

# Android Forensics: Simplifying Cell Phone Examinations

Jeff Lessard  
Champlain College  
*j.lessard802@gmail.com*

Gary C. Kessler  
Gary Kessler Associates  
Edith Cowan University  
*gck@garykessler.net*

## Authors' Note

This paper was initially written during the fall of 2009 and since that time, several new versions of Android OS have been available to customers via upgrades or new phone purchases. With each new phone and firmware update, there are initial challenges to the forensic community; the fundamentals of acquiring and analyzing an image, however, have remained the same.

## Introduction

It is hardly appropriate to call the devices many use to receive the occasional phone call a *telephone* any more. The capability of these devices is growing, as is the number of people utilizing them. By the end of 2009, 46.3% of mobile phones in use in the United States were reported to be smart phones (AdMob, 2010).

With the increased availability of these powerful devices, there is also a potential increase for criminals to use this technology as well. Criminals could use smart phones for a number of activities such as committing fraud over e-mail, harassment through text messages, trafficking of child pornography, communications related to narcotics, etc. The data stored on smart phones could be extremely useful to analysts through the course of an investigation. Indeed, mobile devices are already showing themselves to have a large volume of probative information that is linked to an individual with just basic call history, contact, and text message data; smart phones contain even more useful information, such as e-mail, browser history, and chat logs. Mobile devices probably have more probative information that can be linked to an individual per byte examined than most computers -- and this data is harder to acquire in a forensically proper fashion.

Part of the problem lies in the plethora of cell phones available today and a general lack of hardware, software, and/or interface standardization within the industry. These differences range from the media on which data is stored and the file system to the operating system and the effectiveness of certain tools. Even different model cell phones made by the same manufacture may require different data cables and software to access the phone's information.

The good news is there are numerous people in the field working on making smart phone forensics easier. Already there is material available on how to conduct an examination on Blackberry phones and a growing number of resources about the iPhone. However, there is a new smart phone OS on the market named Android and it will likely gain in appeal and market share over the next year. While Android initially launched with only one phone on T-Mobile, phones are now available on Sprint, Verizon and AT&T as well.

## Introduction to Android

Android is an operating system (OS) developed by the Open Handset Alliance (OHA). The Alliance is a coalition of more than 50 mobile technology companies ranging from handset manufactures and service providers to semiconductor manufacturers and software developers, including Acer, ARM, Google, eBay, HTC, Intel, LG Electronics, Qualcomm, Sprint, and T-Mobile. The stated goal of the OHA is to "accelerate innovation in mobile and offer consumers a richer, less expensive, and better mobile experience" (OHA, 2009, n.p.).

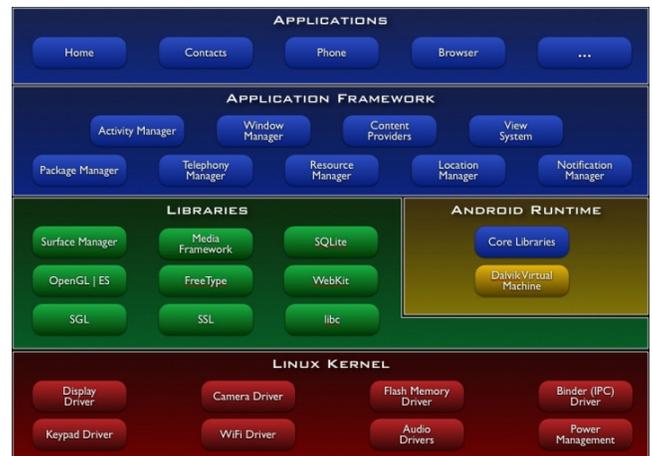


Figure 1. Android architecture (Android.com, 2009b).

The basic architecture of Android is shown in Figure 1. At its core, Android OS builds are based on the Linux 2.6 kernel. When running on a hard drive, the Linux system device defaults to the first physical hard drive, or /dev/hd0. In

addition, Linux only understands character and block devices, such as keyboards and disk drives, respectively. With Linux on flash, however, a Flash Transition layer provides the system device functionality. A Memory Technology Device (MTD) is needed to provide an interface between the Linux OS and the physical flash device because flash memory devices are not seen as character or block devices (Dedekind, 2009).

The Android Runtime System utilizes the Dalvik virtual machine (VM), which allows multiple applications to be run concurrently as each application is its own separate VM. Android applications (the *apps* of today's common parlance) are compiled into Dalvik executable (.dex) files (DalvikVM.com, 2008). During a forensic examination one will be mainly concerned with the Libraries and, in particular, the SQLite databases. This is where one will find the majority of data that could be of interest in an investigation. Files can be stored on either the device's storage or on the removable secure digital (SD) memory card (Android.com, 2009b).

Unlike the typical desktop operating system, data or other files created by one Android app cannot automatically be viewed by other applications by default. The VM nature of Android allows each application to run its own process. Security is permissions-based and attached at the process level by assigning user and group identifiers to the applications. Application cannot interfere with each other without being given the explicit permissions to do so (Android.com, 2009a).

The security mechanisms of the Android OS could impede a forensic examination although some of the basic tools and techniques could allow investigators to recover data from the device. The first, most obvious step is to perform a traditional forensics analysis of the microSD card from the phone. This is the least effective method as it can only access the data that apps directly store on the SD card. SD cards use the FAT32 file system and are easily imaged and examined using traditional forensics tools (including write-blocking hardware) (TalkForensics, 2009).

The Android file system is Yet Another Flash File System 2 (YAFFS2). YAFFS, developed in 2002, was the first file system designed for NAND (Not-AND) flash memory devices. YAFFS2 was designed in 2004 in response to the availability of larger sized NAND flash devices; older chips support a 512 byte page size whereas newer NAND memory has 2096 byte pages. YAFFS2 is backward compatible with YAFFS (Manning, 2002).

### Acquiring a Physical Image of an Android Device

Since Android is still an emerging OS and, forensics is in its infancy, this section will explore the steps of the analysis of an Android device. The following methods were assembled from research done and methods created by the android/htcmocking community as well as assistance from Andrew Hoog and ViaForensics.



Figure 2. Sprint HTC Hero (left) and information screen of test device (right).

As of July 2010, the latest version of Android available was v2.2 (Froyo) and v3.0 (Gingerbread) is expected before the end of the year. The analysis described below was performed during the fall of 2009 on a Sprint HTC Hero running Android v1.5 (aka Cupcake) (Figure 2). The Hero is a little different than a standard Android phone because HTC employs its own Sense user interface (UI) on the device, which will not be used on any Google-branded devices (HTC, 2009; Miller, 2009). While the Sense UI changes the look and feel of the device, it is uncertain how much (if any) this impacts a forensic investigation of the HTC Hero.

### Connecting the device via a data cable

Although the data cable for the Hero is a proprietary HTC cable (ExtUSB), an ordinary mini-USB cable will work for data transfers. The HTC cable handles running music and video over USB and would be desired for consumer applications but is not required for any type of forensics analysis.

### Imaging the memory card

Although an analysis of the removable memory of the phone has its limitation and phone system data is likely not stored to the memory card, it can still be a valuable tool. Making an image from the phone's memory card is quite simple and normal procedures for imaging a device can be used. In the analysis here, AccessData's FTK Imager v2.5.1 was employed.

The phone first needs to be connected to the examination machine using a write blocker to ensure the integrity of the data. Once the phone is connected, it will prompt that the USB cable is connected and ask the user to select to copy files to/from the host computer. Another screen then appears asking the user to mount the device (Figure 3).



Figure 3. "USB Connected" screen.

Once connected, the device will look for any necessary appropriate drivers. If issues arise, drivers are made available on HTC's website.

Now in FTK Imager, go to the File pulldown menu, and select the Add Evidence open, and then choose Physical Drive. Select the drive that is appropriate to the Android device (Figure 4). Note that the device will be the same size as the memory card (in this case, there is an 8GB microSD card in the device).

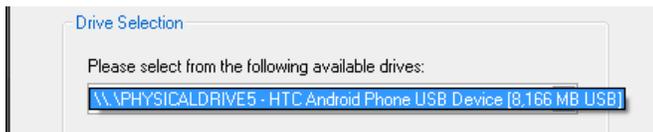


Figure 4. FTK Imager "Drive Selection" screen.

Save the image by using the File, Export disk image option. Make sure to take a physical image of the entire drive rather than a logical image of the partition. In this case, \\PHYSICALDRIVE5 was selected and imaged, sending the output to a raw dd image file. (The rationale for using dd for image files is provided below.) As with any image file, be sure to verify the hash prior to any subsequent analysis (Figure 5). Note that the SD card should be put aside and *not* replaced in the phone.

General	
Name	sdcard2.001
Sector count	15949824
MD5 Hash	
Computed hash	e3cbc7b88bc00cbc30227c528f31ade2
Report Hash	e3cbc7b88bc00cbc30227c528f31ade2
Verify result	Match
SHA1 Hash	
Computed hash	6c86800c1841e4a0aa80d1783248660d7ff06594
Report Hash	6c86800c1841e4a0aa80d1783248660d7ff06594
Verify result	Match

Figure 5. FTK Imager image summary screen.

### Importance of rooting the device in order to obtain a dd image

The ability to physically image memory is the holy grail of mobile device forensics. The device's memory can contain extremely valuable data, such as: the contact list, call logs, text messages, and other phone data. Additional information can also be hidden and uncovered, such as Web history, e-mails, images viewed on the phone, passwords, and fragments of other data. Access to memory can be accomplished by rooting the phone.

While the term *rooting* can have a negative connotation (similar to *jailbreaking* an iPhone), it has a different meaning than is generally perceived. Rooting a device merely means to gain access to the root directory (/) and having the appropriate permissions to take root actions. The modding community -- i.e., modern day hackers (in the 1970s sense of the word) who like to modify devices beyond the intentions of the device designers or vendors -- uses the term to mean accessing the root directory/permissions and then substantially modifying the phone to increase battery life or performance, run homebrewed applications, and/or install custom firmware on the phone (Purdy, 2009). Obviously, changing the data in such a way is not forensically sound and would not be done in an investigation.

Obtaining a dd image file is possible when the permissions are altered to gain access to the root directory. It is important to note that this method (at least for the Sprint HTC Hero), in its current iteration, needs to have a third party program installed on the device in order to get root permissions and likely would not be admissible in a court room setting. There are different ways to gain root permissions on other devices that do not involve adding anything to the phone but this is not the case on the Hero. The following method, then, should be viewed more of a proof of concept that could be tailored to be forensically sound if an alternate way to obtain root is found.

### USB Debugging

In order to acquire access to the root directory, Universal Serial Bus (USB) debugging will have to be enabled on the phone. Although the default setting is "disabled," going to Settings, selecting Applications, choosing Development and touching the checkbox, can turn on this function.

Root access will not be possible if an examiner encounters a locked Android device that does not have USB debugging enabled. If presented with a locked device, one may either attempt the method and hope that USB debugging is currently enabled by the user or must defeat the lock screen by some other method and then enable debugging using the outlined method above.

## Preparing the Hero for rooting

The method described here is based upon descriptions at The Unlockr.com (2009) and is the result of the work of many users at the XDA Developers forum (*forum.xda-developers.com*). The Android root access software was created by Christopher Lais at ZenThought.org. Be sure to insert a fresh SD card in the phone (do *not* replace the original SD card in the phone as it contains evidence that this process will alter).

The first step is to set up the Android Development Tools (ADT) on the host Linux, MacOS, or Windows computer system. The ADT is part of the Android Software Development Kit (SDK) (Android Developers, 2009). For a Windows system, download the SDK ZIP file and extract the files to the host computer.

The next step is to ensure that the phone and the Android development bridge (ADB) are both functioning as expected. In the Windows command line, move to the AndroidSDK folder, navigate to the `tools` subfolder, and run the `adb devices` command. If everything is working properly, a list of attached devices will show up with a corresponding serial number (Figure 6). If not presented with a list of devices, one must check that drivers are functioning properly and that USB debugging is enabled.



```

C:\>cd androidsdk
C:\AndroidSDK>cd tools
C:\AndroidSDK\tools>adb devices
* daemon not running. starting it now *
* daemon started successfully *
List of devices attached
HT99JHF00626    device

```

Figure 6. Starting the Android SDK in Windows.

**The method necessary to obtain root is specific to each phone and OS variant.** The following method was designed for the Sprint HTC Hero running OS version 1.5 and utilizes a program called AsRoot2 (ZenThought, 2009). The archive needs to be downloaded and the files extracted the files to the `Tools` folder and then execute the following commands (Figure 7):

```

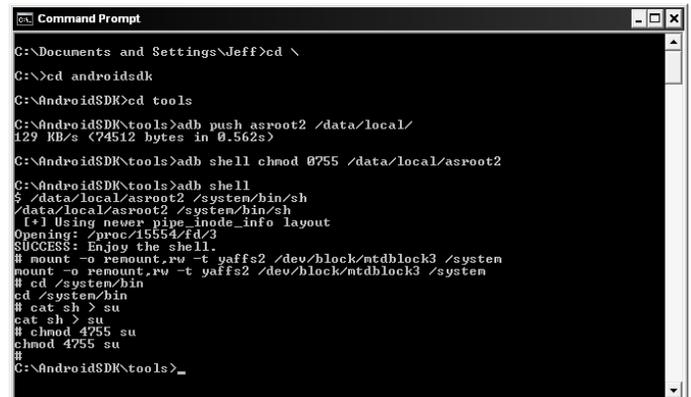
> adb push asroot2 /data/local/
> adb shell chmod 0755
/data/local/asroot2
> adb shell
$ /data/local/asroot2 /system/bin/sh
# mount -o remount,rw -t yaffs2
/dev/block/mtdblock3 /system

```

```

# cd /system/bin
# cat sh>su
# chmod 4755 su

```



```

C:\Documents and Settings\Jeff>cd \
C:\>cd androidsdk
C:\AndroidSDK>cd tools
C:\AndroidSDK\tools>adb push asroot2 /data/local/
129 KB/s (74512 bytes in 0.562s)
C:\AndroidSDK\tools>adb shell chmod 0755 /data/local/asroot2
C:\AndroidSDK\tools>adb shell
$ /data/local/asroot2 /system/bin/sh
/data/local/asroot2 /system/bin/sh
[*] Using newer pipe_inode_info layout
Opening: /proc/45554/fd/3
SUCCESS: Enjoy the shell.
# mount -o remount,rw -t yaffs2 /dev/block/mtdblock3 /system
mount -o remount,rw -t yaffs2 /dev/block/mtdblock3 /system
# cd /system/bin
# cat sh > su
cat sh > su
# chmod 4755 su
chmod 4755 su
#
C:\AndroidSDK\tools>_

```

Figure 7. Obtaining root access of the Android device in Windows.

If these steps all work correctly, the examiner should now have root permissions and can image the Android device. It should be noted that there is no real indicator that root access is available; to test out if it is functioning properly, continue and try to make a dd image of the memory (per the instructions below).

## Creating a dd image of memory

The file system of the Android device is stored in a few different places within `/dev`. Without the use of a traditional hard drive, the Linux kernel makes use of an MTD that allows for the embedded OS running directly on flash (SSI Embedded Systems, 2008). Although it may differ for other android phones, there are six files of interest located in `/dev/mtd/` (Android-DLs.com, 2009):

- `mtd0` handles miscellaneous tasks
- `mtd1` holds a recovery image
- `mtd2` contains the boot partition
- `mtd3` contains system files
- `mtd4` holds cache
- `mtd5` holds user data

Although it is important to image each file to obtain the complete operating system, the majority of this examination will focus on the information in `mtd3` and `mtd5`.

In order to image memory, the Android SDK shell will need to again be launched. As before, navigate to the `AndroidSDK\tools` directory, start the shell by executing the `adb shell` command, and then entering the `/data/local/asroot2 /system/bin/sh` instruction.

Once in the shell, the `dd` command can be used to image the memory files, using the command (Hoog, 2009a):

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

The command above will make a bit-for-bit image of the mtd0 file, using a block size of 1024 bytes, and copy the image file to the SD card. Repeat this command five more times in order to image the remaining five files of interest (Figure 8).

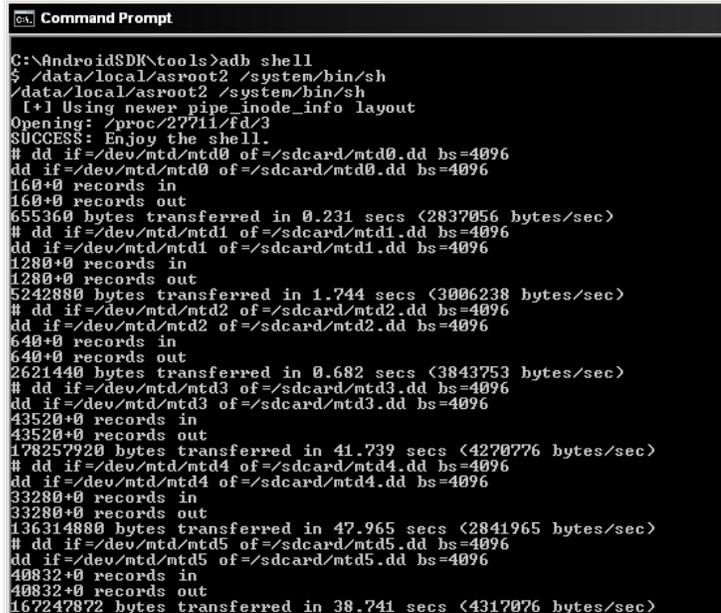


Figure 8. Obtaining root access of the Android device in Windows.

Note that this command will direct the output to the SD card. For this reason, it is imperative that a formatted and wiped SD card is placed into the phone and that the evidentiary SD card is put aside. It is also extremely important to not mix up the input file (if) and output file (of) parameters so as to not inadvertently destroy any data.

At this point, the dd files can be analyzed using any forensics software. Be sure to use a write-blocker when accessing the files on the SD card.

**Examination of Memory**

The examination of the memory image files was performed using Access Data's Forensic Tool Kit (FTK) v1.81. FTK was selected because of its data carving and searching capabilities; since today's forensic software does not mount the YAFFS2 file system, the ability for string searches was paramount.

When setting up the analysis in FTK, select options for full indexing and data carving, and add all six files for analysis. In this case, the subject phone was approximately two months old and had been used extensively for data applications. After data carving, 207 Hypertext Markup Language (HTML)

and Portable Data Format (PDF) documents were recovered, as were 12,709 BitMap (BMP), Graphics Interchange Format (GIF), Joint Photographic Experts Group (JPEG), and Portable Network Graphics (PNG) images.

**Recovered documents**

Most of the recovered documents were not of a real evidentiary value. A large portion of the HTML files were advertisements and only four files were complete snapshots of Web pages (Figure 9, left). The HTML files included 28 Exchangeable Image File (EXIF) data for JPEGs; this information can be helpful to determine what specific camera took an image.

**[edit] Miranda rights**



A CBP officer reading the Miranda rights to a suspect.

The Supreme Court did not specify the exact wording to be used when informing a suspect of his or her rights. However, the Court did create a set of guidelines which must be followed. The ruling states:

“...The person in custody must, prior to interrogation, be clearly informed that he or she has the [right to remain silent](#), and that anything the person says in a custodial situation, the typical warning is as follows:

“ You have the right to remain silent. Anything you say can and will be used against you in a court of law. You have the right to an attorney present during questioning. If you cannot afford an attorney, one will be appointed for you. Do you understand these rights? ”

The courts have since ruled that the warning must be “meaningful”, so it is usually required that the suspect be asked if he understands his rights. Sometimes, firm answers of “yes” are required. Some departments and jurisdictions require that an officer ask “do you understand?” after every sentence in the warning. An arrestee's silence is not a waiver. Evidence has been ruled inadmissible because of an arrestee's poor knowledge of English and the failure of arresting officers to provide the warning in the

```
amazon external hard drive
ST305004EXA101|
runstar
touchscreen gloves amazon
conductive thread
conductive gloves
morgan freeman
samoyed puppies
party hard lyrics
sublime scarlet begonias lyrics
ball and chain lyrics
pulp fiction soundtrack
where the wild parties are woot
```

Figure 9. Recovered files: Web page (left) and Google search history (right).

One particularly interesting document that contained useful information was the single recovered PDF file. This file was extremely fragmented and while Acrobat Reader reported that the file was corrupt and could not be opened, FTK was able to view the contents. The file was 2 MB in size and was substantially larger than all of the other recovered documents. It contained information such as text messages, phone book information, browser history, Facebook status updates, Google search history (Figure 9, right), YouTube videos visited, and music played from the SD card. It was difficult to look through

because it was so fragmented but searching the document made information easier to find.

**Recovered images**

As on a typical computer, this Android device had nearly 13,000 images, only some of which would be interesting in a forensics examination. The first noteworthy images found were the ones displayed as the phone is booting up. There are three different images: the HTC logo, a Hero splash screen, and a Sprint screen. The HTC logo screen is displayed at two points in the booting process and features the HTC logo in a beveled silver text on a reflective black background. As the phone boots, the source of light in the image changes as it pans across the logo – this seems like a loading screen, indicating something is happening like a progress bar would. This logo was merely an animated GIF file.

The mtd3.dd file contained images for different applications. Backgrounds for a labyrinth style game; images for bookmarks, weather, alarm clocks, keyboards, and widgets; grids for Sudoku games; and icons for check boxes, contacts, camera, and navigation apps were found.



Figure 10. Recovered images: Corrupted image file (left) and intact image file (right).

The mtd4.dd file contains contents of the Android cache. Recovered images from this location included some that were viewed from e-mail; some of the images were corrupted while others were perfectly intact (Figure 10).

Interestingly, only 30 images from the user's Gmail account were found. The highly fragmented condition of some of these images suggests that the amount of space allowed for caching of images viewed from Gmail is not large. Alternatively, it is possible that FTK was not able to locate or identify the images.

Another interesting result was that two of the images in the cache, although on the Gmail account, were never specifically called up or viewed on the phone. The best explanation is that they were preloaded from viewing the email, although the user never selected to download or view them.

The mtd5.dd file contains the user data and, not surprisingly, is where the majority of the recovered images were found. These were the types of pictures one would expect to find, namely images ranging from contact photos, downloads

from browser Web pages, pictures taken with the Hero's camera and sent to someone via the Multimedia Messaging Service (MMS) or e-mail to those from applications such as Facebook, cover art from Pandora, image previews of videos from SprintTV and YouTube, and icons from applications.

**Searching**

While browsing through images and documents yielded some helpful information, FTK was unable to locate text messages, e-mails, contacts, and call history. The search tool is quite powerful but in order to use it, an examiner needs to have an idea of what to search for. When trying to find emails, a logical starting point would be to search for the suspect's e-mail address. A search for *j.lessard802@gmail.com*, for example, yielded 1628 hits over 92 files. The files generally started with the e-mail address, followed by a preview of the body of the message and then the rest of the e-mail and recipient information. Many of the strings found looked like this one:

*j.lessard802@gmail.com* >..ö7`à..ö7c\$Ryan and Ysa I quite impressed with the talk they gave our class. Maybe impre....Ryan and Ysa<br><br>I quite impressed with the talk they gave our class. Maybe impressed isnt quite the right word for it - perhaps amazed they let everyone in to their life like that. I never really thought about the difficulty of communicating across cultures and how it would impact a relationship. Specifically if they didnt speak each others language. I guess the international language is truly dance.<br>

It is likely that if the suspect were using a mobile e-mail client (such as a gmail application) would yield more messages than a system where only Web mail has been employed.

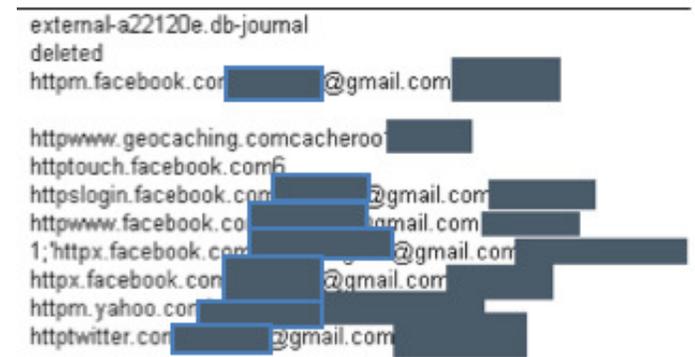


Figure 11. User names and passwords found in plaintext, blacked out for publication.



128	419	http://www.facebook.com/...	1	1259724574607	0
129	420	http://www.youtube.com/...	1	1259766087329	0
130	421	http://m.youtube.com/...	1	1259766119161	0
131	422	http://www.newegg.com/...	1	1259794961773	0

Figure 19. Browser history.

237	237	where the wild partie	1259461432984
238	238	pulp fiction soundtra	1259465046279
239	239	ball and chain lyrics	1259469379237
240	240	sublime scarlet bego	1259469890406
241	241	party hard lyrics	1259472218623

Figure 20. Browser search history.

The database file /data/data/com.android.browser/databases/browser.db is a separate database for the Android browser. The contents of this file included usernames, URLs, and plaintext passwords (Figure 17), data typed into forms (Figure 18), web browser history (Figure 19) and search history (although it was thought that this information had been deleted) (Figure 20).

Another interesting file is the /data/data/com.android.browser/gears/geolocation.db, which stores the last known location as reported by the GPS satellites (Figure 21).

Name	Latitude	Longitude	Altitude	Accuracy	AltitudeAccuracy	Timestamp	StreetNumber
LastPosition	42.735285	-71.479529	-1000	1589		1259013414291	

Figure 21. Last known geographic location.

The Google maps database can be found in /data/data/com.google.android.apps.maps/databases/search\_history.db. This file contains the history saved for all searches entered into the Google maps application (Figure 22).

_id	data1	singleResult
1	9 camp heartland ny	
2	10 west milford	
3	11 loc: north st at lafour	
4	12 23 hazen drive	
5	13 44.477128,-73.1986	

Figure 22. Google maps database.

_id	username	encryptedpassword	sha1hash	flags	registered
1	jlessard@gmail.com	AEv<DtoN5HrNDtp0imDzhpCFmgE.sHmzshAQHEBCTKSAMIAw9QX&E	40c55493045b0019	1	1

Figure 23. Google apps account information.

The Google applications database is found in /data/data/com.google.android/.googleapps/databases/accounts.db. This file contains Google apps account information, including the user name and the encrypted passwords (Figure 23).

person	date	protocol	read	status	type	reply	su	body
24	1255386664946		1	-1	2			Well I have to be at both
24	1255386710649	0	1	-1	1	1		Ok i will see what i am d
24	1255386765399		1	-1	2			Cool. Thanks!
58	1255438621715		1	-1	2			Btw. I found your soap a
30	1255447130041	0	1	-1	1	0		Ancielo:

Figure 24. MMS/SMS message information.

The /data/data/com.android.providers.telephony/databases/ directory contains information related to the messaging applications, including picture and text message data. The mmssms.db database contains the MMS and Short Message Service (SMS) messages [Address field truncated] (Figure 24). Note that the contents in this database included some deleted messages although no messages that were deleted more than 45 days prior were available. It is likely that the retained deleted messages would depend on the phone and individual user.

```

UN-1C.AMR
UN-9.AMR
UN-8.AMR
UN-12.AMR
UN-1A.AMR
UN-1.AMR
UN-13.AMR
UN-7.AMR
UN-5.AMR
UN-11.AMR
UN-17.AMR
UN-F.AMR
UN-10.AMR
UN-6.AMR
UN-19.AMR
UN-16.AMR
UN-a.AMR
UN-D.AMR
UN-8.AMR
UN-C.AMR
# dd if=/data/data/com.coremobility.app.unotes/files/UN-7.AMR of=/sdcard/UN-7.AMR
R
dd if=/data/data/com.coremobility.app.unotes/files/UN-7.AMR of=/sdcard/UN-7.AMR
73+1 records in
73+1 records out
37606 bytes transferred in 0.015 secs (2507066 bytes/sec)
# dd if=/data/data/com.coremobility.app.unotes/files/UN-C.AMR of=/sdcard/UN-C.AMR
R
dd if=/data/data/com.coremobility.app.unotes/files/UN-C.AMR of=/sdcard/UN-C.AMR
37+1 records in
37+1 records out
19334 bytes transferred in 0.008 secs (2416750 bytes/sec)
# dd if=/data/data/com.coremobility.app.unotes/files/UN-9.AMR of=/sdcard/UN-9.AMR
R
dd if=/data/data/com.coremobility.app.unotes/files/UN-9.AMR of=/sdcard/UN-9.AMR
64+1 records in
64+1 records out
32870 bytes transferred in 0.013 secs (2528461 bytes/sec)
#
    
```

Figure 25. Voice mail audio files.



while in place in the phone. The UFED can acquire data logically or physically, although physical acquisition is not supported for the HTC Hero. The UFED acts as a write blocker so no information is written to the phone when conducting an examination (CelleBrite, 2010).

In order to connect the HTC Hero to the UFED, USB storage and USB debugging both need to be turned on. The UFED walks an examiner through the steps needed to logically acquire data; the examination output in this case was an HTML file directed to a USB thumb drive.



Figure 35. Two of the picture files extracted by the UFED.

#	File Name	File Size	File Date/Time	File Link
1	VIDE00001.3gp MD5: 0569D5FE42A0AC9BB1AE797ED3BDC0F0 SHA256: 271A3C34 8A26480 2B536D8 0F46743 04DBCB1 94398B3 B874EBD FA924B 0E2949D	1250247 Bytes	10/20/09 23:47:13	<a href="#">VIDE00001.3gp</a>

Figure 36. Video file extracted by the UFED.

The 69 pictures that were extracted from the phone came from shots taken by the phone's camera, screenshots of bookmarked Websites, and those received and downloaded as MMS messages. Two images are shown in Figure 35. Note that the EXIF information suggests that this phone may have taken the image at the top, while the picture at the bottom was not taken by this phone. Note also the different picture file naming format, further evidence that the files were created by different cameras. The one video that was found was taken with the camcorder feature in the Hero (Figure 36).

**Summary of Results**

This experiment in acquiring information from an Android device using multiple methods is far from conclusive, although it provided some interesting insights:

- dd analysis with FTK
  - **Pros:** Found deleted text messages and contacts that would have likely not been located utilizing another method, found passwords with relative ease.
  - **Cons:** Required root access, results extremely fragmented, countless hours would have to be spent to try to locate and piece everything together (although another forensic suite may have netted better handling of the file system and FTK easily could in the future with an update).
- Logical analysis of specific databases
  - **Pros:** Recovered virtually everything that could be helpful to a mobile forensic investigation including call history, Web and search history, pictures, MMS/SMS messages, e-mail data with complete messages, and even GPS data, voice mail and passwords.
  - **Cons:** Required root access, did not find all deleted SMS messages, phone records, and contact info.

**Phone Examination Report Properties**

Selected Manufacturer:	HTC
Selected Model:	HTC Hero CDMA (Android)
Detected Manufacturer:	sprint
Detected Model:	HERO200
Revision:	1.5 CUPCAKE eng.u70000.20090921.205629
MEID:	270113178313016459 (HEX: A1000007C69D8B)
IMSI:	31006032060645
Extraction start date/time:	11/06/09 16:39:45
Extraction end date/time:	11/06/09 16:51:23
Phone Date/Time:	11/06/09 20:38:51 (GMT)
Connection Type:	USB Cable
UFED Version:	Software: 1.1.2.4 UFED , Full Image: 1.0.2.4 , Tiny Image: 1.0.2.1
UFED S/N:	5518965

Figure 32. Phone identifying information from the UFED.

4	* Twitter	10/11/09 13:40:26 (GMT)	Read	Inbox	Phon
171658	* Shannon Maguire	10/11/09 06:18:47 (GMT)	Read	Inbox	Phon
052307	* Steve Charbonneau	10/11/09 04:19:44 (GMT)	Read	Inbox	Phon
052307	* Steve Charbonneau	10/11/09 03:49:45 (GMT)	Sent	Sent	Phon

Figure 33. Some of the SMS messages extracted by the UFED [Phone numbers truncated for publication].

104	Incoming		* Jim Lessard	11/05/11
105	Incoming		* Shannon Maguire	11/06/11
106	Incoming		* Britney Langdon	11/06/11
107	Incoming		NA	11/06/11
189	Outgoing	54	* Mike Anziello	11/06/11
190	Outgoing	57	* Kathleen Wanner	11/06/11
191	Outgoing	82	* Sam Masfield	11/06/11
192	Outgoing	13	NA	11/06/11
46	Missed		* Andrew Cote	11/05/11
47	Missed		* Jodi Lessard	11/05/11
48	Missed		* Shannon Maguire	11/06/11
49	Missed		NA	11/06/11

Figure 34. Some of the call history information extracted by the UFED; incoming calls (top), outgoing calls (middle), and missed calls (bottom).

The CelleBrite device starts its report with basic phone identifying information, such as the acquired device type, software level, mobile equipment identifier (MEID), and the data and time of the data acquisition (Figure 32). In this instance, the UFED recovered 1070 SMS messages (Figure 33), 56 contacts, 107 incoming calls, 192 outgoing calls, 49 missed calls (Figure 34), 69 pictures, and one video. It was able to report on each category 100% correctly, as confirmed by examination of the phone itself.

- Data extraction with the CelleBrite UFED
  - **Pros:** Recovered MMS/SMS messages, call logs, photos, video, and contact information; simple, stand-alone method.
  - **Cons:** Logical extraction only (physical acquisition not yet supported); did not recover e-mails, browser, or search history.

It appears that browsing the databases logically netted the most information in an easily viewable way. Obtaining a dd image is extremely valuable but, aside from the user reconstructing where all the pieces fit, it was not the best method in this case. A different tool or forensics software with specific YAFFS2 support would make the physical analysis a winner. As it stands now, however, FTK would be most valuable when searching for very specific strings of text.

## Conclusion

Cell phones are becoming even more sophisticated and able. Both law enforcement and the private sector need to invest time and money into learning about new operating systems and developing new forensic methods.

While Android forensics is still in its infancy, steps are being made to meet the new technology. CelleBrite (2010), Paraben (2008), and .XRY (Micro Systemation, 2008) all currently offer some type of Android solution and more tools will be adding support as Android gains in popularity. Android is not just for phones either; it can be used on computers, kitchen appliances, and military applications (Spencer, 2009). Expect to begin seeing it everywhere.

The number of Android phones will be continuously increasing as more manufactures adopt the budding OS. As it stands now, Android sales, by some estimates, will overtake iPhone sales within the next two to three years (Lomas, 2009). While Android is powerful, complex, has multiple firmware implementations and some with manufactures making custom UIs, the standardization will make mobile forensics simpler in the long run. Indeed, as the market for Android continues to grow, learning how to forensically acquire information from these devices becomes essential for mobile device examiners.

## Author Information

Jeff Lessard received a B.S. degree in Computer & Digital Forensics from Champlain College (Burlington, Vermont) in December 2009. This paper is an expansion of his senior thesis project. All screen shots, unless otherwise noted, were taken by Jeff.

Gary C. Kessler, Ed.S., is president of Gary Kessler Associates, adjunct associate professor at Edith Cowan University (Perth, Western Australia), and mobile device examiner for the Vermont Internet Crimes Against Children (ICAC) Task Force. At the time of this project, he was an Associate Professor and director of the M.S. in Digital Investigation Management

program at Champlain College. He is a Certified Computer Examiner (CCE) and Certified Information Systems Security Professional (CISSP), and is an associate editor at the *Journal of Digital Forensic Practice* and *Journal of Digital Forensics, Security and Law*.

## References

- AdMob. (2010, January). *AdMob mobile metrics report*. Retrieved February 2, 2010, from <http://metrics.admob.com/wp-content/uploads/2010/01/AdMob-Mobile-Metrics-Dec-09.pdf>
- Android.com. (2009a, December 16). Android security and permissions. Retrieved December 21, 2009, from <http://developer.android.com/guide/topics/security/security.html>
- Android.com. (2009b, December 16). What is android? Retrieved December 21, 2009, from <http://developer.android.com/guide/basics/what-is-android.html>
- Android-DLs.com. (2009, December 7). Edit and re-pack boot images. *Android-DLs Web site*. Retrieved December 21, 2009, from [http://android-dls.com/wiki/index.php?title=HOWTO:\\_Unpack%2C\\_Edit%2C\\_and\\_Re-Pack\\_Boot\\_Images](http://android-dls.com/wiki/index.php?title=HOWTO:_Unpack%2C_Edit%2C_and_Re-Pack_Boot_Images)
- Android Developers. (2009, December). Download the Android SDK. *Android Developers Web site*. Retrieved December 21, 2009, from <http://developer.android.com/sdk/index.html>
- CelleBrite. (2010). UFED standard kit. *CelleBrite Web site*. Retrieved August 15, 2010, from <http://www.cellebrite.com/UFED-Standard-Kit.html>
- DalvikVM.com. (2008). Dalvik virtual machine. Retrieved December 21, 2009, from <http://www.dalvikvm.com/>
- Dedekind. (2009, January 12). Memory technology devices. *Linux Memory Technology Devices FAQ*. Retrieved December 21, 2009, from <http://www.linux-mtd.infradead.org/faq/general.html>
- Hoog, A. (2009a, March 16). Input/output error trying to dd Android /dev/block devices. *viaForensics Web site*. Retrieved December 21, 2009, from <http://viaforensics.com/forum/android-forensics/inputoutput-error-trying-to-dd-android-devblock-devices/>
- Hoog, A. (2009b, October 19). Android browser stores passwords and other sensitive data in plain text. *viaForensics Web site*. Retrieved December 21, 2009, from <http://viaforensics.com/android-forensics/android-browser-stores-passwords-sensitive-data-plain-text.html>
- HTC. (2009). HTC Sense user interface [Video]. *HTC Web site*. Retrieved December 21, 2009, from <http://www.htc.com/us/content/interactive/mediagallery/htc-sense.flv>
- Lomas, N. (2009, March 6). Android could overtake iPhone by 2012. *BusinessWeek Online*. Retrieved December 21, 2009, from [http://www.businessweek.com/globalbiz/content/mar2009/gb2009036\\_886305.htm](http://www.businessweek.com/globalbiz/content/mar2009/gb2009036_886305.htm)

Manning, C. (2002, September 20). YAFFS The NAND-specific flash file system. Retrieved December 21, 2009, from <http://www.yaffs.net/yaffs-nand-specific-flash-file-system-introductory-article>

Micro Systemation. (2008, July 1). .XRY system. *Micro Systemation Web site*. Retrieved December 21, 2009, from <http://www.msab.com/en/mobile-forensic-products/XRY-Mobile-Version-Forensic-Software/>

Miller, R. (2009, June 25). HTC's Sense UI not coming to any "Google" branded phones. *engadget Web site*. Retrieved December 21, 2009, from <http://www.engadget.com/2009/06/25/htcs-sense-ui-not-coming-to-any-google-branded-phones/>

Open Handset Alliance (OHA). (2009). Open handset alliance home page. Retrieved December 21, 2009, from <http://www.openhandsetalliance.com>

Paraben Corp. (2008). Paraben's Device Seizure - Cell phone forensic software. *Paraben Forensic Tools Web site*. Retrieved December 21, 2009, from [http://www.paraben-forensics.com/cell\\_models.html](http://www.paraben-forensics.com/cell_models.html)

Purdy, K. (2009, August 21). Five great reasons to root your Android phone. *lifehacker Web site*. Retrieved December 21, 2009, from <http://lifehacker.com/5342237/five-great-reasons-to-root-your-android-phone>

SSI Embedded Systems. (2008). Embedded Linux - Managing flash memory. *SSI Embedded Systems Programming Web site*. Retrieved December 21, 2009, from [http://www.ssiembedded.com/embedded\\_linux\\_managing\\_memory.html](http://www.ssiembedded.com/embedded_linux_managing_memory.html)

Spencer, S. (2009, July 24). Android appliances on the horizon. *PocketGamer.biz Web site*. Retrieved December 21, 2009, from <http://www.pocketgamer.biz/r/PG.Biz/Android/news.asp?c=14567>

TalkForensics. (2009, September 27). *Andrew Hoog of viaForensics talks about Android forensics* [Audio Podcast]. Retrieved December 21, 2009, from <http://www.blogtalkradio.com/show.aspx?userurl=TalkForensics&year=2009&month=09&day=27&url=Andrew-Hoog-of-viaForensics-talks-about-Android-forensics>

The Unlockr.com. (2009, November 7). How to: Root your Sprint HTC Hero. Retrieved December 21, 2009, from <http://theunlockr.com/2009/11/07/how-to-root-your-cdma-htc-hero-sprint-verizon/>

ZenThought. (2009). ASRoot2 software. *ZenThought.org Web site*. Retrieved December 21, 2009, from <http://zenthought.org/tmp/asroot2>