



May 27th, 10:45 AM

## Forensic Analysis of Smartphones: The Android Data Extractor Lite (ADEL)

Felix Freiling

*University of Erlangen-Nuremberg, Germany*


Michael Spreitzenbarth

*University of Mannheim, Germany*

Sven Schmitt

*University of Mannheim, Germany*

Follow this and additional works at: <https://commons.erau.edu/adfsl>

 Part of the [Computer Engineering Commons](#), [Computer Law Commons](#), [Electrical and Computer Engineering Commons](#), [Forensic Science and Technology Commons](#), and the [Information Security Commons](#)

---

### Scholarly Commons Citation

Freiling, Felix; Spreitzenbarth, Michael; and Schmitt, Sven, "Forensic Analysis of Smartphones: The Android Data Extractor Lite (ADEL)" (2011). *Annual ADFSLS Conference on Digital Forensics, Security and Law*. 5.

<https://commons.erau.edu/adfsl/2011/friday/5>

This Peer Reviewed Paper is brought to you for free and open access by the Conferences at Scholarly Commons. It has been accepted for inclusion in Annual ADFSLS Conference on Digital Forensics, Security and Law by an authorized administrator of Scholarly Commons. For more information, please contact [commons@erau.edu](mailto:commons@erau.edu).

**EMBRY-RIDDLE**  
Aeronautical University™  
SCHOLARLY COMMONS

(c)ADFSLS



# **FORENSIC ANALYSIS OF SMARTPHONES: THE ANDROID DATA EXTRACTOR LITE (ADEL)**

**Felix Freiling**

University of Erlangen-Nuremberg  
Germany

**Michael Spreitzenbarth**

University of Mannheim  
Germany

**Sven Schmitt**

University of Mannheim  
Germany

## **ABSTRACT**

Due to the ubiquitous use of smartphones, these devices become an increasingly important source of digital evidence in forensic investigations. Thus, the recovery of digital traces from smartphones often plays an essential role for the examination and clarification of the facts in a case. Although some tools already exist regarding the examination of smartphone data, there is still a strong demand to develop further methods and tools for forensic extraction and analysis of data that is stored on smartphones. In this paper we describe specifications of smartphones running Android. We further introduce a newly developed tool – called ADEL – that is able to forensically extract and analyze data from SQLite databases on Android devices. Finally, a detailed report containing the results of the examination is created by the tool. The whole process is fully automated and takes account of main forensic principles.

Keywords: Android, Smartphones, Mobile devices, Forensics.

## **1. INTRODUCTION**

### **1.1 Why Forensic Analysis of Smartphones is Relevant**

In the recent years, smartphones became a very popular medium of communication. The associated communication market is one of the world's fastest growing markets [Gre10]. Among all cellular standards, GSM (Global System for Mobile communications) is the most widely used standard with 75% market share. It is used in 200 countries and has more than 1.2 billion users in over 630 mobile networks [Sch03].

As smartphones offer more and more diversity through a growing set of features, increasing amounts of sensitive data are created and stored on such devices. Through the ubiquitous use of smartphones, an increasing amount of such devices also becomes part of forensic investigations pursued by private organizations or law enforcement. These organizations need to be able to extract and analyze data that is stored on smartphones. Thus, there is a concrete demand for methods and tools that enable the execution of the before mentioned tasks in a forensically correct way. Furthermore, the rapid development of smartphone technologies makes it necessary to frequently scrutinize and adapt existing as well as develop new methods and tools for use in small scale digital device forensics.

### **1.2 The Case of Android**

According to Gartner [Gar10], the global distribution of smartphone operating systems is as follows: Symbian (Symbian Foundation) currently spearheads the market with about 40 percent of market

share followed by Android (Google) in the second and BlackBerry (RIM) in the third place, both with about 17 percent of market share. With about 15 percent iOS (Apple) makes it to the fourth place. The market share of Android grows with between 13 and 14 percent per year. According to an additional forecast by Gartner [Gar10] Android will be at the forefront of the worldwide mobile communications market by 2015. Therefore the Android platform is relevant for research in smartphone forensics.

### **1.3 Challenges of Forensic Investigations**

While forensic analysis of standard computer hardware – like hard disks – has developed into a stable discipline [Car05], there is still much debate on techniques to analyze non-standard hardware or transient evidence. Despite their increasing role in digital investigations, smartphones are still to be considered non-standard because of their heterogeneity. Within all investigations it is necessary to follow basic forensic principles. The two main principles are:

1. Greatest care must be taken that evidence is not manipulated or changed.
2. The course of a digital investigation must be understandable and open to scrutiny. At best, the results of the investigation must be reproducible by independent investigators.

Especially the first principle is a challenge in the setting of smartphones since they employ specific operating systems and hardware protection methods that prevent unrestricted access to the data on the system.

### **1.4 Contributions**

In this paper we give an overview over the problems that investigators are faced with when analyzing smartphones based on Android. Furthermore, we report on a prototype tool developed by the authors to perform a forensic analysis of Android smartphones. More specifically, we make the following contributions:

- We give an insight into the Android platform focusing on the specifics from the perspective of digital forensics.
- We discuss alternatives for extraction and analysis of data stored on Android devices.
- We present an overview of the SQLite data format, a central data format used in Android.
- We report on a prototype tool that we developed for forensic analysis of digital data stored in SQLite databases on Android devices.

### **1.5 Related Work**

The paper written by Lessard and Kessler [LK10] as well as the talk of Hoog [Hoo09] describe the forensic analysis of Android smartphones on the example of creating memory images and analyzing those with the help of well-known tools like Access Data's Forensic Tool Kit (FTK). This proposal basically refers to data carving and data recovery techniques. However, the SQLite databases that are found in the memory are not automatically parsed and the analysis of recovered data is handed over to the investigator. Mohindra steps up a quite similar environment like we do by explicitly dumping the SQLite databases from the device [Moh08]. But in contrast to our procedure, he manually analyzes the databases. Lee et al. make use of an interesting approach within their paper [LX10]; they use a prepared SD card on which they have placed an own forensic software (similar to the applications outlined in Section 3.1) in order to analyze the smartphones' data. By using an own SD card any change of data on the device is avoided. Databases are accessed with the help of Android system calls and SQL commands. In this case, an in-depth analysis of the SQLite data structures does not take place. Instead, the authors rely on the data delivered by the system.

## 1.6 Roadmap

This paper is organized as follows: We give an introduction to the structures of Android in Section 2. We then discuss alternatives for forensic data extraction of smartphones in Section 3. This is followed by an introduction to the SQLite file format, an important file format used in Android (Section 4). In Section 5 we present a new software solution that allows for automated examination of data stored in SQLite databases on Android devices while taking forensic principles into account. We conclude in Section 6.

## 2. OVERVIEW OF ANDROID

At the beginning of this section we will illustrate the Android version history and the market share of all versions active on the market in Table 1. Afterwards we will discuss a brief overview of the Android platform from a forensic point of view.

Version	Date	Market Share	Releases and used Hardware
1.x	September 2008 - October 2009	10 %	3 major releases (1.1, 1.5, 1.6) only used on smart phones
2.x	October 2009 - now	89 %	3 major releases (2.0, 2.2, 2.3) used on smartphones and tablets
3.x	January 2011 - now	1 %	only used on tablets

Table 1: Android version history and market share [Gog11]

Regarding Figure 1 the base of the Android platform is a Linux Kernel providing the necessary hardware drivers. The Dalvik Virtual Machine (DVM) is the core of the runtime environment. If an Android application is started, it runs in its own “sandbox” and with its own DVM. Although this costs extra resources it leads to more security and availability because applications do not share common memory. The application layer of Android accesses a plurality of fixedly implemented libraries, all deployed for operating required functionalities. Android provides several programming interfaces (APIs) which allow communication between applications and between the end user and applications. The top layer of the system is represented by the collectivity of applications. Within this layer, the interaction between humans and machines and the communication between applications take place. Each application makes thereby use of the underlying programming interfaces.

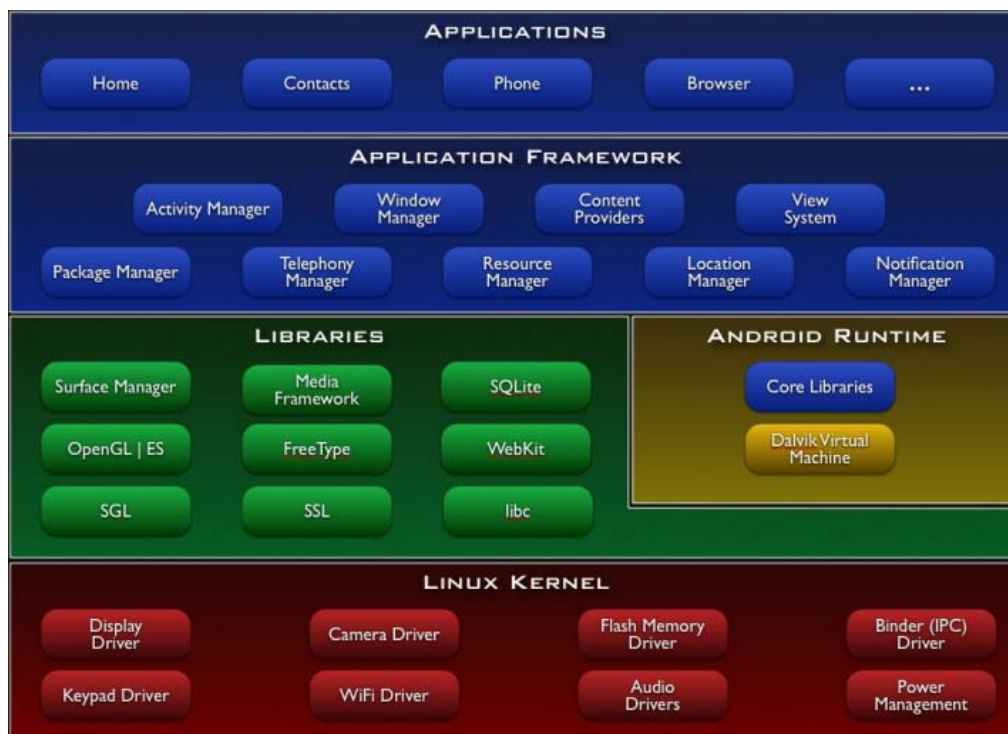


Figure 1: System architecture of Android [Gog10]

In addition to the execution in a virtual machine, applications that are run by Android are subject to several security mechanisms. They control the execution of any application and – if necessary – the access to data of other applications installed on the device. Meaning that when an application tries to interact with another database or application, the reference monitor, located in the application framework, looks at the permission labels assigned to this application and, if the target data access permission label is in that collection, allows the process to proceed. If the label is not in the collection, establishment is denied. The security mechanisms consist of the following three main parts [And10]:

- At the Linux kernel level user and group IDs are assigned to an application and thus provide a kind of isolation from the rest of the file system.
- A fine-grained permission mechanism enforces restrictions regarding specific operations that a particular process is allowed to perform.
- The root access is disabled on smartphones running in production mode.

These security mechanisms hinder the forensic examination of data stored in Android devices, because adequate permissions are required to access the data. At the same time, access is only possible through the Android platform itself, which is potentially manipulated or can lead to unintentional changes in the file system. This again violates the forensic principle that examined data must not be changed. To overcome these limitations, we have re-established root access to the smartphone and directly interact with the file system through the Android Debug Bridge (adb). For a more detailed explanation please refer to Section 5.

### **3. ALTERNATIVES TO THE EXTRACTION OF DATA**

There are two different approaches that principally exist regarding the forensic analysis of smartphone data. In the following, we will describe each while pointing out advantages and disadvantages.

#### **3.1 Software-based Approach: Software Agents**

In mobile phone forensics software agents are small programs that are installed on or copied to mobile phones to collect and analyze data locally or export data using the interface of the phone to examine it later. Two examples for forensic software agents are the “Open Source Android Forensic Agent” [San10] and “Panoptes” [Spr10]. Both agents need to be installed on the target Android device. After installation, they can be executed directly on the device and provide the investigator with CSV-files which contain the data extracted from the device in an already edited format. Through the use of “content providers” and the corresponding permissions, the sandbox of Android is broken and direct access to databases of other installed applications is granted. Due to this procedure stored and protected data from installed applications can be read.

Some advantages that result through the use of software agents are that little technical knowledge, no “rooting” of the device and no special hardware are required to read data from the device. With a software agent it is also possible to recover deleted data as long as it is still visible in the database files. However, a major disadvantage of agents is that data on the mobile phone is modified through copying or installing the agent and thereby a forensic principle is violated.

#### **3.2 Hardware-based Approach: Desoldering Memory Chips**

In the context of forensic investigations a common procedure is to remove any required memory chip from the circuit board of the device. Therefore, the memory chip is desoldered and then contacted through special hardware, such as PC-3000 Flash [Ace10]. The advantage of the hardware-based approach is that no intermediate layer potentially manipulates or prevents read access to the unaltered data on the chip. So, a high degree of forensic credibility can be assigned to the extracted data. However, one disadvantage of this method is the relatively higher effort that has to be made, compared to the software-based approach: Advanced technical equipment and knowledge are required for desoldering memory chips. When desoldering a chip from a circuit board, there also is a certain risk to damage or destroy the chip - and any potentially relevant digital trace with it. Nevertheless, this approach is a common method used in practice.

#### **3.3 Using the Android Debug Bridge**

To dump data from an Android device, it is possible to use the Android Software Development Kit (Android SDK). The Android SDK contains the Android Debug Bridge (adb) which is a client-server program that is able to connect to Android devices and execute a variety of commands on the connected device. Therefore an instance of the adb daemon must be running on the device which can be achieved by activating the option “USB debugging” on the target device.

As a further requirement, permissions on the device must grant access to the files that are to be dumped. Since permissions of Android devices in production mode deny the access to databases via the adb, one has to modify the firmware of the device or use a bootable “goldcard” in order to change the status of the phone in a way so that these security restrictions have been deactivated or can be bypassed.

Basically, this approach is a software-based approach but it is not necessary to inject new software into the smartphone. Furthermore, it requires some Android specific settings. Therefore it can be considered as an intermediate approach between software and hardware approaches. We use this approach later in this paper for our analysis tool ADEL.

## **4. THE SQLITE DATABASE FILE FORMAT**

### **4.1 Why SQLite is Relevant for Smartphones**

SQLite is a software library that implements a SQL database engine for embedded use. It can be integrated into other applications and – if necessary – be adapted to their specific requirements. One of the applications that make use of the SQLite software library is Android. It uses SQLite to store certain data on the underlying hardware device. This data contains information that is created by the user or by the OS, e. g. contacts, call lists, GPS data and SMS messages. Such data is of major interest within forensic examinations of mobile devices. This is why we took a closer look on how data is exactly stored by SQLite, which is defined by the SQLite database file format [SQL11]. This section will give a short introduction about important structures of the SQLite database file format. Each SQLite database is associated with a single file – the main database file – in the Android file system.

### **4.2 SQLite Internals**

The main database file holds all of the data stored in the associated database. During the execution of database operations temporarily created files may additionally be used, e. g. to be able to rollback database operations after a power failure (rollback-journal). However, we will not discuss temporarily created files in this paper, due to the fact that most of the time all data is stored within the main database file. This file consists of one or more data blocks, called pages, with a well-defined size. The page size is a constant amount of bytes and is valid for all of the pages within the same database file. The leading 100 bytes of the first page of a database file are used to store the database header. It contains general information about the database file, e. g. the size of the database in pages. The information in the database header allows for accessing the remaining contents of the database in a structured way. For a detailed overview of each of the database header fields, refer to the official documentation [SQL11].

Each table of a database is internally represented by a single b-tree structure. Within such a b-tree structure interior pages and leaf pages may appear. While interior pages store pointers to either further interior pages or to leaf pages, the actual content of the database is exclusively stored in leaf pages. The first page of a b-tree is called the root page and may be either an interior or a leaf page. The first page of a SQLite database file represents the root page to a special b-tree structure that holds contents of the so called “sqlite\_master” table. This table stores the complete database schema, including the SQL CREATE statements and pointers to the b-tree root page of each table in the database.

Each b-tree page (interior and leaf pages) is divided into different regions. In general those regions are:

- the page header,
- the cell pointer array,
- unallocated space,
- the cell content area.

There are only a few exceptions to the given order. One example is the first page of a database file that additionally stores the database header in the first place. Each page header holds general information about the layout of the page, e. g. the number of cells on this page. Immediately after the page header follows the cell pointer array. Each entry in the cell pointer array points to an offset on the same page at which the cell content is stored. For interior pages the cell content consists of two elements: the page number of the left child and a certain key value. For leaf pages each cell stores the content of a row – belonging to the corresponding table – and consists of four elements: the length of the payload in bytes, the SQLite internal ID of the row (rowID), the actual payload and an optional pointer to the first overflow page. Overflow pages are organized as linked lists and are used to store cell content that does not fit on the same page.

Database contents can thus be extracted by parsing the b-tree for each table contained in the database and extracting the contents of the cells found in any leaf page that belongs to the same b-tree (see Section 5.3).

### 5. ANDROID DATA EXTRACTOR LITE (ADEL)

We developed a tool named ADEL which is meant as an abbreviation of “Android Data Extractor Lite”. ADEL was developed for versions 2.x of Android and is able to automatically dump selected SQLite database files from Android devices and extract the contents stored within the dumped files. In this section we describe the main tasks of ADEL and what steps the tool actually performs. However, there are conditions that must apply for ADEL to work correctly. These conditions are stated in the following sections, corresponding to the relevant tasks. A flow chart showing the structure of ADEL is depicted in Figure 2.

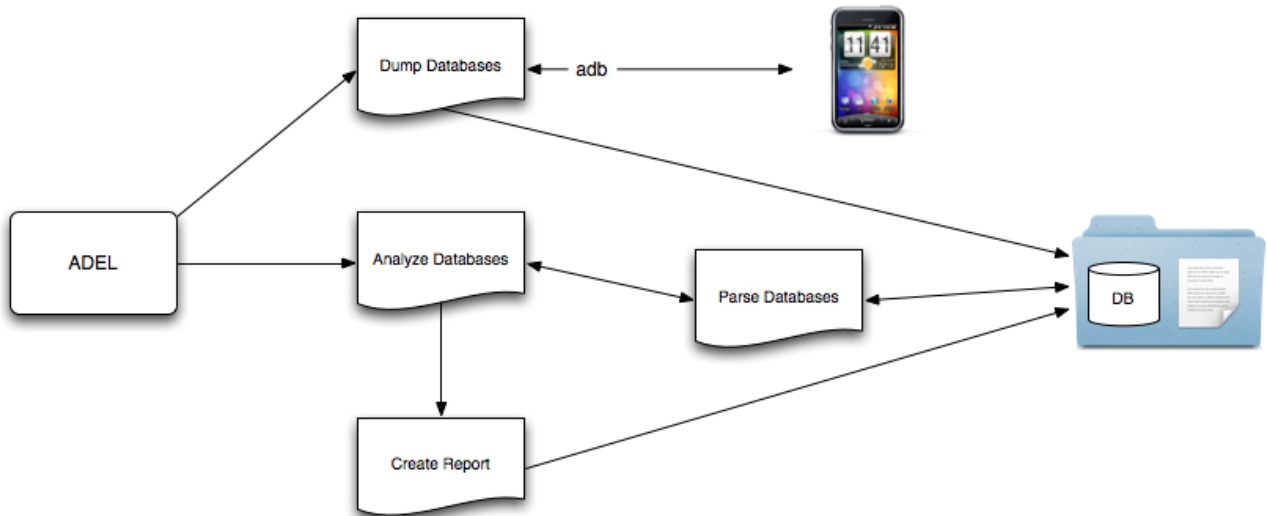


Figure 2: Structure of ADEL

#### 5.1 Basic Development Guidelines

During the development of ADEL we primarily took into account the following design guidelines:

**Forensic principles:** ADEL is intended to treat data in a forensically correct way. This goal is reached by the fact that activities are not conducted directly on the phone but on a copy of the databases. This procedure assures that data does not become changed, neither by the users of ADEL nor by an uncompromised operating system. In order to proof the forensic correctness of ADEL, hash values are calculated prior and after each analysis, to guarantee that dumped data did not become changed during analysis.

**Extendibility:** ADEL has been modularly built and contains two separate modules: the analysis and the report module. Predefined interfaces exist between these modules and both of them can be easily amended by additional functions. The modular structure allows for dumping and analyzing further databases of smartphones without great effort and facilitates updates of the system in the future.

**Usability:** The use of ADEL is intended to be as simple as possible to allow its use by both qualified persons and non-experts. At best, the analysis of the mobile phone is conducted in an autonomous way so that the user does not receive any notice of internal processes. Moreover, the report module creates



a detailed report in a readable form, including all of the decoded data. During the execution, ADEL optionally writes an extensive log file where all of the important steps that were executed are traced.

### **5.2 Data Extraction**

ADEL makes use of the Android Software Development Kit (Android SDK) to dump database files to the investigator's machine (see Section 3.3).

### **5.3 Parsing SQLite Database Files**

To extract contents contained within a SQLite database file ADEL parses the low-level data structures described in Section 4.2. After having opened the database file that is to be parsed in read-only mode, ADEL reads the database header (first 100 bytes of the file) and extracts the values for each of the header fields. Not all, but some of the values in the header fields are necessary to be able to parse the rest of the database file. An important value is the size of the pages in the database file which is required for parsing the b-tree structures (page-wise). After having read the database header fields, ADEL parses the b-tree that contains the “sqlite\_master” table for which the first page of the database always is the root page. The SQL CREATE statement and the page number of the b-tree root page are extracted for each of the database tables. Additionally, the SQL CREATE statement is further analyzed to extract the name and the data type for each column of the corresponding table. Finally the complete b-tree structure is parsed for each table, beginning at the b-tree root page that was extracted from the “sqlite\_master” table. Every leaf page of the b-tree is identified by following the pointers of all of the interior pages. Finally the row contents of each table are extracted from the cells found in any leaf page that belongs to the same table b-tree.

### **5.4 Reporting**

Within this section we address the report module and its functionalities. In the current development state, the following databases are forensically treated and parsed as described in Section 5.3:

- telephone and SIM-card information (e. g. IMSI and serial number)
- telephone book and call lists,
- calendar entries,
- SMS messages.

Data retrieved this way is written to an XML-File by the report module in order to ease further use and depiction of the data. As the analysis module, it can be easily updated regarding possible changes in future Android versions or in the underlying database schemas. Therefore, we have created different tuple – e. g. [table, row, column] – to define the data that is exchanged between both modules. If the database design changes in the future, only the tuple have to be adapted. The report module automatically creates XML-files for each of the data types listed above. In addition, a report is created which contains all data extracted from the analyzed databases. With the help of a XSL-file the report will be graphically refurbished. All files created by ADEL are stored in a subfolder of the current project.

## **6. CONCLUSION AND FUTURE WORK**

In this paper characteristics of the Android platform and the SQLite database engine have been discussed. Both aspects will become increasingly important for forensic examinations of Android mobile phones in the future.

We presented existing methods to analyze mobile phones and pointed out their advantages and disadvantages. Since these methods either violate forensic principles or necessitate advanced knowledge to perform the analysis, we have presented the tool ADEL which enables automated analysis. ADEL accesses the device via the Android Developer Interface in order to retrieve a copy of

selected SQLite databases. Subsequently, the SQLite databases are parsed and data is extracted and finally transformed into a XML-report by the modularly built analysis framework. During the development of ADEL main forensic principles have been taken into consideration.

#### REFERENCES

- [Ace10] ACE Laboratory (2010), 'Professional Data Recovery Product for SSD and Flash drives', <http://www.pc-3000flash.com>, 2010-10-13
- [And10] Android Developers (2010), 'What is Android?', <http://developer.android.com/guide/basics/what-is-android.html>, 2010-10-13
- [Car05] Carrier Brian (2005), 'File System Forensic Analysis', Addison-Wesley, Boston
- [Gar10] Gartner (2010), 'Mobile Communications by Open Operating System, Worldwide', <http://www.gartner.com>, 2010-06-15
- [Gog10] Google (2010), 'Android System Architecture', <http://www.techflare.com.au/media/102-android%20-%20system-architecture.jpg>, 2010-10-13
- [Gog11] Google Developer (2011), 'Google Developer - Platform Versions', <http://developer.android.com/resources/dashboard/platform-versions.html>, 2011-02-06
- [Gre10] Björn Greif (2010), 'Mobile telephone market increases 2009 up to 6.6 percent', [http://www.zdnet.de/news/wirtschaft\\_unternehmen\\_business\\_bitkom\\_mobilfunkmarkt\\_waechst\\_2009\\_um\\_6\\_6\\_prozent\\_story-39001020-41000202-1.htm](http://www.zdnet.de/news/wirtschaft_unternehmen_business_bitkom_mobilfunkmarkt_waechst_2009_um_6_6_prozent_story-39001020-41000202-1.htm), 2009-02-19
- [Hoo09] Hoog Andrew (2009), 'Android Forensics', Mobile Forensics World, 2009-05-26 to 2009-05-30, Chicago
- [LK10] Lessard Jeff and Kessler G. C. (2010), 'Android Forensics: Simplifying Cell Phone Examinations', Small Scale Digital Device Forensics Journal, Vol. 4 No. 1
- [LYC+10] Xinfang Lee, Chunghuang Yang, Shihjen Chen and Jainshing Wu (2010), 'Design and Implementation of Forensic System in Android Smart Phone', National Science Council of Taiwan
- [Moh08] Mohindra Dhruv (2008), 'Android, Incident Response and Forensics', SPRING, 2008-08-08, Mannheim
- [San10] SANS (2010), 'Open Source Android Digital Forensics Application', <https://blogs.sans.org/computer-forensics/author/andrewhoog/>, 2010-10-13
- [Sch03] Schiller J.H. (2003), 'Mobile Communications', Addison-Wesley, Boston
- [Spr10] Spreitzenbarth, Michael (2010), 'Panoptes: An Android Forensic Software Agent', University of Mannheim
- [SQL11] SQLite (2011), 'The SQLite Database File Format', <http://www.sqlite.org/fileformat2.html>, 2011-02-06

