<i>6 hours</i>	<i>12 hours</i>
2011-08-12 22.29.31.jpg	Yes	No	No
2011-08-12 22.29.16.jpg	Yes	No	No
2011-08-12 22.29.12.jpg	Yes	No	No
2011-08-12 22.29.07.jpg	Yes	No	No
2011-08-12 22.28.58.jpg	Yes	No	No
2011-08-12 22.28.55.jpg	Yes	Yes	No

Table 16: For what files information was found in the Android logging system.

Figure 8 demonstrates the information that was found for each image on the main buffer of the Android logging system. This information was repeated for a every file from the time the sending procedure started until the file was sent. As we can see from Figure 8 *mRequestDirection* indicates that the connection taking place is outbound meaning that the smartphone is sending a file on another device. Information regarding the file it is sent are given by *mFilename*, *mFilenameHint* and *mMimeType* which are about the path, the name and the type of the file. Moreover, *mRemoteAddress* and *mRemoteName* provide the MAC Address and the name of the device that it is accepting the files. In addition, *mCurrentBytes* and *mTotalBytes* provide the bytes that were sent and

the total number of bytes that were sent until the current time and the total number of bytes that need to be sent respectively. Finally, *mStartDateTime* and *mEndDateTime* provide the date that the connection for sending the file was started and the date the connection ended. It is vital to note that this date seems to be in the Unix epoch time, which is basically an integer which counts the seconds passed since Coordinated Universal Time (UTC) of January 1, 1970. The *mStartDateTime* and *mEndDateTime* variables seems to take in consideration not only seconds but the milliseconds as well. Also the *mEndDateTime* has value equal to -1 until the sending procedure has finished. When the sending procedure is done *mEndDateTime* gets an integer value in Unix epoch time format, which represents the date the procedure was finished.

```
08-13 00:29:45.144 D/TimeoutWatchdog( 1351): [Binder Thread #1/5] pet(newTimeout)=
15000
08-13 00:29:45.144 D/RequestQueueHelper( 1351): [Binder Thread #1/5] Updating request
status (content://com.android.bluetooth.opp/btopp/21 with content mId = 21
08-13 00:29:45.144 D/RequestQueueHelper( 1351): mRequestDirection = OUTBOUND
08-13 00:29:45.144 D/RequestQueueHelper( 1351): mRequestType = PUSH
08-13 00:29:45.144 D/RequestQueueHelper( 1351): mRequestId = C1313191781858-1
08-13 00:29:45.144 D/RequestQueueHelper( 1351): mTransactionId = C1313191781858-1
08-13 00:29:45.144 D/RequestQueueHelper( 1351): mUri =
content://media/external/images/media/1
08-13 00:29:45.144 D/RequestQueueHelper( 1351): mFilename =
/sdcard/DCIM/Camera/2011-08-12 22.28.55.jpg
08-13 00:29:45.144 D/RequestQueueHelper( 1351): mFilenameHint = 2011-08-12
22.28.55.jpg
08-13 00:29:45.144 D/RequestQueueHelper( 1351): mMimeType = image/jpeg
08-13 00:29:45.144 D/RequestQueueHelper( 1351): mRemoteAddress = 00:27:13:3A:C1:B0
08-13 00:29:45.144 D/RequestQueueHelper( 1351): mRemoteName = Broadcom Bluetooth
Device
08-13 00:29:45.144 D/RequestQueueHelper( 1351): mStatus = 192
08-13 00:29:45.144 D/RequestQueueHelper( 1351): mCurrentBytes = 165324
08-13 00:29:45.144 D/RequestQueueHelper( 1351): mTotalBytes = 398844
08-13 00:29:45.144 D/RequestQueueHelper( 1351): mErrorCode =
08-13 00:29:45.144 D/RequestQueueHelper( 1351): mErrorInfo =
08-13 00:29:45.144 D/RequestQueueHelper( 1351): mStartDateTime = 1313191782187
08-13 00:29:45.144 D/RequestQueueHelper( 1351): mEndDateTime = -1
08-13 00:29:45.154 D/RequestQueueHelper( 1351): [Binder Thread #1/5] Rows Updated =
1
```

Figure 8: Information on the main buffer of the Android's logging system

After the investigation of the Android's logging system buffers the investigation of the images of the

partitions that were seized from the smartphone device took place. During this investigation the method described in section 4.1.3 “Mounting image files with the dd command” was used, in order to mount and unmount the images of the filesystem partitions and a thorough investigation took place for every image. These investigations had as a result the acquisition of information regarding MAC Addresses of the all paired devices of the smartphone device, the last connected dates of the paired devices and also the names that the paired devices have been using. Hence, no information was found which was about a specific file, but the information that was found was regarding the paired devices of the smartphone and some evidences regarding the date that the connections took place.

It is also noteworthy to state that all of the information that was found did not vary from occasion to occasion. In all three occasions where 30 minutes, 6 hours and 12 hours had passed before the examination all of the information that it is stated below was found with no difference between them. Hence, in this kind of evidences the time that passed does not affect any information regarding the bluetooth connections.

```
00:16:41:99:26:CA 2011-08-12 23:25:33 GMT
00:27:13:3A:C1:B0 2011-08-12 17:42:19 GMT
00:24:EF:14:EC:51 2011-07-30 14:14:46 GMT
5C:17:D3:AA:D5:08 2011-08-12 17:43:25 GMT
BC:47:60:A6:F9:1F 2011-08-12 23:25:07 GMT
```

a) Information on lastseen file

```
00:27:13:3A:C1:B0 Broadcom Bluetooth Device
00:16:41:99:26:CA CHRISTINA
5C:17:D3:AA:D5:08 LG-P500
BC:47:60:A6:F9:1F Galaxy S
```

b) Information on names file

```
00:27:13:3A:C1:B0
00:16:41:99:26:CA 00001108-0000-1000-8000-00805F9B34FB
0000111E-0000-1000-8000-00805F9B34FB
5C:17:D3:AA:D5:08
BC:47:60:A6:F9:1F
```

c) Information on profiles file

Figure 9: Information Found on several files in data partition

After the investigations of the filesystem images of all the occasions, it was noted that not many information is stored on the smartphone device partitions regarding bluetooth connections and the information was the same for all the occasions. The information that is related with the bluetooth connection that took place for sending the photographs to another device was found in the data partition (/dev/stl13) under the path /data/misc/bluetoothd/BC:47:60:K1:P4:06. In this folder there were 6 files regarding bluetooth paired devices named:

- classes
- config
- lastseen
- linkkeys

- names
- profiles

Figure 9 illustrates some screenshots of some of the files that were found. All of the files that were found had the MAC Addresses of the paired devices. The lastseen file is the most interestingly file since it stores information of the last seen date (year, month, day and the accurate time in human readable format). The date provided in the lastseen file, though, is in GMT whereas the time provided in the logcat output is GMT +1:00 which was the timezone the smartphone device was using. Moreover, names file seems to store information regarding the name the device was using when the connection took place.

4.3 Scenario 2

4.3.1 Implementation

The story behind the second scenario is that a man was arrested after some witnesses have seen him leaning over and holding what appeared to be a smartphone below the girl's skirt line. His smartphone was seized. In order to simulate the above scenario I used the steps shown in Figure 6b (page 27). This, as it was stated before, was repeated three times where different period of times (30 minutes, 6 hours and 12 hours) were used prior to the investigation of the smartphone device.

There were nine images already stored on the smartphone device which had no relation to the scenario and were named: 2011-08-15 19.30.49.jpg, 2011-08-15 19.30.52.jpg, 2011-08-15 19.30.54.jpg, 2011-08-15 19.30.58.jpg, 2011-08-15 19.31.01.jpg, 2011-08-15 19.31.17.jpg, 2011-08-15 19.31.28.jpg, 2011-08-15 19.58.34.jpg and 2011-08-15 19.58.38.jpg.

Three images were taken from the smartphone's built in camera and were considered the photographs that were related with the scenario. Since the names of the images were different for the three occasions, since the application camera of the smartphone device automatically names them, three standard names will be used to identify the first, the second and the third image. Hence, the names which are going to be used are:

1. Image 1 will represent the first image
2. Image 2 will represent the second image
3. Image 3 will represent the third image

These images were supposed to be sent by the suspect's smartphone device to a laptop. Hence, the images were sent over a bluetooth connection to my laptop. Then I used the smartphone device as a regular smartphone device, for example I used some applications, received and answered text messages, accepted phone calls etc for the specified period of time. When the time passed, I followed the procedure described in Scenario 1 for the acquisition of the evidences (page 35). This was repeated three times, one for each occasion (30 minutes, 6 hours, 12 hours), from taking the images with the smartphone's built in camera until the investigation.

4.3.2 Investigation and Results

After the simulation of the three occasions of the second scenario there were three log files and three filesystem images for each occasion that needed to be investigated. Firstly, I investigated the three log files that were taken from the logcat tool. Information that was found in the logcat varies for each occasion. In Table 17 it is illustrated if information was found and for what file it was found.

<i>Filename</i>	<i>30 minutes</i>	<i>6 hours</i>	<i>12 hours</i>
Photograph 1	Yes	No	No
Photograph 2	Yes	Yes	No
Photograph 3	Yes	Yes	Yes

Table 17: For which photographs information was found in the Android logging system.

Like the first scenario, information was found only on the main buffer of the android logging system whereas the other two buffers radio and events did not hold any information regarding the bluetooth connection. As it can be seen from the Table 17 on the first occasion where the smartphone device was used for 30 minutes before making the examination of the device, the main buffer of the Android's logging system had information for all the files that were sent through the phone. In the occasion where 6 hours had passed before making the examination, information was found only for the last two files that were sent. Moreover, regarding the occasion where 12 hours had passed before the examination, information was found only for the last file that it was sent. This was surprising, since information was not found for all the photographs in the second situation where 6 hours passed before the examination and information was found for a file in the third situation where 12 hours had passed. This may have happened because the usage of the smartphone device on each occasion was different. It was used as a regular smartphone device during the time before the examination but there was not a specific way of using it. Hence, during the third occasion where 12 hours had to passed the smartphone device may have had a more lightweighted usage when compared to the second occasion where 6 hours had passed. This may also be a drawback of this study and it is discussed in chapter 6 where the critical evaluation is taking place.

Information found in the main buffer of the Android logging system was identical to the first scenario. The information seen in Figure 8 (page 37) of the previous scenario was the same found in this scenario with the same variables indicating the same information: *mRequestDirection* indicates that the connection taking place is outbound meaning that the smartphone is sending a file on another device, *mFilename*, *mFilenameHint* and *mMimeType* are about the path, the name and the type of the file, *mRemoteAddress* and *mRemoteName* provide the MAC Address and the name of the device that it is accepting the files, *mCurrentBytes* and *mTotalBytes* provide the bytes that were send and the total number of bytes that were sent until the current time respectively. *mStartDateTime* and *mEndDateTime* provide the date that the connection for sending the file was started and the date the connection ended in Unix epoch time format.

After the investigations of the android logging buffer the investigation of the image files of all the occasions took place. It was noted that not many information is stored on the smartphone device partitions regarding bluetooth connections. The information was the same for all the occasions and was identical with the information found in the first scenario. The information that is related with the bluetooth connection that took place for sending the photographs to another device was found in the data partition (/dev/stl13) under the path /data/misc/bluetoothd/BC:47:60:K1:P4:06. In this folder there were the 6 files (the same files that were founded for the first scenario) which had information regarding the bluetooth paired devices, like the MAC Address of the paired device, last connected date of the paired device etc.

4.4 Scenario 3

4.4.1 Implementation

This scenario is about a student who has posted some questions from a test on an online forum and

requests the answers. Evidences are needed to verify that the smartphone was connected to the wireless network of the building and that the post was made by his smartphone. In order to simulate this scenario I followed the steps which are shown in Figure 6c (page 27). This, as it was stated before, was repeated three times, where different period of times (30 minutes, 6 hours and 12 hours) were used prior to the smartphone device investigation.

In addition for the needs of this project a forum was created at www.forensicmsc.createaforum.com and a member account was created prior to the implementation of the scenario in order to be used. I again used the procedure described in Scenario 1 (page 35).

4.4.2 Investigation and Results

After the simulation of the three occasions of the third scenario there were three log files and three filesystem images for each occasion that needed to be investigated. Firstly, I investigated the three log files that were taken from the logcat tool. Information was found only inside the main buffer of the android logging system whereas the other two buffers, radio and events, did not hold any information regarding the Wi-Fi connection. Information was found only in the occasion were 30 minutes passed before the acquisition of evidences whereas for the rest of the occasions there was not any information regarding this scenario.

Figure 18 demonstrates the information that was found inside the main buffer of the Android logging system.

```
09-03 13:41:39.718 I/SearchDialog( 1222): Starting (as ourselves)
#Intent;action=android.intent.action.SEARCH;launchFlags=0x10000000;component=com.an
droid.browser/.BrowserActivity;S.query=forensicmsc.ccreateaforum.com;S.user_query=forens
icmsc.createaforum.com;end

09-03 13:41:39.718 I/ActivityManager( 1222): Starting activity: Intent
{ act=android.intent.action.SEARCH flg=0x10000000
cmp=com.android.browser/.BrowserActivity (has extras) }

09-03 13:41:39.728 V/webkit ( 1928): guessURL before queueRequest:
forensicmsc.createaforum.com

09-03 13:43:16.578 I/ActivityManager( 1222): Starting activity: Intent
{ act=android.intent.action.VIEW cat=[android.intent.category.BROWSABLE]
dat=http://forensicmsc.createaforum.com/general-discussion/
cmp=com.android.browser/.BrowserActivity }

09-03 13:43:31.048 I/ActivityManager( 1222): Starting activity: Intent
{ act=android.intent.action.VIEW cat=[android.intent.category.BROWSABLE]
dat=http://forensicmsc.createaforum.com/index.php?action=post;board=1.0
cmp=com.android.browser/.BrowserActivity }
```

Figure 18: Information on the main buffer of the Android's logging system

Basically, what it can be understood is that inside the main buffer of the android logging system, the browsing history can be found. Basically, it states the websites that the browser requested to visit during the time that it was used. Whenever the browser requested a website to visit, this information was showing up in the android logging system.

After the investigations of the android logging buffer the investigation of the filesystem partitions images of all the occasions took place. In these image files, information was found in the /data

partition. To be more specific a file named *wpa_supplicant.conf* was found inside the */data/misc/wifi* folder which stores the names of the remembered wireless networks and their passwords. The passwords were not encrypted and was saved as just text. Figure 10a illustrates the findings inside this file. In addition inside the folder */data/data/com.google.android.server.checkin/databases* an SQLite3 database file exists named *checkin.db*. In order to open this kind of files and examine them, sqlitebrowser tool was used.

According to the Ubuntu Manuals (2007) sqlitebrowser is a visual tool which is able to create, design and edit database files which are compatible with SQLite. It is intentioned to be used in order to create, edit and search data in a spreadsheet-like interface, without knowledge of complicated SQL commands. The *checkin.db* file contains information regarding various states of the smartphone device. Figure 10b demonstrates, among others, when the Wi-Fi was activated and deactivated and provides the date in Unix epoch time format.



a) Screenshot from the *wpa_supplicant.conf* file

	_id	tag	value	date
6	22	NETWORK_DOWN		1315051563210
7	23	NETWORK_UP	NetworkInfo: type: WIFI[], state: CONNECTED/CONNECTED, re	1315051563240
8	24	NETWORK_DOWN		1315052897352
9	25	NETWORK_UP	NetworkInfo: type: MOBILE[UMTS], state: CONNECTED/CONNE	1315052898854
10	26	NETWORK_UP	NetworkInfo: type: WIFI[], state: CONNECTED/CONNECTED, re	1315052909757
11	27	NETWORK_DOWN		1315052910322
12	28	NETWORK_UP	NetworkInfo: type: WIFI[], state: CONNECTED/CONNECTED, re	1315052910381
13	29	NETWORK_DOWN		1315052915904

b) Screenshot from the *checkin.db* file inside the

Figure 10: Screenshots from evidences found in the */data* partition of the android smartphone device

Now when browsing to the folder of the web browser which is located in */data/data/com.android.browser* there was information that might be interesting. All of the rest information described here are found inside this folder. The folder named */cache/webViewCache* has a lot of images from the visited websites that are stored by the browser. These are probably used for

accelerating the speed which the browser loads a website. In addition inside the folder */databases* there are three .db files named *browser.db*, *webView.db* and *webViewCache.db*. In *browser.db* and especially inside the bookmarks table (it sounds strange but it seems that the bookmarks table stores information regarding the websites that were visited and not only for websites that were bookmarked) there is information regarding the visited URLs, title of the page, the number of visits and the last visited data in Unix epoch time format. A screenshot is available at Figure 11a. Moreover, the *webView.db* seems to store information regarding cookies like its value, with which domain is related and when it expires (again in Unix epoch time format) and data regarding websites (for autocomplete reasons). It seems that information stored for autocomplete reasons store only the username and not the password. Details can be seen in Figure 11b and 11c. Finally, information inside the *webViewCache.db* basically correlates the files found in the */cache/webViewCache* folder with the websites that are using those files. The most notable information that is stored in the database file is about the URL of the file, the path, the last modified date in readable format, the date that the file expires in Unix epoch time format and the type of the file. In Figure 11d you can see some results from the *webViewCache.db* file.

It is noteworthy to state that if the user clears the private data stored in the browser, through the settings of the browser, then information that is stored inside the browser folder is deleted.

	_id	title	url	visits	date	created	description
6	6	Soupa Man - Οι τ	http://www.apoel.net/a	1	1315051620598	0	
7	7	Είδηση - Μετρημ	http://www.apoel.net/a	2	1315052048200	0	
8	8	Είδηση - Ανεβάζ	http://www.apoel.net/a	3	1315052852051	0	
9	9	forensic - Index	http://forensicmsc.crea	1	1315053700461	0	
10	10	http://forensicms	http://forensicmsc.crea	2	1315053792527	0	
11	11	forensic - Index	http://forensicmsc.crea	2	1315053792570	0	
12	12	General Discussi	http://forensicmsc.crea	2	1315053870298	0	
13	13	Start new topic	http://forensicmsc.crea	1	1315053811691	0	
14	14	http://forensicms	http://forensicmsc.crea	1	1315053870238	0	

a) Information stored in *browser.db* file

	_id	urlid	name	value
1	1	3	user	forensic
2	2	6	subject	forensic analysis

b) Information found in *webView.db* file

	_id	name	value	domain	path	expires	secure
40	133	MRES	xjyc3Peq1tiolo9W	www.google.co	/m	1317644925000	0
41	135	mc	4e62209c-6a896-	.quantserve.cor	/	1472906524000	0
42	137	_qca	P0-306434213-13	.createaforum.c	/	2145916800000	0
43	138	vglnk.Agent.p	0461af69479e73f	.viglink.com	/	1630413725000	0
44	139	ucid	MXXnF/2kALcqEFv	.gigya.com	/	1630672926000	0
45	141	PHPSESSID	1b2b7cbb428554	forensicmsc.cre	/		0
46	142	SMFforensicmsc	a%3A4%3A%7Bi%	forensicmsc.cre	/	1504269739000	0
47	143	UID	68bd9818-92.123	.scorecardresea	/	1378125902000	0

c) Information found in *webView.db* file

	_id	url	filepath	lastmodify	etag	expires	mimetype
155	219	http://www.google	c2bfa713			0	text/javascript
156	220	http://forensicmsc	424e01a2	Sat, 03 Sep 20		0	text/html
157	221	http://forensicmsc	66de6394	Fri, 24 Jun 201	"10000000a!	1316288586442	text/css
158	222	http://images.smf	45d77b6d	Sat, 02 Jul 201	"4291ca00b7	1316140420257	application/x-jav
159	223	http://cdn.gigya.c	cec7d542	Fri, 02 Sep 20		1315054600867	text/javascript
160	224	http://images.smf	f6c7b9e1	Sat, 02 Jul 201	"aaa650f124	1316140420574	application/x-jav
161	225	http://forensicmsc	f31d574d	Wed, 29 Jun 2	"10000000a!	1316201940234	text/css
162	226	http://edge.quant	43d53b85	Thu, 30 Sep 2		1315140102037	application/x-jav
163	227	http://images.smf	187348bd	Sat, 02 Jul 201	"bf58b77335	1316140421926	application/x-jav

d) Information found in webViewCache.db file

Figure 11: Information found inside the folder of the browser

4.5 Scenario 4

4.5.1 Implementation

This scenario is about a man who is believed he is uploading child pornography images from his neighbour's unsecured Wi-Fi network via his smartphone. He is using his dropbox application to upload these images. In order to simulate this scenario I followed the steps which are shown in Figure 6d (page 27). This, as it was stated before, was repeated three times, where different period of times were used (30 minutes, 6 hours and 12 hours) prior to the investigation of the smartphone device.

Two images were stored on the smartphone's device and were the files that were considered vital for the investigation:

- 2011-09-03 16.02.25.jpg
- 2011-09-03 16.02.29.jpg

These images were supposed to be uploaded by the suspect's smartphone device via the official dropbox application over an unsecured Wi-Fi network. Hence, I downloaded and installed the official dropbox application on the smartphone device, created an account and uploaded some files. Then at different time I uploaded the two images stated above using an unsecured Wi-Fi network. As with the rest of the scenarios, I used the smartphone device as a regular smartphone device, for example I used some applications, received and answered text messages, accepted phone calls etc for a specified period of time. When the time passed I followed the procedure described in Scenario 1 (page 35) for the acquisition of the evidences three times, one for each occasion (30 minutes, 6 hours, 12 hours).

4.5.2 Investigation and Results

After the simulation of the three occasions of the fourth scenario there were three log files and three filesystem images for each occasion that needed to be investigated. Firstly, I investigated the three log files that were taken from the logcat tool. Information was found only on the main buffer of the android logging system whereas the other two buffers, radio and events, did not hold any information regarding the Wi-Fi connection. As it can be seen from Table 19 information was found

for all images in the occasion were 30 minutes and 6 hours passed before the acquisition of evidences whereas for the occasion where 12 hours information was found only for the last file.

<i>Filename</i>	<i>30 minutes</i>	<i>6 hours</i>	<i>12 hours</i>
2011-09-03 16.02.25.jpg	Yes	Yes	No
2011-09-03 16.02.29.jpg	Yes	Yes	Yes

Table 19: For what photographs information was found in the Android logging system.

Figure 12 demonstrates the information that was found for each photograph on the main buffer of the Android logging system.

```

09-03 16:03:28.148 E/com.dropbox.android.provider.DropboxProvider( 7516): Adding new
file (from import, probably): content://media/external/images/media/815
09-03 16:03:28.158 I/com.dropbox.android.taskqueue.DbTaskQueue( 7516): Task
content://media/external/images/media/815~/ adding to task DB
09-03 16:03:28.368 I/com.dropbox.android.service.DropboxNetworkReceiver( 7516): Setting
receiver enabled: true
09-03 16:03:28.378 I/com.dropbox.android.service.DropboxService( 7516): Dropbox service
has been started
09-03 16:03:28.378 I/com.dropbox.android.taskqueue.UploadTask( 7516): Uploading file
from URI: content://media/external/images/media/814
09-03 16:03:28.388 D/KeyguardViewMediator( 1222): setHidden false
09-03 16:03:28.578 D/StatusBar( 1222): makeNotificationView :
NotificationData(package=com.dropbox.android tickerText=Uploading file 1 of 2
ongoingEvent=true contentIntent=PendingIntent{44d3e168:
PendingIntentRecord{44dbd050 com.dropbox.android startActivity}} deleteIntent=null
clearable=false contentView=android.widget.RemoteViews@44ce1d88
when=1315062208550 twQuickPanelEvent=0 twQuickPanelDescription=null
missedCount=0 contact=null)
09-03 16:03:28.598 I/com.dropbox.android.taskqueue.UploadTask( 7516): Uploading uri
content://media/external/images/media/814 to / as 2011-09-03 16.02.25.jpg
09-03 16:03:28.698 D/StatusBar( 1222): updateExpandedViewPos before
expandedPosition=-10000 mTrackingParams.y=-301 mTrackingPosition=-301
09-03 16:03:28.738 D/KeyguardViewMediator( 1222): setHidden false

```

Figure 12: Information on the main buffer of the Android's logging system

As it can be seen from Figure 12 the main buffer of the android logging system stores information regarding the files that it were uploaded. The name of the file that it is uploaded can be found with the exact date and time (see the log at 09-03 16:03:28.598). Moreover, it provides information on how many files are being uploaded (see log at 09-03 16:03:28.578).

After the investigation of the android logging buffer the investigation of the filesystem image files of all the occasions took place. In these images, information was found in the /data partition. To be more specific identical information like scenario 3 was found – *wpa_supplicant.conf* file and the *checkin.db* file – with the difference that inside the *wpa_supplicant.conf* there was not any password related to the network since it was unsecured.

Now when browsing to the folder of the official dropbox application which is located in */data/data/com.dropbox.android* there was information that might be interesting. Basically, there is a file named *db.db* which stores information regarding the files that are stored on the dropbox account. Figure 13 demonstrates the information that were found in this file. As it can be seen the information concerns the last modified date of the file on the dropbox account, the size of each file, the path where it is stored, its type and name.

	id	_da	modified	bytes	revision	hash	icon
13	13		Sat, 03 Sep 2011 14:07:45 +0000	717269	c03c08952		page_white_pic
14	14		Sat, 03 Sep 2011 14:09:25 +0000	502763	d03c08952		page_white_pic
15	15		Sat, 03 Sep 2011 15:04:09 +0000	472571	e03c08952		page_white_pic
16	16		Sat, 03 Sep 2011 15:04:13 +0000	469478	f03c08952		page_white_pic

a) Information found in *db.db* file about the size and the last modified date

	is_dir	path	canon_path	root	size	mime_type
13	0	/IMG825.jpg	/img825.jpg	dropbox	700.5KB	image/jpeg
14	0	/IMG824.jpg	/img824.jpg	dropbox	491KB	image/jpeg
15	0	/2011-09-03 16.02.25.jpg	/2011-09-03 16.02.25.jpg	dropbox	461.5KB	image/jpeg
16	0	/2011-09-03 16.02.29.jpg	/2011-09-03 16.02.29.jpg	dropbox	458.5KB	image/jpeg

b) Information found in *db.db* file about the path, the size and the type

	thumb_exists	parent_path	canon_parent_p	_display_name
13	1	/	/	IMG825.jpg
14	1	/	/	IMG824.jpg
15	1	/	/	2011-09-03 16.02.25.jpg
16	1	/	/	2011-09-03 16.02.29.jpg

c) Information found in *db.db* file about the size and the last modified date

Figure 13: Information inside the *db.db* file of the dropbox application

4.6 Analysis of the results

After the examination of the scenarios, the results needed to be analysed and investigate what they can demonstrate regarding a bluetooth or a Wi-Fi connection. Concerning the first two scenarios which were about bluetooth connections between the smartphone device and other devices two kind of evidences were found as demonstrated before, the logs from the android logging system and the files inside the */data* partition.

Regarding the logs of the Android system, it seems that they provide a better point of view regarding the bluetooth connections that took place, since they provide information like when the connection started, when the connection ended, what it was sent during this connection between the two devices, name and MAC address of the device that the files are being sent to. The disadvantage of the evidences found in the logging system is that due to the fact that the logs are stored in a circular buffer they do not seem to have a long lifetime. According to the results of the four scenarios that were implemented for the needs of this study, it seems that the logging system keeps information over the last six hours. This period of time must not be taken for granted, since the smartphone device was used as regular smartphone until the examination for evidences. Hence, it

may be possible that a more exhaustive use of the smartphone device for example using the smartphone device to share the smartphone's data connection with another device may have lead the period of time which information is stored inside the android's logging buffer to be reduced. Hence, although the logs from the android logging system provide most detailed evidences regarding the connections, it seems that they do not have a long life time even if the smartphone device is left idle (If the smartphone is left idle the logs are updated with network and battery updates which do not affect the logs in a big extent, but they do affect them).

An important detail was discovered when investigating the filesystem images of scenarios 1 and 2 regarding the lastseen file and the date details stored on it. It seems that the date stored on the lastseen file changes only when a scan for devices procedure takes place. To be more precise lets assume two situations as seen in Figure 14. For the two situations, we are assuming that the smartphone device was paired with the device A with MAC Address 11:22:33:44:55:66 in Friday 16th of September at 10:00 AM. Now, think of a user who wants to find a file and then share it via bluetooth. Now, there are two options, either scan for devices or choose one from the existing paired devices to sent to and this is where the two situations differ. In the first situation the user chooses to scan for devices then chooses from the list of the available devices, the device A. In the second situation the user chooses to sent the file to the device A from the list of the paired devices without scanning for devices. In both situations the file is sent to device A.

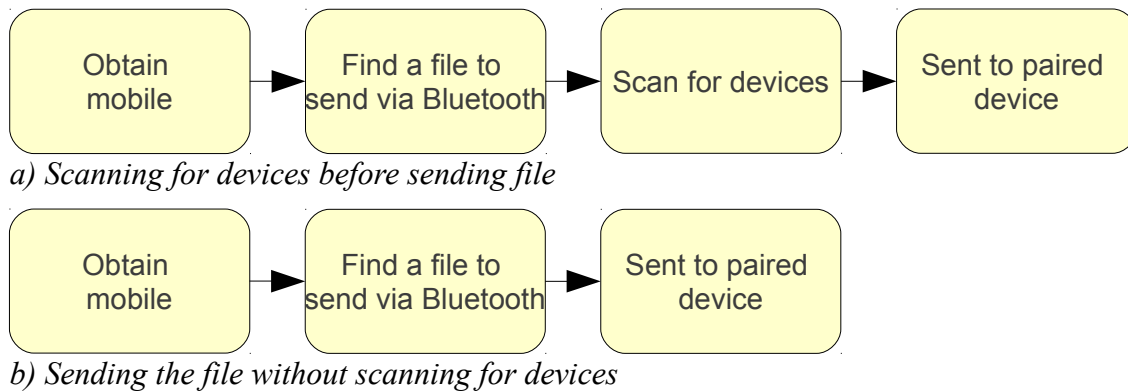


Figure 14: Two situations for sending files to paired devices

After the implementation of the two situations, it is expected that the last seen date of the lastseen file will be the date that the file was sent to device A. But, Android does not seem to work like this. It seems that in the first situation where the user scans for devices prior of sending the file, the date which is stored in the lastseen file (Friday 16th of September at 10:00AM in our example) is overwritten with the new date. In the second situation where the user just sends the file to the paired device A and does not scan for new devices, the date written in the lastseen file remains Friday 16th of September at 10:00AM. This was tested a few times with different files and still got the same results. Hence, it can be concluded that in order to change the last seen date inside the lastseen file, a scan for devices procedure must take place.

In contrast with the information found in the logs of the Android, information found in the data partition do not have a specific life time, but do not provide many information regarding the bluetooth connections of the device. The only vital information that they provide, is the MAC address of the paired devices and the last seen date of that paired device. But as it was described above there is a possibility of false information stored in this file. Hence, although there might have been a connection between the smartphone device and another device the last seen variable may indicate a false last connection date.

To sum up, the amount of time between the time the connection took place and the examination must be consider crucial when it comes to bluetooth connections. If the logs of the android has

overwritten the information of the bluetooth connections it may be impossible to prove that a connection took place between the smartphone device and another device. If information stored in the /data partition is used it will be possible to prove that a connection took place, but the exact date of the last connection with the specific device will be correlated with the method the user used to send the file (whether the user scanned for devices or not) which does not always provide correct information.

Concerning the third scenario where the official browser was used to post something on a forum, information regarding this connection was found in the android logging system, the package of the browser inside the /data partition and a folder named /misc/wifi about the Wi-Fi settings again inside the /data partition.

It seems that the android logging system in contrast with the scenarios regarding bluetooth connections does not hold too much information regarding the use of the browser. As it can be seen from the results I found from this scenario, information inside the logging system is more about the browsing history of the browser, what websites the browser requested to open and when the request was done. Moreover, this information on the logging system is again depended on the period of time that has passed before the examination of the smartphone device. Finally, it seems that the logging system does not provide more information regarding the Wi-Fi connection when compared to the information that is found on the folder of the browser inside the /data partition.

When it comes to the information found inside the folder of the browser of the /data partition it seems that they have a lot of evidences. The sqlite databases hold a lot of information like all of the browsing history, the number of times the smartphone has been used to visit a website and the specific date with time (again in Unix epoch time). In addition, they store some images in order to load more quickly the various websites that have been visited by the user of the smartphone device. If found during a forensic investigation, this kind of information can indicate whether the smartphone device has been used to browse to these websites or not. It is vital to note that these sqlite databases store this information not only for a specific period of time but either forever or until the user chooses to delete his or her browsing history through the browser's settings.

In addition, there was another file found during the investigations named wpa_supplicant.conf which stores information regarding the Wi-Fi networks that the smartphone device was connected to. Basically it stores the name of the wireless network and the password related to that network. Not many conclusions can be made regarding this information due to the fact that a lot of wireless networks can have the same name and the password used for the encryption can be changed. Hence, if it is found that a smartphone device was connected on a wireless network named "A" which was encrypted under the password "ABCDEFGH" may not necessarily mean that the smartphone device was connected to a specific wireless network named with the same name and having the same password.

In contrast with the bluetooth connection scenarios, android logging system does not hold too much information regarding the Wi-Fi connection when it is used by the official android browser. The information found in the android logging system can also be found in the browser's folder in the /data partition except if the user has cleared his browsing history. In addition, the logs regarding the wireless state (this is not the same with the android logging system) is also vital, since it provides information regarding when the wireless was activated and deactivated on the smartphone device but do not provide information on what network it was connected and for how long.

Regarding the last scenario where the official dropbox application was used, information regarding the wireless connections was also found in android logging system as this was described in the previous section. Again the information that were found in the logging system provided information regarding the name of the uploading files and how many of the uploading files were uploaded. Hence, the exact time of the wireless connection can be provided and the name of the file that was

sent. Again this information depends on the time between the connection was made and the time the forensic examination took place. Again, the log state was found for the Wi-Fi state which indicates when the Wi-Fi was activated or deactivated.

Information about this scenario was also found in the folder of the dropbox application. Information was found regarding the files that were on the suspect's dropbox account. Specifically for each file there was stored its location inside the dropbox folder, the size of the file, type, the last modified date etc. In this scenario, the information found on both the android logging system and the application folder is important. The information found on the android logging system can prove that the files were uploaded from the smartphone device in contrast with the information found on the dropbox's folder where it can be just proved that the files are indeed inside the dropbox's folder. Since the information on the android logging system is time depended in contrast with the rest of the information, it is vital for an investigation related to the dropbox application to have all the kind of informations that it can get. This is because it can be easily understood that a combination of the information of the android's logging system and of the information found on the dropbox's application folder is more acceptable rather than just information from the dropbox application which just states that the files are stored on the dropbox's folder, since these could be uploaded from any device and not the specific android device.

Concerning all of the scenarios, the cache.cell and cache.wifi files were founded as stated by Arthur C. (2011) and were parsed with the android-locdump tool. Cache.cell provided some GPS location data which were not so accurate, probably they were indicating a position captured by cell tower triangulation whereas cache.wifi file did not provide any data.

5. Methodology for acquiring evidences regarding bluetooth and Wi-Fi connections

5.1 Definition

After the completion of the investigation and the analysis of the results the methodology for acquiring evidences regarding bluetooth and Wi-Fi connections had to be determined, as its the outcome of this research. Taking into consideration the evidences that were found for each scenario a methodology is defined below and will be evaluated in the next section with the use of the ACPO's Guidelines for good forensic practice. This method must be strictly followed and implemented fast but without making any mistakes, since as it was demonstrated by the investigation of the scenarios, time may be vital in this kind of examination. In addition, the machine that the examination will be taking place must be running Ubuntu 11.04 since this was the operating system that the method was tested and used during the investigation of the scenarios. It can theoretically be used in more Linux distributions with the same results but as the methodology was not tested, it is not recommended.

In addition, the methodology is divided in three parts. Basically the first part is about the seizure of the smartphone device and what steps have to be followed in order to preserve the evidences. The second part concerns the acquisition of evidences regarding the android logging system whereas the third part is about the acquisition of evidences from the internal memory storage partition of the Android system. It is vital to strictly implement these steps in order. It is also noteworthy to state that this method requires the Android device that it is being investigated to be able to accept micro SD card. In the situation where a smartphone device allows a part of its internal memory to be mounted as SD card, this method may theoretically work, but it was not tested in the context of this study due to the fact that a smartphone device which did not have an SD card slot was not provided.

Below is the first part:

1. Make sure that all the actions that are being made during the investigation are recorded. Date and time must be clearly marked on the record.
2. Obtain the mobile device, find the micro SD slot of the phone and check whether the smartphone device has any micro SD card.
3. Activate the airplane mode which turns off all the wireless connections of the smartphone device, either that is Wi-Fi, bluetooth, infrared or mobile networking.
4. Activate the USB Debugging mode (located in Settings → Applications → Development → USB Debugging)

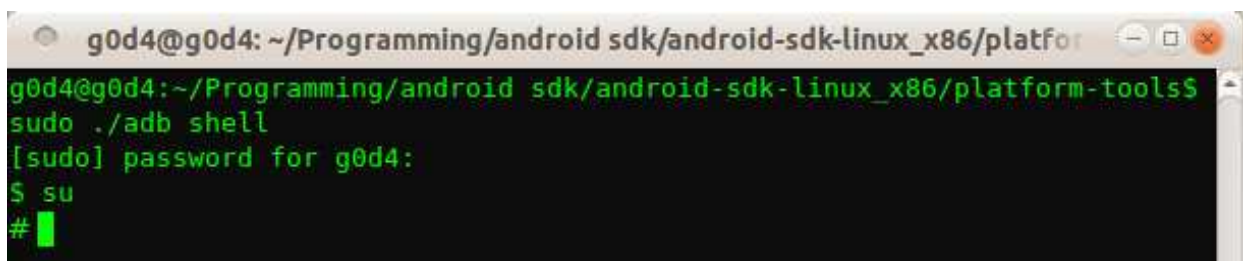
Once the smartphone device has been taken to the examination unit the second part of the investigation must start. It is advised to make the examination of the next step as quickly as possible. Remember that the examination unit PC must be running Ubuntu 11.04 and have access to the android SDK and specifically platform tools where the adb tool is located.

Below is the second part of the investigation.

5. Connect the USB Cable with the smartphone device and with the machine that the investigation will be held.
6. Browse to the folder where ADB tool is and start the ADB server by typing the command:
./adb start-server
7. Use the logcat tool to copy the logs of the Android smartphone device which are formatted to show the time using the command *./adb logcat -v -time -b main > FILE_OF_YOUR_DESIRE.txt*

This is where the second part finishes and part three begins. Part three is about acquiring evidences from the internal memory storage of the smartphone device. Below is part three:

8. Turn off the device, remove any SIM card or SD card and send them for further examination (if this is needed)
9. Install a new formatted SD card in the smartphone device and boot up the device (in the occasion where the smartphone device does not have an SD card slot this step should be ignored)
10. Gain root access on the smartphone device (A discussion regarding this step is done in the evaluation of the methodology)
11. Repeat steps 5 - 6 in order to start the adb server.
12. Browse to the folder where ADB tool is and start an ADB shell with the command: `sudo ./adb shell` (this must be run with super user privileges. In the study of Lessard J. and Kessler C G. (2010) the `sudo` is not required but in my occasion it was necessary to run the adb shell with super user privileges in order to gain super user access on the Android device)
13. If the device is using the Memory Technology Device partitions use the command `cat /proc/mtd` to find out the partitions structure. If the smartphone device does not use the Memory Technology Device system then use the `df` command to identify the partitions.
14. When the shell has been established use the command `su` to gain root access (# indicates that the shell is running in super user access as seen in Figure 15)



```
g0d4@g0d4: ~/Programming/android sdk/android-sdk-linux_x86/platform-tools$
g0d4@g0d4:~/Programming/android sdk/android-sdk-linux_x86/platform-tools$
sudo ./adb shell
[sudo] password for g0d4:
$ su
#
```

Figure 15: Hash (#) indicates root access on the adb shell

15. Use the command

`dd if=/XX/YY of=/SDCARD/NAME.img bs=1024`

to image the data partition. /XX/YY represents the partition that will be copied from the smartphone device, 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.

16. If there is any error while executing the `dd` command use the method described by (Linux Sleuthing, 2011) at the end of section 4.1.1 ADB Tool to make an image of the partition.
17. When the partition have been successfully imaged exit the adb shell by typing the command `exit` twice. One to exit from the `su` access and a second one to exit the adb shell.
18. Use the command `./adb pull /SDCARD/NAME.img ./PATH_TO_DOWNLOAD` to download the image that has been created, where SDCARD is the path on the smartphone device that the image file was saved, NAME is the name of the image and `PATH_TO_DOWNLOAD` is the desirable path to store the image file on the computer.
19. Use the command `./adb kill-server` to stop the server
20. Remove the USB cable.

21. Browse to the folder where the image file was downloaded from the device and use the command `sudo mount NAME.img /PATH -o loop` to mount the image. NAME is the name of the image file that will be mounted and PATH is the path where the image file will be mounted
22. Find the appropriate files that best suit the case and investigate the evidences found on each one of them. Files according to the scenarios can be found in Table 20
23. When finished use the command `sudo umount /PATH` where PATH is the path that the image file was mounted to unmount the image file.

This is the end of the methodology. At Table 20 there are the files that were founded for each scenario and must be investigated when examining a smartphone device regarding bluetooth and Wi-Fi connections.

<i>Name</i>	<i>Path</i>	<i>Relation</i>
Logcat main buffer	-	Bluetooth and Wi-Fi
classes, config, lastseen, linkkeys, names, profiles	/misc/bluetoothd/MAC_ADDRESS where MAC_ADDRESS is the mac address for the bluetooth on the smartphone device	Bluetooth
wpa_supplicant.conf	/misc/wifi	Wi-Fi
checkin.db	/data/com.google.android.server.checkin/database	Wi-Fi
browser.db, webView.db, webViewCache.db	/data/com.android.browser/database	Wi-Fi Browser
db.db	/data/com.dropbox.android/databases	Wi-Fi Dropbox
cache.wifi, cache.cell	/data/com.google.android.location/files	Bluetooth and Wi-Fi

Table 20: Files which need to be investigated when examining for bluetooth and Wi-Fi connections

It also noteworthy to state that the use of MD5 cryptographic hash functions was considered in order to verify that the partitions are exact copies of the partitions on the smartphone device but as it was stated by Owen P. and Thomas P. (2011) if a cryptographic hash is obtained by a mobile device it will change when the mobile device change its state, for example when the date and time is changed. Hence it will be infeasible to use this technique. An attempt was made to use md5sum which according to Ubuntu Documentation (2011) implements the MD5 (Message-Digest algorithm 5) 128-bit cryptographic hash. This command is available through busybox using the command

busybox md5sum INPUT

where INPUT is the name of the file. This command was used three consecutive times and used the Android's data partition as input with 1 minute difference between the commands and with no other interaction on the smartphone device. This resulted with 2 same hash values and a third different one as it can be seen in Figure 16. This basically verifies what it was stated by Owen P. and Thomas P. (2011) that it is infeasible to use a cryptographic hash function to verify that the partition obtained from the smartphone device is an exact copy of the original one, due to the constantly changing state of the original partition. It is illustrated that in a period of maximum two minutes the smartphone device changed its condition.


```
# busybox md5sum /dev/stl13
f4004d774e33bd1144876fc4f5ea95bf /dev/stl13
# busybox md5sum /dev/stl13
f4004d774e33bd1144876fc4f5ea95bf /dev/stl13
# busybox md5sum /dev/stl13
cdd0fe5ceb540a7e2edfca24c6da0eee /dev/stl13
```

Figure 16: Using md5sum on /dev/stl13 (data partition of the Samsung Galaxy Europa)

5.2 Evaluation of the methodology

The above methodology which was a result of the scenarios examinations must be evaluated. This will be done as stated before with the use of the ACPO's guidelines for good forensic practice. In order to make this evaluation, each principle, of the four contained in the guidelines, will be stated and an evaluation regarding that principle will be made. These guidelines have a special section which specialises on mobile forensics. Specific notes that can be founded for each principle on how it can be particularly used on mobile devices forensics will be seriously taken into account.

“Principle 1: No action taken by law enforcement agencies or their agents should change data held on a computer or storage media which may subsequently be relied upon in court.” (ACPO, 2008:4)

This is a more general principle which is divided by the ACPO Guidelines in various parts. The first part is about the seizure and preservation of evidence . ACPO (2008) provide two techniques that can be used as a solution for this problem:

1. Turn the smartphone device off at the point of seizure
2. Place the smartphone device in a shielded container/bag

Each technique has its pros and cons, but according to our situation where information regarding bluetooth and Wi-Fi connections needs to be acquired a technique must be chosen wisely.

The main disadvantage of the first solution is that it actually requires the smartphone to be turned off. But for the needs of this methodology where the Android logging system has some crucial information stored on it, turning off the device alters the information that is being stored. In addition, you can not acquire the evidences that are stored in the logging system when the smartphone device is off, hence it will be needed to be turned on again which will again alter to some extent the android logging system. It is easily understood that the first solution is not suitable for the situation where evidences are found in the android logging system.

Regarding the second solution proposed by ACPO Guidelines there are three main disadvantages. First of all placing the mobile device in a shielded container or bag will take away the ability of interacting with it which is something crucial for the needs of this methodology. This is because, a lot of Android smartphone devices do not have buttons for interacting with the device but instead they have touch buttons. Touchscreens and touch buttons are difficult to handle when they are inside a bag hence it will be not easy for the examination to take place. Another disadvantage is that when a smartphone device is placed in a shielded bag or a container the smartphone increases the battery consumption since it is trying to connect to a network. When the battery is decreasing the android logging system is updated and this may overwrite information that is stored inside the logging buffer. Moreover, the smartphone device may ran out of battery in its try to connect to a network since this an exhaustive and battery consuming procedure. Moreover, the methodology that was

defined above requires a connection of a USB cable. If placed in a shielded container/bag the smartphone device will not be able to be connected with a USB Cable.

Hence the technique that will be used, will not require the smartphone device to be turned off neither to be kept in a shielded bag or container. As stated in step 2 of the methodology, the airplane mode will be activated. As stated by Mislan, R.P., Casey, E., and Kessler, G.C. (2010) in order to isolate the evidential smartphone device from the networks to prevent any communication with telecommunications, Wi-Fi and bluetooth networks the airplane mode of the smartphone device can be activated. This is a drawback of the methodology since the technique of activating the airplane mode is not stated in ACPO guidelines as a technique which preserves the evidences but it is only supported by the above researchers.

ACPO (2008) also state in the terms of Principle 1 that the smartphone device must be isolated from network during the examination procedure and states five different techniques:

1. Use a jamming device
2. Use a shielded room
3. Use a shielded container/box
4. Use an “access card” type SIM that will mimic the identity of the original SIM card and will not allow network access
5. Request that service provider disable the subscriber account

All of the above techniques that are provided by ACPO again have their advantages and disadvantages. Concerning the first technique, using a jamming device except from the fact that it may interfere with network coverage near the examination area it is also illegal in many countries as stated by ACPO (2008). Shielded room is not a good idea due to the mobility factor. It basically requires the smartphone device to be inside this room for the examination, hence the smartphone device needs to be moved to the shielded room when acquired. A shielded room may not be in a close distance, so time will be wasted while transferring the smartphone device to the shielded room and as it was noted before, time in the occasion of bluetooth and Wi-Fi connections is crucial. The disadvantages of the third technique were also discussed above and the reasons for not using this technique were explained above. Moreover, the use of a SIM card which will mimic the original one is again not recommended due to the fact that most of the mobiles require a battery removal which leads to turning off the device. As it was described before, this is consider a big disadvantage due to the addition of information inside the android logging system. The last solution of disabling the subscriber account as stated by ACPO (2008) is more theoretical since as it is noted it was not tested before and nevertheless it requires the intervention of the service provider who may not be willing to co-operate.

Hence, the technique that was used in the methodology, is the technique that was described before, activating the airplane mode on the smartphone device. This is again a drawback of the methodology with the same reasons which were stated above.

In addition, again for the needs of the Principle 1 of the guidelines, ACPO (2008) state that the software that is used during the investigation must be designed for forensic use. If a tool which was not designed for forensic use is going to be used it must have been tested before under a safe environment with same device so that their operation and effects are known to the investigator. ADB and busybox was not developed as a forensic tools, but they were recommended by researchers (Hoog A., 2009 and Linux Sleuthing, 2011) which had already tested them in their studies. Hence, a continuous research on how these tools work on various devices must be taking place in order to make sure about the effects of these tools on the devices.

Moreover, Principle 1 requires as noted in the ACPO Guidelines to use a secure reliable connection interface which minimises data change on the device. Four ways of connection interface are introduced and include:

1. Cable
2. Infra-red
3. Bluetooth
4. Wi-Fi.

As stated before, infra-red is becoming obsolete in new smartphone devices. Nevertheless, ACPO (2008) state that cable is secure, reliable and has the least impact on the device whereas Infra-red is less secure and reliable. In addition Bluetooth is considered to be the least secure and Wi-Fi is still not available but in the near future this may be possible. The methodology which was defined on this study uses a USB Cable which is the best of the methods that was introduced by ACPO (2008).

ACPO (2008) also note that the examination process must be planned in a way that it avoids the loss of data which is vital for the case. ACPO (2008) also states that the sequence of examination may depend on various factors like case specific subjects (for example attention must be given to date and time) and may lead to data loss. The above methodology was designed in a way that the vital and more crucial data that are subject to deletion, like the information stored in the android circular buffer, must be taken into consideration and investigated first. In contrast to the rest of the information which is not so vulnerable to deletion.

In addition ACPO (2008) states two last notes for Principle 1. First of all it states that the investigators have to be familiarised with the smartphone their investigating before they start the investigation. This statement is addressed to the investigators. The methodology that was presented here can not make sure that it will be followed by a trained person hence the man in charge for the examination must be sure that the person who is making the investigation is trained. Secondly, the ACPO Guidelines state that all of the examinations done in mobile phones have some manual examination like navigating through a menu of the phone. This can also be seen in this methodology at steps 3 and 4 of the methodology, where some functionalities of the smartphone device are activated (Airplane mode and USB debugging mode).

Principle 2 of the ACPO Guidelines state that *“In circumstances where a person finds it necessary to access original data held on a computer or on storage media, that person must be competent to do so and be able to give evidence explaining the relevance and the implications of their actions”* (ACPO, 2008:4).

This principle is the most debatable, concerning the ACPO guidelines and how they are implemented in the methodology. In step 10 of the methodology it is stated that in order to continue the examination of the smartphone device the investigator must gain root access on the smartphone device. The method for gaining root access in the four scenarios that were investigated has been done using the SuperOneClick application v1.9.5. This method it is not tailored as a forensically sound one since it was not created to be used in a forensic investigation. The above method was used in the four scenarios as a proof of concept, meaning that if the rooting procedure is done using a forensically sound way then the above evidences can be found.

There is a company named Oxygen Forensic that has published an Android rooting add-on for their forensic suite, Oxygen Forensic Suite, and provides root access on Android smartphones versioned 2.0 – 2.2 but it is not guaranteed that it will work 100% on all smartphone devices. On its official website Oxygen Forensic (2011) does not state if this method implies with any rules of good forensic practice. Hence, it is unknown what implications this rooting procedure has on an Android smartphone device.

In addition Hoog A (2011) illustrates that there major challenges when obtaining root access to the smartphone:

1. Gaining root access may change the device in many situations.
2. Various techniques exist for rooting android smartphone devices and this is because Android is an open source project. Each manufacturer may choose to alter the code, use different version of the Linux kernel etc. Hence, there are various ways of rooting a device depending on its model, version etc.
3. Many of the techniques used for acquiring root access may not work and have inaccurate information.

Hoog A. (2011) also states that there are three types of root access:

1. Temporary root access which is achieved using an exploit which permits to have root access until the device is rebooted
2. Full root access which is achieved using a persistent root exploit.
3. Recovery mode³ root access which is gained with the use of flashing a custom recovery partition.

He also adds that a forensic investigator will be interested in gaining temporary root access with temporary root access or recovery mode root access.

It can be understood that it is there is not a method of rooting an Android device which is forensically sound correct hence a method for gaining root access must be chosen with extra caution. As long as there is no way for gaining root access in a forensically sound of manner or at least there is not a published way for rooting the android in a forensically sound of mane(due to competition, Oxygen Forensics) step 10 of the above methodology violates Principle 2 of the ACPO Guidelines.

Regarding the Principle 3 of the ACPO Guidelines which states that “*An audit trail or other record of all processes applied to computer-based electronic evidence should be created and preserved. An independent third party should be able to examine those processes and achieve the same result*” (ACPO, 2008:4).

The defined methodology does not violate this principle since step 1 of the methodology clearly states that all actions that are being made during the forensic examination are being recorded. In addition, ACPO Guidelines note that if there is any change of the data which may occur that change must be noted. The only step that will not achieve the same result if it is repeated is part 2 of the methodology which acquires the information found in the android logging buffer. As stated before, this is a circular buffer, and overwrites information and can not be stopped from doing so. Hence, there is no way of preventing the buffer from changing its condition. This must be clearly be noted on the report of the investigation.

Lastly Principle 4 states “*The person in charge of the investigation (the case officer) has overall responsibility for ensuring that the law and these principles are adhered to*” (ACPO, 2008:4).

ACPO (2008) also states that the examiner must suggest a strategy which is the most appropriate for the needs of each case. The defined methodology has achieved these by dividing the methodology in three parts. Each part has its aim and to be more specific, the first part concerns the seizure of the smartphone device and what steps have to be followed in order to preserve the evidences. The second part concerns the acquisition of evidences regarding the android logging system whereas the third part is about the acquisition of evidences from the internal memory storage partition of the

3 Recovery mode according to Hoog A. (2011) is an operating mode for Android that was designed to apply updates or format the device or perform other maintenance procedures on the device.

android system.

In addition, there are some drawbacks of the methodology not regarding the ACPO guidelines but in general with the methodology itself. Basically, the drawback is that this methodology depends on the USB Debugging mode to be enabled. But what will happen if the phone is locked with the use of a visible pattern or a password? As stated by Lessard J. and Kessler C G. (2010) If a forensic investigator is presented a locked device he or she may try to connect the device with the investigation unit with a USB cable and hope that the USB debugging mode is enabled or he or she can overwhelm the lock screen using another method and then enable the USB debugging mode. Nevertheless, they do not state a method for overwhelming the lock screen, hence it is debatable if the second method can be really implemented. In addition, Micro Systemation (MSAB) a company which deals with forensic technology for mobile device examination in their blog advocate that their team which is investigating the Android operating system has concluded that XRY (Micro Systemation's forensic tool) can extract data from the device without the need to enter the smartphone device password or pattern. This method works on version 1.5 and 2.2 of Android but again needs the USB debugging mode to be enabled (Micro Systemation Blog, 2011).

6. Conclusion

6.1 Critical Evaluation

As stated in the beginning of this study, this project had two main aims. The first aim was about finding out what information is stored on a smartphone device regarding bluetooth and Wi-Fi connections that were taking place on the smartphone device. The second aim was to investigate and analyse how can this information be extracted from the Android smartphone device during a forensic examination under the principle of good forensic practice, ACPO guidelines, assess and document the intervention as well as the implications from the smartphone's operation.

In order to accomplish the aims of the project a detailed background research needed to be done. Smartphone devices were identified and distinguished from the other kinds of mobile devices. In addition a thorough introduction to Android took place, where the architecture, the way the applications are used, the way the Android operating system partitions its memory etc were detailed described.

In addition, the term digital forensics was explained and how this implies for the subcategory of mobile forensics which is an emerging field of digital forensics. A research about the various tools that have been used by other researchers throughout the years on different mobile platforms was done in order to familiarise with the area of forensics. Nevertheless, a research regarding the guidelines of good forensic practice needed to be done since it was one of the project's main goal. Hence, various guidelines were discussed.

The project has clearly fulfilled its aims firstly by making an investigation of four different scenarios and finding what information was stored on the Android smartphone device. The four scenarios were all based on real life scenarios in order to make the investigation more realistic and to prove that research in this field is necessary. The investigation of the four scenarios was the most important factor for the achievement of this project, since without these investigations, none of the information was going to be found, neither the definition of the methodology for acquiring evidences.

In addition, the method for acquiring evidences during a forensic investigation was defined for the needs of this project. An evaluation of this technique with the use of ACPO Guidelines took place in order to examine whether this method can be utilised by forensic investigators. Some of the principles were violated either because there were no other options to choose from:

1. The case Principle 1 and the seizure and preservation of evidences was violated because the airplane mode was used but it was not an option to choose from (even though some researchers advise to use this method)
2. The case of Principle 2 was violated because a method for gaining root access in a forensically sound way has not yet been implemented or released to the public but it is necessary for the completion of the method.

It is possible that the Principles may change in a future ACPO Guidelines release, hence the method may have to change all over again and the evaluation of the above method with the new guidelines must take place again.

Moreover, all of the objectives of the project have been accomplished, since a method for acquiring an image from the Android operating system has been found. In addition, in order to acquire the evidences from the Android smartphone device, a method for identifying the filesystems that are stored on the Android operating system was found. The logging system that is used by Android has also been thoroughly investigated in order to help a forensic investigator during an examination. The implementation of the four scenarios was vital for the successfulness of this project since the

findings from each of the scenarios were clearly described, identified and illustrated. They were also analysed in order to identify which are more important and what information needs more attention during a forensic examination.

There are some improvements that can be done regarding the project which may improve the outcomes of the project. First of all, an investigation of the scenarios with different usage between the period the bluetooth or Wi-Fi connection took place until the forensic investigation could have been done in order to examine how the usage of the smartphone device affects the scenarios. In this project only the time which passed between the “illegal” action and the forensic investigation was considered whereas the smartphone device was used as a regular smartphone device (accepting phone calls, sending text messages, open and closing few applications). Hence, what if the smartphone device was used for more advanced reasons like, sharing the data plan of the smartphone device with other devices or, in contrast, what if the smartphone was left idle for all of the time. What implications would this kind of usage could have on the evidences found on the Android smartphone device?

Moreover, a good idea is to acquire image files before and after the bluetooth or Wi-Fi connections have been done in order to compare them and see their differences. The comparison of the two image files (the one prior with the one after the connection) will lead to more precise results regarding to what information was been altered during the connection. It must be also taken in consideration the fact that not only the bluetooth or Wi-Fi connection may have alter information on some files but some of the other running processes may have done some modification. Nevertheless, this will shrink the areas to look for.

6.2 Future work

Future work that needs to be done regarding this project is basically repeat the scenarios taking into consideration the improvements that were discussed at the end of the previous section. This may provide more information regarding the evidences and their life time. Furthermore, a technique which will root the phone in a forensically sound way will provide the investigators a more secure way for acquiring evidences. Hence, research must be done in this field in order to find a way for rooting all Android smartphone device in a forensically sound way. In addition, a way of activating the USB Debugging mode when the android smartphone device has a locked screen or overwhelming the locked screen in a forensically sound way can be considered future work in the area of smartphone forensics.

In addition, the above method was only tested with one smartphone device, Samsung Galaxy Europa which was running Android version 2.2.1, whereas there are thousands of Android smartphone devices in the market. Hence, more investigations of the scenarios can be done with other smartphone devices to verify the results of this study. Moreover, due to the fact that android operating system is an open source project, manufacturers may alter the source code as they would like, make modifications in order to make the operating system ran more smoothly on their device. Nevertheless, the latest Android release is versioned 3.2 whereas there are still smartphone devices which are running Android 1.5 version. Hence, it is sensible to test this method with other smartphone devices. Finally, a way for using a cryptographic hash function to verify that image files that are acquired from smartphone device are identical with the smartphone device partition needs to be found in order to verify that the image file acquired is identical with the partition of the smartphone device.

References

- Ahmed R and Dharaskar R. (2008) Mobile forensics: an overview, tools, future trends and challenges from law enforcement perspective In: 6th International Conference on E- Governance, ICEG, Emerging Technologies in E-Government, M-Government, pp. 312E23.
- Android.com (2010) Licenses [Online] Available: <http://source.android.com/source/licenses.html> Accessed: September 2011
- Android Developers (2010) Saving Data Safely. [Online] Available: <http://android-developers.blogspot.com/2010/12/saving-data-safely.html> Accessed: September 2011
- Android Developers, Developers' Guide (2011) Tools of Android. in the official developers' guide of Android website [Online] Available: <http://developer.android.com/guide/developing/tools/index.html> Accessed: September 2011
- Android Developers (2011) Platform Versions: Current Distribution. [Online] Available: <http://developer.android.com/resources/dashboard/platform-versions.html> Accessed: September 2011.
- Android-Locdump (2011) Official website of the android-locdump tool [Online] Available: <https://github.com/packetlss/android-locdump> Accessed: September 2011
- Arthur C. (2011) Android phones record user-locations according to research, The Guardian [Online] Available: <http://www.guardian.co.uk/technology/2011/apr/21/android-phones-record-user-locations> Accessed: September 2011
- Ayers R., Jansen W, Cilleros N and Daniellou R (2005) Cell Phone Forensics Tools: An Overview and Analysis, NIST Interagency Report (IR) 7250.
- Bradford P G, Brown Ma, Perdue J, Self B. (2004) Towards proactive computer-system forensics. In: Proceedings of the international conference on information technology: coding and computing.
- Busybox (2008) BusyBox: The swiss Army Knife of Embedded Linux [Online] Available: <http://www.busybox.net/> Accessed: September 2011
- Cannings R. (2011) An Update on Android Market Security. (Official Google mobile team website). [Online] Available :<http://googlemobile.blogspot.com/2011/03/update-on-android-market-security.html> Accessed: September 2011
- Casey, E., Bann M. and Doyle J (2010) Introduction to Windows Mobile Forensics. Digital Investigation Volume 6, Issues 3-4, May 2010, Pages 136-146 Embedded Systems Forensics: Smart Phones, GPS Devices, and Gaming Consoles
- Chessman S. (1996) What is dd?. The Linux Journal Issue #32. December 1996 [Online] Available: <http://www.linuxjournal.com/article/1320> Accessed: September 2011
- Daily Mail (2009) Barclays IT director jailed for downloading hundreds of child porn images to company laptop [Online] Available: <http://www.dailymail.co.uk/news/article-1171352/Barclays-IT-director-jailed-downloading-hundreds-child-porn-images-company-laptop.html> Accessed: September 2011
- Distefano A., Grillo .A, Lentini A. and Italiano G. F. (2010) SecureMyDroid: Enforcing Security in the Mobile Devices Lifecycle, Proc. 6th Annual Cyber Security and Information Intelligence Research Workshop (CSIIRW 2010), Oak Ridge, TN, USA, April 21 - 23, 2010.
- Distefano A and Me G. (2008) An overall assessment of mobile internal acquisition tool. Digital Investigation 2008;5:S121e7.

- Distefano A, Me G. and Pace F. (2010) Android anti-forensics through a local paradigm. Digital Investigation, Vol. 7, pp. S83-S94
- Faisal A., Dennis B., Brian D., Aaron D., and Ian I. (2011) An App to Simplify Security for Android Mobile Phone Users
- Google's Official Blog (2011) Android: Momentum, mobile and more [Online] Available: <http://googleblog.blogspot.com/2011/05/android-momentum-mobile-and-more-at.html> Accessed: May 2011.
- Höebarth S. and Mayrhofer R. (2011) A framework for on-device privilege escalation exploit execution on android, in *Proc. IWSSI/SPMU 2011*, June 2011
- Hoog , A. (2009) Presentation with title “Android Forensics” at Mobile Forensics World 2009
- Hoog, A. (2010) An introduction to Android forensics. [Online] Available: <http://www.dfineews.com/article/introduction-android-forensics> Accessed: September 2011.
- Hoog A, Strzempka K. (2010) iPhone forensics White paper. viaForensics; [Online] Available: <http://viaforensics.com/education/white-papers/iphone-forensics/> Accessed: September 2011.
- Hoog , A. (2011) Presentation with title “SQLite Forensics , The little database that could ” at Mobile Forensics Conference (MFW) in Myrtle Beach, SC. June 2011 [Online] Available: <http://viaforensics.com/computer-forensics/sqlite-forensics-presentation-andrew-hoog.html> Accessed: September 2011.
- Hughes P. (1994) The DF Command. The Linux Journal Issue #1. March 1994 [Online] Available: <http://www.linuxjournal.com/article/2747> Accessed: September 2011
- Huffington Post (2011) Innocent Man Accused Of Child Pornography After Neighbor Pirates His Wi-Fi [Online] Available: http://www.huffingtonpost.com/2011/04/24/unsecured-wifi-child-pornography-innocent_n_852996.html Accessed: September 2011
- Induruwa, A. (2009). Mobile phone forensics: an overview of technical and legal aspects. International Journal of Electronic Security and Digital Forensics, 2(2), 169-181.
- International Data Corporation (2011) Worldwide Quarterly Mobile Phone Tracker, March 29, 2011
- Jansen, W. and Ayers, R. (2004) Guidelines on PDA Forensics: An Overview and Analysis [Online] Available: <http://csrc.nist.gov/publications/nistir/nistir-7100-PDAForensics.pdf> Accessed: September 2011
- Jansen, W. and Ayers, R. (2006) Guidelines on cell phone forensics. Technical report, National Institution of Standards and Technology Special Publication, vol. 800, pp. 101.
- Kalba K, (2008) The adoption of mobile phones in emerging markets: global diffusion and the rural challenge. International Journal of Communication 2008;2:631e61.
- Klaver C. (2010) Windows Mobile Advanced Forensics. Digital Investigation Volume 6, Issues 3-4, May 2010, Pages 147-167. Embedded Systems Forensics: Smart Phones, GPS Devices, and Gaming Consoles
- L. Flash Software Group, Samsung Electronics Co. (2008) LinustoreII Samsung OneNAND Flash Linux Device Driver - GPL Compliance [Online] Available: http://www.samsung.com/global/business/semiconductor/products/flash/downloads/LinuStoreII_GPL%20Compliance_10.pdf Accessed: September: 2011
- Lessard J. and Kessler C G. (2010) Android Forensics: Simplifying Cell Phone Examinations. Small scale digital device forensics Journal Vol. 4, No. 1 ISSN#1941-6164

- Liles, S., Kovacik, S., and O'Day, D. (2010) A Proposed Methodology for Victim Android Forensics.
- Linux Sleuthing (2011) Defeating the Droid: Let the Pillaging Begin [Online] Available: <http://linuxsleuthing.blogspot.com/2011/06/defeating-droid-let-pillaging-begin.html> Accessed: September 2011
- Maus S., Höfken H. and Schuba M. (2011) Forensic Analysis of Geodata in Android Smartphones, Cyberforensics 2011, Glasgow, June 2011.
- McCarthy P., (2005) Forensic Analysis of Mobile Phones, BS CIS Thesis, University of South Australia, School of Computer and Information Science, Mawson Lakes, Oct. 2005.
- Micro Systemation Blog (2011) Android pass code. Android - extract data without entering the pass code? [Online] Available: <http://www.msab.com/posts/blog/android-pass-code> Accessed: September 2011
- Mislan, R.P., Casey, E., and Kessler, G.C. (2010) The growing need for on-scene triage of mobile devices. In Proceedings of Digital Investigation. 2010, 112-124.
- Mokhonoana P. and Olivier M. (2007) Acquisition of a symbian smart phone's content with an on-phone forensic tool. In Southern African Telecommunication Networks and Applications Conference 2007 (SATNAC 2007) Proceedings, 2007.
- Moore T., (2006) The Economics of Digital Forensics. Fifth Workshop on the Economics of Information Security, Cambridge, UK
- Mulazzani M., Schrittwieser S., Leithner M., Huber M. and Weippl E. (2011) Dark Clouds on the Horizon: Using Cloud Storage as Attack Vector and Online Slack Space
- Open Handset Alliance (OHA). (2009). Open handset alliance home page. [Online] Available: <http://www.openhandsetalliance.com> Accessed: September 2011
- Owen P. and Thomas P. (2011) An Analysis of Digital Forensic Examinations: Mobile Devices versus Hard Disk Drives utilising ACPO & NIST Guidelines. Small Scale Digital Forensic Journal Elsevier.
- Oxygen Forensic (2011) Android Rooting Add-on [Online] Available: <http://www.oxygen-forensic.com/en/features/androidroot/> Accessed: September 2011
- Palmer, Gary. (2001) A road map for digital forensics research - report from the first Digital Forensics Research Workshop (DFRWS). Technical Report DTR-T001-01 Final, Air Force Research Laboratory, Rome Research Site.
- Patzakis, J. M. (2003) Maintaining The Digital Chain of Custody. Password - The ISSA Journal, Oak Creek/U.S., p. 14-15, 2003. ISSN February 2003.
- Pooters I., (2010) Full user data acquisition from Symbian smart phones. Digital Investigation Volume 6, Issues 3-4, May 2010, Pages 125-135. Embedded Systems Forensics: Smart Phones, GPS Devices, and Gaming Consoles
- Punja S. G. and Mislan, R., (2008) Mobile device analysis. Small Scale Digital Device Forensic Journal, vol. 2.
- San Gabriel Valley Tribune (2011) Man accused of taking inappropriate pictures of girl at Arcadia shopping mall [Online] Available: <http://www.insidesocal.com/sgvcrime/2011/04/man-accused-of-taking-inapprop.html> Accessed: September 2011
- Schmidt A.D., Schmidt H.-G., Clausen J., Yuksel K. A., Kiraz O., Camtepe A., and Albayrak S.

(2008) Enhancing security of linux-based Android devices. In Proceedings of 15th International Linux Kongress. Lehmann, Oct 2008.

Spreitzenbarth M. (2011) Presentation at Graduate Workshop on Reactive Security with title “Tools and Processes for Forensic Analyses of Smartphones and Mobile Malware” [Online] Available: http://www1.hgi.rub.de/spring/content/spring6_07_slides_spreitzenbarth.pdf Accessed: September 2011.

SWGDE/SWGIT (Scientific Working Groups on Digital Evidence and Imaging Technology) (2011) Digital & Multimedia Evidence Glossary Version: 2.4 (January 14, 2011) [Online]. Available: http://www.theiai.org/guidelines/swgit/swgde/swgde_swgit_glossary_v2-4.pdf Accessed: September 2011

The Guardian (2011) Mobile phone exam cheat shocks Japanese meritocracy [Online] Available: <http://www.guardian.co.uk/world/2011/mar/04/japan-mobile-phone-exam-cheat> Accessed: September 2011

Thing V., Kian-Yong Ng and Ee-Chien Chang (2010) Live memory forensics of mobile phone. The Proceedings of the Tenth Annual DFRWS Conference, Volume 7, Supplement 1, August 2010, Pages S74-S82

Ubuntu Documentation (2010) Mount [Online] Available: <https://help.ubuntu.com/community/Mount> Accessed: September 2011

Ubuntu Manuals (2007) sqlitebrowser - light GUI editor for SQLite databases [Online] Available: <http://manpages.ubuntu.com/manpages/hardy/man1/sqlitebrowser.1.html> Accessed: September 2011

van der Knijff R. (2007) Speech with title: “10 good reasons why you should shift focus to small scale digital device forensics” [Online]. Available: http://www.dfrws.org/2007/proceedings/vanderknijff_pres.pdf Accessed: September 2011.

Vidas T. , Zhang C. and Christin N (2011), Towards a general collection methodology for android devices. Digital Forensics Research Conference, DFRWS 2011, Aug. 2011.

Williamson B, Apeldoorn P, Cheam B, McDonald M. (2006) Forensic analysis of the contents of Nokia mobile phones. In: Proceedings of the 4th Australian digital forensics conference .

Xinfang Li, Chunghuang Yang, Shihjen Chen and Jainshing Wu (August 2010) Design and Implementation of Forensic System in Android Smart Phone. The 5th Joint Workshop on Information Security (JWIS) South China Agricultural University, Guangzhou, China.