Forensic Analysis of Wireless Networking Evidence of Android Smartphones

Panagiotis Andriotis[1], George Oikonomou[2], Theo Tryfonas[3]

Bristol Cryptography Group, Faculty of Engineering, University of Bristol
Queen's Building, University Walk, Bristol, BS8 1TR, UK,
[1] P.Andriotis@bristol.ac.uk
[2] G.Oikonomou@bristol.ac.uk
[3] T.Tryfonas@bristol.ac.uk

***Abstract*—This paper introduces a method for acquiring forensic-grade evidence from Android smartphones using open source tools. We investigate in particular cases where the suspect has made use of the smartphone's Wi-Fi or Bluetooth interfaces. We discuss the forensic analysis of four case studies, which revealed traces that were left in the inner structure of three mobile Android devices and also indicated security vulnerabilities. Subsequently, we propose a detailed plan for forensic examiners to follow when dealing with investigations of potential crimes committed using the wireless facilities of a suspect Android smartphone. This method can be followed to perform physical acquisition of data without using commercial tools and then to examine them safely in order to discover any activity associated with wireless communications. We evaluate our method using the Association of Chief Police Officers' (ACPO) guidelines of good practice for computer-based, electronic evidence and demonstrate that it is made up of an acceptable host of procedures for mobile forensic analysis, focused specifically on the device's Bluetooth and Wi-Fi facilities.**

I. Introduction

Despite the rapid evolution of mobile devices over the recent past years, mobile phone forensics as a field within forensic science is at an early stage, when compared to traditional computer forensics. The increase and proliferation of mobile phones has noted the urgent need for the creation of new analysis tools and techniques [1]. Additionally, forensics investigators have to cope with different types of mobile phones, proprietary embedded firmware systems and a large amount of cable connectors for different models of phones [1].

Moreover, according to industrial research findings [2] on the worldwide smartphone operating system market share, sales indicate that Android is the leading operating system followed by the iOS. Thus, it is more likely to seize an Android phone from a crime scene for investigation, than a device running under any other operating system. Digital devices such as mobile phones, digital cameras and storage devices have become embedded in people's daily life. Ahmed and Dharaskar [1] note that in order to maintain acceptable standards during investigation, various guidelines of good forensic practice that include aspects of mobile phone processing have been issued. A good example is the United Kingdom's ACPO guidelines for computer-based electronic evidence [3].

The aim of our research is to introduce a method for acquiring evidence from an Android smartphone in a proper and acceptable forensic manner and also help the investigators to prove the use of the Bluetooth technology and wireless communications by a suspect in the course of a crime investigation. We follow the ACPO guidelines to prescribe a method that focuses on evidence related to the use of the wireless facilities of the phone, for which there is a gap in the literature. To meet our objectives we based the research on four case studies, which we reconstructed based on real cases that appeared in the press. After the execution of the experiments, we gathered information from three different smartphones. The investigation revealed traces common to different versions of the Android operating system. This adds value to the method because an investigator can refer to it regardless of the Android version of the suspect's smartphone. Moreover, the process is free of software cost since it is based on open source tools.

The rest of this paper is organized as follows. In section II we state our motivation and discuss related work on Android forensics. In section III we introduce the case studies and the method used to acquire physical images of the devices under examination. Experimental results are presented in section IV, alongside a critical evaluation of the method. Our conclusions are drawn in section V, where we also propose future work on this field of research.

II. Background and Related Work

A. Basics of Android

Android is an operating system developed by the Open Handset Alliance. Its architecture has four main levels: Linux Kernel, Libraries and Android Runtime, Application framework, Applications. Lessard and Kessler [4] 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. Thus, the use of a Flash Transition layer is needed in order to provide the mobile phone its functionality, due to the fact that flash memory devices are neither character nor block devices. The

WIFS'2012, December, 2-5, 2012, Tenerife, Spain. 978-1-4244-9080-6/10/$26.00 ©2012 IEEE.

connection between the Linux kernel and the physical flash device is accomplished using a Memory Technology Device (MTD). Vidas, Zhang and Christin [5] add that a lot of Android smartphone devices use MTD, an abstraction layer for raw devices which permits software to use an interface in order to use various flash technologies. They also illustrate that most Android devices have various partitions that are usually mapped to MTD devices. *User data, cache, boot* and *recovery* are some of the partitions in a typical Android system. The most common file systems an investigator can face during the analysis are the YAFFS2, FAT, EXT3 or 4 or other proprietary systems such as Samsung's Robust FAT file system (RFS). Especially with Samsung's RFS, in order to work timely, Linux kernel does not use MTD devices but creates several Sector Translation Layers (STL) and Block Management Layers (BML) block devices (*/dev/block/*).

Previous work indicates that the most interesting partitions for a forensic investigator will be the user data and system partitions [5]. According to Lessard and Kessler [4], a forensic examination of an Android based mobile device is mainly focused on the Libraries component of the architecture and specifically on the SQLite databases, where the majority of the data is stored. Moreover, in [6] we find that each application has its default home directory, which is located at */data/data/<package_name>/* and contains four directories: databases, libs, files and shared_prefs. Also, various manufactures may want to alter these directories to their own needs, for example Samsung uses */dbdata/databases/<package_name>/* for preinstalled applications. Hence, an investigator must know that there are various default locations on Android devices for applications to be stored.

B. Mobile Forensics and Guidelines of Good Practice

Mobile phone forensics can be defined as the art of retrieving digital evidence from a mobile phone using acceptable methods under forensically sound conditions. Mobile phones often retain important information which can be linked to a person per byte examined than most personal computers [4]. Because of their ability to hold a large amount of information, smartphones are considered as very interesting devices from a forensic perspective.

There is an increasing trend for using data stored on a mobile phone as evidence in civil or criminal cases [1]. Various types of information can be acquired from a mobile device using commercial forensic tools. The data could be phone books, logs of phone calls, SMS and MMS messages, Email, Instant Messaging content, URLs and content of visited Web sites, Audio, Videos, Images, SIM content [7]. Due to the fact that mobile devices can be found in many potential crime scenes, the capability to perform forensic analysis on them has become important to law enforcement.

Hoog introduced numerous forensics acquisition techniques for the Android Operating System in [7], however their drawback is that the investigator must have root privileges to perform the analysis on the phone. In our research we make use of the following.

- *Live analysis* utilising the Android Debug Bridge (ADB) tool, which can be installed along with the Android Software Development Kit (SDK) and provides the connection between an Android smartphone device and a remote workstation, such as a personal computer.
- *Physical imaging* provided by the utility *dd* (or *nand dump*): Due to its nature, Unix-like traditional forensic command-line tools such as dd, which allows a bit-by-bit physical imaging of Unix files, can work with the Android system conveniently [8].

The analyst of an Android smartphone should follow well-defined procedures to perform a forensically sound investigation. The Association of Chief Police Officers (ACPO) has issued a set of guidelines for obtaining computer-based electronic evidence that obey four basic principles: It is vital the data *not to be changed* by any action of the law enforcement agencies and in circumstances where a person finds it necessary to access original data held on a computer, this individual should be *competent* to do so. Also, there should be kept an adequate *audit trail* when investigating digital evidence and finally the case officer has *overall responsibility* for ensuring that the law and these principles are adhered to.

Thus, when a mobile phone acquisition is taking place, the critical part is to avoid altering the contents of the storage media of the phone. Owen and Thomas [9] advocate that the forensic examination of mobile devices is correlated with the methods and tools provided by the manufactures and that ACPO guidelines do recognise this fact. In addition, ACPO guidelines provide legal considerations and principles that must be followed in order to ensure that the integrity of the evidence remains intact.

Despite the plethora of sources that cover the forensic examination of Android smartphones in general, to the best of our knowledge there is no particular research that focuses on analysis of evidence of use of the wireless features of a smartphone. However, there exist sources documenting how Android smartphone devices are collecting data regarding the user's wireless activity. Apart from keeping a record of the user's locations, they also retain information such as the unique IDs of the last mobile mast they have communicated and the last Wi-Fi networks they have scanned. This kind of information is stored in a circular buffer, which overwrites the oldest information when the list is full. There is also a program capable of parsing those files and presenting the information in a human readable way [10].

III. Case Studies and Method

For the needs of our research four scenarios were created in order to investigate information related to the use of wireless communications stored in smartphones running the Android operating system. The scenarios we implemented were based on real cases reported by the global media.

1) Child pornography photographs were found in a company's laptop. The Police believe that the photographs were taken using the suspect's smartphone and were transferred to his laptop over a Bluetooth connection.

2) A man was arrested after witnesses saw him leaning over and holding what appeared to be a smartphone below a girl's skirt line. Authorities seized his smartphone and searched for Bluetooth activity.
3) A student has posted some questions from a test to an online forum and requested the answers from other users of the forum. The investigator aims to find information confirming that the smartphone was connected to the building's wireless network and that the relevant post was made by the suspect's device.
4) Someone is believed to be uploading inappropriate images using his neighbour's unsecured Wi-Fi network via his smartphone. To upload these photos, he is using his Dropbox application (a popular service for storing data to a cloud service).

We used three Android smartphones in our experiments. The main specifications of those devices can be found in table I. We should note that the phones were chosen in a way that covers all the updates of the second version of Android. In addition, the methodology we used to implement each case was the following:

Scenario 1: Obtain mobile, send files to device, wait a for period of time (use phone as usual), acquire data.

Scenario 2: Obtain mobile, take photos, send files to device, delete photos, wait (use phone as usual), acquire data.

Scenario 3: Obtain mobile, connect to Wi-Fi, post to forum, wait (use phone as usual), acquire data.

Scenario 4: Obtain mobile, connect to Wi-Fi, upload to Dropbox, wait (use phone as usual), acquire data.

The developing machine we used was running under the 64-bit Ubuntu 12.04 LTS operating system as the aim of our research was to provide a study of the remnants the use of wireless communications left on the Android system with open source utilities. The developing machine must have the Android Debug Bridge (ADB) tool installed, which is provided with the Android's software development kit (SDK) from the official Android developers site. Especially for the 64-bit machines the investigators should first install some 32-bit libraries by typing:

```
sudo apt-get install ia32-libs
```

Before starting the investigation of the seized smartphone the forensics examiner is advised to read the documentation for the ADB tool which can be found online in the official Android developers web site. ADB commands can be executed from the command line on the development machine and its format must be like this:

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

For the research presented here, we used ADB's *start-server, kill-server, shell, pull, logcat* commands. The syntax of the logcat tool is:

```
[adb] logcat [<options>] [<filter-specs>]
```

The ADB tool requires the USB debugging option of the phone to be enabled. At this point we should clarify that the method will be successful on *rooted* Android smartphones, meaning that the investigator has Super User privileges and also the BusyBox application is installed and works properly. This is far from ideal, but it is often encountered and discussed in relevant bibliography [11], [12]. A popular way to root an Android phone is by using the program SuperOneClick provided by *shortfuse.org*. The Super User and BusyBox Apps are provided by the Play Store (Android Market) at no cost. The BusyBox application contains Unix utilities for the phone and we will use the *df* command with the option *-h* to see the partitions relevant to our investigation. If the phone uses MTD devices we can see the partitions by inspecting */proc/mtd*. Moreover, the *mount* command provides information about the file system of the partitions.

The method for evidence acquisition from the smartphone consists of two major parts. Firstly, we keep the phone's log files stored in the *main* and the *events* ring buffers. We do not use the logs from the *system* and *radio* buffers in our cases because they are irrelevant to the use of the phone's wireless facilities. Afterwards we take a physical image of the device's critical partitions (*data* and *system*, as discussed earlier). Having performed those steps, we are able to safely examine the images by mounting them in our development machine as a virtual disk. The use of an SD card is vital for the method. The SD card must be formatted and wiped and its capacity must be bigger than the phone's internal memory.

The process a forensic investigator should follow to examine an Android phone for potential use of its wireless facilities is given below. The investigator should:

1) Make sure all the actions taken during the investigation are recorded. Date and time must be clearly marked on the record.
2) Obtain the mobile device and check whether the smartphone device has any SD card.
3) Activate the 'airplane mode', which turns off all the wireless connections of the smartphone device.
4) Activate the USB Debugging mode (located in Settings, Applications, Development, USB Debugging).
5) Connect the USB Cable from the developing machine to the smartphone device.
6) Start the ADB server by typing the command[1]:
   ```
   sudo adb start-server
   ```
7) Use the *logcat* tool to dump (-d option) the logs of the Android smartphone device which are formatted to show the time (-v option) using the commands:
   ```
   adb logcat -d -v -time -b main > file
   adb logcat -d -v -time -b events > file
   ```
8) Use the *md5sum* utility of the developing machine to produce a checksum for each and every file and then type:
   ```
   sudo adb kill-server
   ```
9) Turn off the device, remove any SIM card or SD card and send them to specialised laboratories for further examination.

[1] Commands below assume that the invoked tool is stored in the path

TABLE I
SPECIFICATIONS OF DEVICES

Specifications	Samsung Galaxy Europa	HTC Desire	LG Optimus E400
CPU	600 MHz	1 GHz Scorpion	800 MHz
Internal Memory	170 MB	576 MB RAM, 512 MB ROM	1 GB storage, 384 MB RAM
Android OS	v2.1-update1 (Eclair)	v2.1-upgraded to v2.2.2 (Froyo)	v2.3.6 (Gingerbread)
Bluetooth	v2.1 with A2DP	v2.1 with A2DP	v3.0 with A2DP
WLAN	Wi-Fi 802.11 b/g/n	Wi-Fi 802.11 b/g	Wi-Fi 802.11 b/g/n

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

IV. RESULTS AND EVALUATION

In order to evaluate the impact of time, i.e. to establish how volatile the evidence is and how long it is kept, we repeated each scenario three times. In the first scenario, we sent the images to a computer via Bluetooth and waited for 30 minutes before proceeding to the examination. After that, we used the Factory Data Reset facility of the Android phones to bring them in their initial state and repeated the experiments again. The second time we waited for 6 hours and in the third iteration waited for 12 hours. During these periods we used the phone as usual, made some telephone calls, visited web pages, received and sent emails. The methodology was repeated for all scenarios. We verified our results by performing a live analysis on the HTC Desire instead of doing a physical acquisition of the partitions. We already had the findings from the newest (2.1) and oldest (2.3) versions of Android and our basic concern was to confirm our results with version 2.2.

A. Bluetooth Connection

1) Log Files: The log files are stored on the device in circular buffers and the storage ability of those buffers differs between Android smartphones. For instance, the main buffer on a Samsung Galaxy is 256 Kb, while HTC and LG can only store up to 64 Kb. Because of the nature of circular buffers, time becomes a crucial enemy for the forensic examiner. Thus, it is highly possible to find important information, if the investigator seizes the phone right after a crime has been committed. In our cases we found information revealing the use of Bluetooth in the main and the events buffer. The events buffer revealed the use of Bluetooth up to 6 hours after the photos have been uploaded in both cases. However, we were not able to reveal the use of Bluetooth in the main buffer's log files. The only difference to this rule was noted in Samsung's main buffer, in instances when the seizure of the phone took place within 30 minutes after the photos were uploaded. The specific log file provided a lot of useful information, such as the name of the uploaded photos, the MAC addresses of the paired devices, the number of bytes sent and timestamps. On the other hand, the logs from the events buffer where able to prove the use of Bluetooth but not the files that were involved in this transaction. Furthermore, the logs that were taken from the phones longer than 12 hours after the criminal activity did not hold any relevant information.

As a conclusion, the log files are not always very informative about Bluetooth usage details and they depend on the time of seizure, the phone's storage capacity and its usage before the seizure.

2) Bluetooth Transaction History on Data Images: Forensic examiners will find all the information they need when they mount the data images on their development machines. We used the following commands to mount images from Samsumg Galaxy Europa and LG Optimus, respectively:

```
  sudo mount -o loop IMAGE.img /PATH
  sudo mount -t ext4 -o loop,ro,noexec,noload
IMAGE.img /PATH
```

The examination of the data partition images showed that under the path */misc/bluetoothd/MAC* (MAC depends on the device), six main files can be found: *classes, config, lastseen, linkkeys, names, profiles*. These files are used from the system to keep record of the last Bluetooth connections (lastseen). An examiner can find the MAC addresses of the paired devices, their given names and the link keys among other information. Furthermore, any transaction that has been made via a Bluetooth connection is recorded in an sqlite3 database which is stored in */data/com.android.bluetooth/databases*. The *btopp.db* file will reveal a lot to an examiner. For example, the fields of btopp.db taken from the LG E400 are: *id, uri, hint, data, mimetype, direction, owner, destination, visibility, confirm, status, total bytes, current bytes, timestamp, scanned*. During our experiments we were able to retrieve data about all the files that were exchanged, the filenames, their destination (MAC address), their timestamps, size, mime type. We used the popular tool SQLite Database Browser to browse the databases.

The two scenarios involving Bluetooth connections (scenarios 1 and 2) gave the same results. The log files were only sufficient to prove the use of the Bluetooth facility, however the databases found in the data partitions were much more revealing and useful for our investigation.

B. Wi-Fi Connection

1) Log Files: Similarly to the cases that involved Bluetooth connections, the log files we took after the completion of the third and fourth scenario did not contain sufficient information about the use of wireless networks. We were able to gather data from Samsung's main buffer when we seized the smartphone 30 minutes after the criminal activities took place and we also found information in the events buffer. Besides that, in any other case the log files did not reveal anything that could indicate the use of either the forum (scenario 3) or the Dropbox application (scenario 4).

2) Visited Wi-Fi Networks: However, the examination of the images of the data partitions showed that a forensic examiner can count on the databases of the Android system in order to find evidence. The first file we noticed was located in the */misc/wifi* folder. The name of the file is *wpa_supplicant.conf* and stores critical information about the visited wireless networks. The interesting part is that the examiner can find the passwords of the visited wireless networks in this file, which are stored as plain text. Additionally, in Samsung's data partition there is a database named *checkin.db* inside the folder */data/com.google.android.server.checkin/databases*. On the HTC, the respective database *htcCheckin.db* was found in */data/com.google.android.gsf/databases*. These databases hold information about the connections an Android device has made by activating and deactivating its Wi-Fi facility. It seems that the phone manufacturers decide where the home directories of those files should be placed.

3) Browsing History: Despite the diversity of the specific information, a forensic examiner should browse to the */data/com.android.browser/databases* folder of the data partition in order to find the browsing history of the suspect's smartphone. We found three files *(browser.db, webviewCache.db, webview.db)* in this folder that hold valuable information about the browsing history of the suspect. A notable finding is that in webview.db we can find the passwords for visited sites. These passwords are stored in webview.db without encryption and this fact can be viewed as a considerable vulnerability. An investigator can also use the locdump tool [10] to parse the files cache.wifi, cache.cell that are located in /data/com.google.android.location/files and correlate the results.

4) Dropbox Activity: The investigation of the fourth case study confirmed the findings we presented in the current subsection. Moreover, the use of Dropbox can be easily proved by the examination of the *db.db* file in the folder */data/com.dropbox.android* of the data partition. The specific database revealed the timestamps for every transaction, the files that were uploaded and relevant information.

Generally, a forensic examiner should investigate the files and folders that are presented in table II in order to collect data about the use of the Android's wireless networking facilities. Considering our four scenarios, the methodology revealed all the data an investigator could extract to close the cases successfully. More specifically, for the first and second scenario timestamps, paired devices, types and sizes of exchanged files were extracted, resulting in acquisition of data 100% useful for the case. For the third scenario we acquired timestamps, the names of wireless networks used to connect to the internet and the web browsing history (revealed the use of the forum during the exams). We had the same results after the investigation of the fourth case study and we also obtained more information from the correlation with the dropbox (db.db) activity (types and names of uploaded or downloaded files). Thus, the proposed scheme derived again 100% of usable data for the cases.

C. Methodology Evaluation

We offer an evaluation of our methodology based on the ACPO guidelines for good forensic practice [3]. We used the airplane mode of the phone to exclude any chance of

TABLE II
SOURCES OF EVIDENCE

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

interaction with a wireless network. This action is in accordance with the first principle, which underlines that no action taken by the examiner should alter data or storage media. Moreover, the investigator uses the md5sum utility to ensure that the investigated images and log files remain intact. However, the rooting procedure is vital for the second phase of the data acquisition and the tools we used (SuperOneClick and Busybox) are not yet registered as forensic tools. Besides that, the methodology does not take into consideration the case that the seized smartphone has activated security features like the locking of the screen with a pattern or a password. Nevertheless, we carefully planned the procedure to ensure that the acquisition of the files that can be affected by time (log files) happens first. In addition, the methodology is divided in three parts (preparation of the phone, acquisition of log files, physical data acquisition) meeting the requirements of the fourth principle and make the workload more distinct for the case officer to manage. We assume that the forensic examiners are competent enough to perform the investigation, are aware of the Android system and know how to handle basic Unix commands. We also provide the commands we used and the overall methodology is written in a detailed way. Further information for mounting an image of a YAFFS2 file system can be found in [5]. Finally, the information extracted from databases residing on the device data partition were sufficient enough not only to prove the use of Bluetooth and wireless networks by the suspects, but also to disclose the exact moves they made during their potentially criminal acts.

V. Conclusions and Future Work

To conclude, our research demonstrated that during the forensic examination of an Android smartphone an investigator is able to obtain information regarding the use of the Bluetooth technology and Wi-Fi networks. We indicated the files and folders the investigator should target and also presented a methodology for evidence acquisition focused around the use of the wireless facilities of the phone. Finally, we highlighted some security problems that occur by the exposure of the user's passwords in certain cases by the Android system.

Further work has to be done to confirm that the security problems have been solved in the third and fourth version of the operating system. In addition, research should be conducted to confirm if the file structure of the system remains the same in newer versions and the examiner can follow the same route we propose here in order to find evidence regarding the use of the wireless facilities of the phone. Finally, the methodology can be strengthened with the addition of new rooting procedures and monitor unlocking techniques.

Acknowledgment

This work has been supported by the European Union's Prevention of and Fight against Crime Programme ”Illegal Use of Internet” - ISEC 2010 Action Grants, grant ref. HOME/2010/ISEC/AG/INT-002. The authors would like to thank G. Kolokotronis, MSc graduate of Computer Science in Bristol, for his contribution to identifying the relevant scenarios discussed in the paper and also K. Anastasopoulou and S. Bradford for kind donations of relevant hardware.

References

[1] R. Ahmed and R. V. Dharaskar, “Mobile forensics: An introduction from indian law enforcement perspective,” in *Proc. Third International Conference on Information Systems, Technology and Management (ICISTM 2009)*, 2009, pp. 173–184.

[2] K. Restino. (2012, Jun.) Android expected to reach its peak this year as mobile phone shipments slow, according to idc. [Online]. Available: http://www.idc.com/getdoc.jsp?containerId=prUS23523812

[3] “Good practice guide for computer based electronic evidence,” Association of Chief Police Officers.

[4] J. Lessard and G. C. Kessler, “Android forensics: Simplifying cell phone examinations,” *Small Scale Digital Device Forensic Journal (SSDDFJ)*, vol. 4, no. 1, Sep. 2010.

[5] T. Vidas, C. Zhang, and N. Christin, “Toward a general collection methodology for android devices,” *Digital Investigation*, vol. 8, no. 1, pp. S14–S24, 2011, 11th Annual DFRWS Conference, New Orleans, LA, Aug 01-03, 2011.

[6] S. Höbarth and R. Mayrhofer, “A framework for on-device privilege escalation exploit execution on android,” in *Proc. IWSSI/SPMU 2011: 3rd International Workshop on Security and Privacy in Spontaneous Interaction and Mobile Phone Use*, Jun. 2011.

[7] A. Hoog, *Android Forensics*. Waltham, MA: Syngress-Elsevier, 2011.

[8] A. Distefano, G. Me, and F. Pace, “Android anti-forensics through a local paradigm,” *Digital Investigation*, vol. 7, no. 1, pp. S83–S94, Aug. 2010, the Proceedings of the Tenth Annual DFRWS Conference.

[9] P. Owen and P. Thomas, “An analysis of digital forensic examinations: Mobile devices versus hard disk drives utilising acpo & nist guidelines,” *Digital Investigation*, vol. 8, no. 2, pp. 135–140, 2011.

[10] M. Eriksson. (2012, Jun.) Android location service cache dumper. [Online]. Available: https://github.com/packetlss/android-locdump

[11] M. Yates and H. Chi, “A framework for designing benchmarks of investigating digital forensics tools for mobile devices,” in *Proc. 49th Annual Southeast Regional Conference*, ser. ACM-SE '11, 2011, pp. 179–184.

[12] J. Sylve, A. Case, L. Marziale, and G. G. Richard, “Acquisition and analysis of volatile memory from android devices,” *Digital Investigation*, vol. 8, no. 34, pp. 175–184, 2012.