Android file system recovering

P. V. Burenin (*mymesbox@mail.ru*)

July 23, 2011

This file describes the process of solving "Scenario 1: Suspicious Death" task.

1 Processing TAR image

Use the algorithm to analyze case1.tgz image : Make "Ungzip" the archive to create case1.tar by standard functions. Make "Untar" function by AndroidProcessor.exe to create following files:

Table 1: Images in Case1.tar

Size	**File name**
2197	cmd1.txt
3173	cmd2.txt
0	dir.txt
1572864	mtdblock0.img
393216	mtdblock1.img
4718592	mtdblock3.img
147193856	mtdblock4.img
97124352	mtdblock5.img
274464768	mtdblock6.img
2097152	mtdblock7.img
336	postAcquisitionHashes.txt
16039018496	SDCard.img
45	SDCard.md5
1856	SDCardHashed.txt
598	SOC-log-may11-2.txt
622	SOC-log-may11.txt

Next step is to determine assignment of each image file (see the beginning of "mtdblock3.img" file):

```
console=ttyS2,115200n8 rw mem=244M@0x80C00000 init=/init ip=off
brdrev=P3A_CDMA mtdparts= omap2-nand.0:640k@128k(mbm), 384k@1408k(cdt),
384k@3328k(lbl), 384k@6272k(misc), 3584k(boot), 4608k(recovery), 143744k(system),
94848k(cache), 268032k(userdata),2m(kpanic)
```

Table 2: Images in Case1 project

Image name	Description
mtdblock1.img	Boot
mtdblock3.img	Recovery
mtdblock4.img	System
mtdblock5.img	Cache
mtdblock6.img	userdata
mtdblock7.img	kpanic
SDCard.img	FAT partition

2 Processing file system images

Now analyze each image:

SDCard.img. Description: whole file system can be restored by the disk imaging software like WinHex. Let's see the directory structure:

Table 3: SDCard.img directory structure

File name
Android
DCIM
download
lst.txt
Android/data
Android/data/com.cooliris.media
Android/data/com.cooliris.media/cache
Android/data/com.cooliris.media/cache/hires-image-cache66588715174662715_1024.cache
Android/data/com.cooliris.media/cache/local-album-cachechunk_0
Android/data/com.cooliris.media/cache/local-album-cacheindex
Android/data/com.cooliris.media/cache/local-image-thumbschunk_0
Android/data/com.cooliris.media/cache/local-image-thumbsindex
Android/data/com.cooliris.media/cache/local-meta-cachechunk_0
Android/data/com.cooliris.media/cache/local-meta-cacheindex
Android/data/com.cooliris.media/cache/local-video-thumbsindex
Android/data/com.cooliris.media/cache/picasa-thumbsindex
DCIM/.thumbnails
DCIM/Camera

File name
DCIM/.thumbnails/.thumbdata3–1967290299
DCIM/.thumbnails/1304706535900.jpg
DCIM/.thumbnails/1304707417417.jpg
DCIM/.thumbnails/image_last_thumb
DCIM/Camera/2011-05-06 14.43.35.jpg
download/2201-4.pdf
download/2201-7.pdf
download/2201-8.pdf
download/2201-9.pdf
download/2228-10.pdf
download/2228-11.pdf
download/2228-12.pdf
download/2228-15.pdf
download/2228-7.pdf
download/292048878.jpg

There are many pdf-documents with schematic diagrams in this image:

Table 4: PDF-documents list

File name
download/2201-4.pdf
download/2201-7.pdf
download/2201-8.pdf
download/2201-9.pdf
download/2228-10.pdf
download/2228-11.pdf
download/2228-12.pdf
download/2228-15.pdf
download/2228-7.pdf

and 2 pictures:

Table 5: Pictures file list

File name
download/292048878.jpg
DCIM/Camera/2011-05-06 14.43.35.jpg

mtdblockX.img. YAFFS2 files like mtdblock4.img, mtdblock6.img don't have spare information so the only one way to restore this images is not to use standard utilities (also in this images there is no "inblock" spare information all because of the image creating process mistake). Thus we need to do following operations to recover these images. First of all, determine the size of each chunk as 0x800. As there is no spare data - so the next chunk offset starts from the current offset plus 0x800.

The information of mtdblock4.img can be "FULL"-restored but it seems don't have a critical data. As this image don't have a many-writable data so the most of files can be restored! Then we must focus on mtdblock6.img.

Use following algorithm to read the image in general:

1. Start to read an image. Set the current data block size as 0. If this is "header" of YAFFS2 than store this information (filename, size, parent are used to determine different files with the same name in different folders, "IsShrink" flag).

2. Read the next chunk. If this is the "header" go to step 3. If this is not the header (to determine the "header" this use following assumption: "crc" is 0xFFFF, a filename consists of "legal-print" symbols and it't length is less than 256) - store its start data, increment its size to 0x800. Repeat step 2.

3. If data block size >0 , header-"type" is FILE and "IsShrink" $=0$ than store this block associated with this header (filename, modify time, filesize, etc). If "IsShrink" = 0xFFFFFFFF than this block is associated with upper header. Also if a block size is bigger than a file size than store the closest part to the current header block. Skipped part of the block must be saved to SkippedBlocks.dat file.

NB: This algorithm can't recover *ALL the files* information (because of a spare information deficit) but it recovers the majority of all files. Also the most complex heuristic algorithm exists that can restore database images. This algorithm based on YAFFS2 model and SQLite file structure.

Three are three kind of files:

- a raw-concatenated image (RCI);
- a full restored image (FRI);
- a full restored file generation (FRG).

We use standard access functions to work with FRI and FRG. Because of prevalence a text formatted data in SQLite databases we can use AndroidProcessor.exe to get messages from RCI(try "Extract" button with appropriate parameters).

All information of the image is stored by the "Process" button in AndroidProcessor.exe (NB: this program skips "cc_data" files because of their amount) to the file yaffs.csv (the "semi"-separated text file). This command produce the following directory system:

```
LST
    |— filename.extension
        |— filename.extension
        |— filename.extension.(1)
        |— filename.extension.(2)
        |— . . .
        |— filename.extension.(n)
        |— filename.extension.(full)
        |— filename.extension.log
    |— yaffs.csv
    |— afs_list.log
!WFD
```

First of all, inspect each file folder: "filename.extension". It consists of file blocks arranged by "modify time" of the file. So these blocks show us the history of the file. The first modification of the file is "filename.extension", next "filename.extension.(1)" and so on. In order to analyze whole file information we must inspect "filename.extension.(full)". This is a "raw-concatenated" image of the file with block separator strings: "———— END OF WHOLE BLOCK hhhhhhhh ————". It helps to gather all possible text information from the file. The process of file blocks creation is logged to "filename.extension.log". It stores following information for each block of the file:

- *Block offset*
- *Block size*
- *File size*
- *Flag "isShrink"*
- *Time of last modification*

Analysis of YAFFS2 and positioning of the block in the image helps us to make the following assumption: if the last block of a file is equal or bigger than the file size, than this is the FULL file image. So long as YAFFS2 is "true log file system" we assume that the chunks file have the straight order. Now we can restore these files - see folder "!WFD" (the whole file directory). "!WFD" folder discover us "the world" of the Android's user data file system. In this folder we find databases with a web-serf history, history of downloads and so on.

File "yaffs.csv" stores the following information about all blocks.

- *Image offset*
- *Type*
- *Parent*
- *Name*
- *Size*
- *Create time*

- *Access time*
- *Modify time*
- *Modify time string*
- *Flag1*
- *Flag2*
- *Flag3*
- *"IsShrink" flag*

This file helps us to understand the filesystem modification process and collect information about file blocks. Let's see an example of such a file:

Table 6: PDF documents and schematics

Image offset, Type, Parent, Name,Size, Create time, Access time Modify time, ModifyT,Flag1, Flag2, Flag3, IsShrink
0x005bb800;1;0x000002bf;gmail.db-journal;0x00000000;0x4dc6df1d; 0x4dc6df1d;0x4dc6df1d;08.05.2011 18:21:17 ; 0; -1; 0; 0
0x005bc000;3;0x000001b6;databases;0xffffffff;0x4dc6df1d;0x4dc1decb; 0x4dc6df1d;08.05.2011 18:21:17 ; 0; -1; 0; 0
0x005bc800;0;0x00000000;'——— DATA;0x00001000;0x0;0x0;0x0
0x005bd800;1;0x000002bf;gmail.db-journal;0x00000a10;0x4dc6df1d; 0x4dc6df1d;0x4dc6df1d;08.05.2011 18:21:17 ; 0; -1; 0; 0
0x005be000;0;0x00000000;'——— DATA;0x00000800;0x0;0x0;0x0

Field "Name" contains a file name for YAFFS2 header structure and "— — DATA" for a data block. It shows us the information about file blocks. AndroidProcessor.exe also produces a folder with all file blocks. Each folder contains the file block description file - "filename.extension.log".

AndroidProcessor.exe also produce a folder with all file blocks. In each folder there is a file block description file - "filename.extension.log".

3 Analysis of the content of the gathered files

First of all, try to define a file assignment by the file name. As soon it's produced, folder "LST" stores all file names in the directory structure. Let's focus our attention on the following files:

- mmssms.db, mmssms.db-journal
- contacts.db, contacts.db-journal
- mailstore.norby441@gmail.com.db, mailstore.norby441@gmail.com.db-journal

mmssms.db. It's seems there's no simple way to compose file blocks into a working file. Than we need to restore a message context by text-gathering function. Work with "mmssms.db.(full)"

Table 7: Mesages from mmssms.db

Message strings
FORWARDED SMS from 6245 at 20110505T173426America/New_York(4, 124,-14400,1, 1304631266) :shandra@cheerful.com (Nearby! Coming for my beer) Hey Yob, I am closing in on Fat Heads. See ya soon.
FORWARDED SMS from 6245 at 20110506T095308America/New_York(5, 125,-14400,1, 1304689988) :sms.dynadel@gmail.com Reminder, planned IT outage this weekend. This maintenance window will start at 3 PM today and continue for approx 48 hours.
FORWARDED SMS from 6245 at 20110506T095455America/New_York(5, 125,-14400,1, 1304690095) :sms.dynadel@gmail.com This effects external services such as website, email, webmail, and the ftp server. Use the secondary email access and helpdesk # for emergencies
FORWARDED SMS from 6245 at 20110505T173426America/New_York(4, 124,-14400,1, 1304631266) :shandra@cheerful.com (Nearby! Coming for my beer) Hey Yob, I am closing in on Fat Heads. See ya soon.
the implementation seems to be working ok, no gold yet though software seems to be working, I was a little worried given the source and short timeline
FORWARDED SMS from 6245 at 20110507T135649America/New_York(6, 126,-14400,1, 1304791009) :shandra@cheerful.com (Save me!) If Luke asks, I'm going out with you to dinner, OK?
I just can't face Mr. Smooth tonight.
FORWARDED SMS from 6245 at 20110507T190532America/New_York(6, 126,-14400,1, 1304809532) :shandra@cheerful.com (Re: Or you can walk down) Walking down now. Hope you are still vertical.
FORWARDED SMS from 6245 at 20110507T190749America/New_York(6, 126,-14400,1, 1304809669) :shandra@cheerful.com (Re: Or you can walk down) Hope you guys are still at double-wide ...
Got some results, I think we need to up the fee, say double?
FORWARDED SMS from 6245 at 20110510T032619America/New_York(2, 129,-14400,1, 1305012379) :shandra@cheerful.com (You around for lunch) Hey – a few of us are go ing to that great Indian buffet for lunch today. You interested?
FORWARDED SMS from 6245 at 20110510T083450America/New_York(2, 129,-14400,1, 1305030890) :shandra@cheerful.com (Re: Sorry, still an Atlanta till) OK. Safe travels!
I just sent you a sample, I think you'll be pleased...
Got some results, I think we need to up the fee, say double?

Message strings
You are joking, right? You can't seriously think about changing the deal now.

mailstore.norby441@gmail.com.db. This file contains Norby's email messages.

Table 8: Mesages from mailstore.norby441@gmail.com.db-journal

EMail messages addresses, themes, text
"norb k" <norby441@gmail.com>"Mr E" <mre@hushmail.com> Sample this is just a taste, much more where this came from.N.
"Gmail Team" <mail-noreply@google.com>"norb k" <norby441@gmail.com> Customize Gmail with colors and themes To spice up your inbox with colors and themes, check out the Themes tab under...
"norb k" <norby441@gmail.com>"Mr E" <mre@hushmail.com> showing i'm serious This information is obviously very valuable. I'd like to keep our relationship, but others would be willing to pay more. This information is obviously very valuable.
"" <mre@hushmail.com>"norb k" <norby441@gmail.com> Re: showing i'm serious I certainly don't want you giving these files to someone else. Expect a call from me shortly.

"web-serf". Analysis temporary files, downloads and "twitter.db" drives us to several contacts:

- Swift Logic, Bob Warr , Robert Warr, P.E.(LinkedIn)
- Suzy Welch, Lib Dems, Willie Mays, David Archuleta, . . . (twitter)

Also there are Facebook and Twitter posts in temporary files, downloads and "twitter.db".

Table 9: Twitter messages

Twitter message
now i get the txt about the outage. great. thanks guys.
Holy crap, just found out there is a planned maintenance outage starting this afternoon! Time to go tell the IT depth to reschedule...
Just found out Nationalpublicgardensday.org is today! Free admission to Phipps!

Twitter message
Anyone up for Greek today?
Look at all the beer! Fatheads, already one of my favorite places
I just installed the new Twidroyd for lunch time, finally. Maybe I should get a gyro, its Greek week!
OMG 10:15 already! Got to get to work early tommorrow, six after working late! At least Verizon was still open!
30 levels into bubble blast, smartphones are awesome!
Yay! Just picked up my new android smartphone! !!!

downloads.db. Inspect some of downloaded files (downloads.db.(69), downloads.db):

Table 10: Downloads

Download database items
http://@50.56.29.109:80/ss/2228-11.pdf
http://@50.56.29.109:80/ss/2228-10.pdf
http://v10.lscache3.c.android.clients.google.com/packages/data/ota/verizon _voles/e48e48ff4252.signed-voles-FRG01B-from-ESE81.e48e48ff.zip
https://android.clients.google.com/market/download/Download?userId= 16236448762498406323&deviceId=2494994194133853365&downloadId= 5035208920056425739&assetId=market-client-update: Ax3MNhn2VIaZrM0sruBpElJnszGxDukGOtVXNJyLY2 _VS3oH_bkmM9CCfQdCLYldUvXPWPYbI4rN 4G9-21KkPw__fapZ-J4CH7z8DDfc1YI&sig= AOGrW-wAAAAATc13rRlxXq J9cUJDfZwnEoNtcERAnmdQ
http://@50.56.29.109:80/ss/2201-8.pdf
http://@50.56.29.109:80/ss/2201-7.pdf

Some of these downloaded files are very interesting (they are stored in "SDCard/download" folder). The file named "SDCard/download/2228-11.pdf" stores scheme of an old Russian car "ВАЗ 2106" aka "Жигули шестёрка"...

Device configuration AndroidProcessor.exe get following device state files:

Table 11: Device state files

File name	Description
accounts.xml	System's accounts with ids
android.accounts. AccountAuthenticator.xml	System's accounts with ids for Twitter and Facebook
deviceName	System's name
packages.xml	All installed packages

File name	Description
persist.sys.country, persist.sys.language, persist.sys.timezone	Locale settings
vending_preferences.xml	

"deviceName" file contains following string: "droid1304551085337". This is the content of "accounts.xml" file.

```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<accounts>
<authority id="0" account="norby441@gmail.com" type="com.google" authority="com.android.contacts" />
<authority id="1" account="norby441@gmail.com" type="com.google" authority="gmail-ls" />
<authority id="2" account="norby441@gmail.com" type="com.google" authority="subscribedfeeds" />
<authority id="3" account="norby441@gmail.com" type="com.google" authority="com.cooliris.picasa.contentprovider" syncable="unknown" />
<authority id="4" account="norby441@gmail.com" type="com.google" authority="calendar" />
</accounts>
```

Following data is a part of "packages.xml" file (list applications installed).

```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<packages>
<permission-trees>
<item name="com.google.android.googleapps.permission.GOOGLE_AUTH" package="com.google.android.googleapps" />
</permission-trees>
<permissions>
<item name="android.permission.CHANGE_WIFI_MULTICAST_STATE" package="android" />
<item name="com.google.android.googleapps.permission.GOOGLE_AUTH.android" package="com.google.android.googleapps" />
<item name="com.google.android.googleapps.permission.GOOGLE_AUTH.orkut" package="com.google.android.googleapps" />
...
<shared-user name="com.google.android.marvin.feedback" userId="10003">
<sigs count="1">
<cert index="5" />
</sigs>
<perms>
<item name="android.permission.READ_PHONE_STATE" />
<item name="android.permission.VIBRATE" />
<item name="android.permission.WRITE_EXTERNAL_STORAGE" />
</perms>
</shared-user>
</packages>
```

Some of gathered (from "packages.xml" file) certificates are stored in "Certs" folder. A part of a email message with PGP-sign was found (see "Mail/signed-_part_of_email_to_mre.html").

```
com.amazon.mp3.cert
com.android.mms.cert
com.android.soundrecorder.cert
com.facebook.katana.cert
com.google.android.location.cert
com.google.android.marvin.soundback.cert
com.google.android.providers.talk.cert
com.twitter.android.cert
com.vzw.vvm.androidclient.cert
```

The content of "packages.xml" file gives us information about all installed applications and packages of the device. Also downloads.db file stores information about a software of the device: "signed-voles-FRG01B-from-ESE81.e48e48ff.zip" (Version 2.2 (FRG01B). As new software installed and no "root" programs on the device it seems the device "unrootted" and "adb" is not enabled (before May 11, 2011).

Contacts2.db. The content of this file helps to gather information about contacts:

Table 12: Contacts

Phone	**Name**	**Contact data**
	Mr E	mre@hushmail.com
4439264768	Taog	443-926-4768
6245	Shandra	shandra@cheerful.com
4124393388		4124393388@VTEXT.COM
	Norby's mail	norby441@gmail.com

To inspect the user "just-in-date" activity parse the following FRI files (arranged by dates and stored in "DeviceConfiguration" folder):

- usage-19700101
- usage-20110504
- usage-20110505
- usage-20110506
- usage-20110508
- usage-20110511 (this file stores the activity of an "Expert")

This information gives us answers to the user activity. Let's see the following listing (the last modification of usage-20110508):

```
com.android.mms
com.android.mms.ui.ComposeMessageActivity
com.android.mms.ui.ConversationList
com.android.browser
com.android.browser.BrowserActivity
com.android.browser.BrowserDownloadPage
com.google.android.systemupdater
com.google.android.systemupdater.SystemUpdateInstallDialog
com.qo.android.gep
com.qo.android.am.pdflib.app.RenderScreen
com.qo.android.quickoffice.QuickofficeDispatcher
com.android.launcher
com.android.launcher.Launcher
com.android.settings
com.android.settings.wifi.WifiSettings
com.android.settings.Settings
com.android.settings.WirelessSettings
com.android.contacts
com.android.contacts.ViewContactActivity
com.android.contacts.ui.EditContactActivity
com.android.contacts.ContactsListActivity
com.android.contacts.DialtactsActivity
com.android.phone
com.android.phone.InCallScreen
com.google.android.gm
com.google.android.gm.HtmlConversationActivity
com.google.android.gm.LabelsActivity
com.google.android.gm.ConversationListActivityGmail
com.google.android.gm.MailboxSelectionActivity
com.google.android.gm.ConversationListActivity
```

4 Solution folder information

The following table describes the content of the solution folder.

Table 13: Directory

Directory	**Description**
Certs	Some of certificates used by Android applications
Contacts	Contact information database parts
DeviceInformation	Device information files. There are ExtractedFiles.mtdblock4.img.zip and ExtractedFiles.mtdblock6.img.zip files. This files store gathered file information from mtdblock6.img and mtdblock4.img images.
DeviceUsage	Log of the user activity
FileExtractor	AndroidProcessor.exe results of image processing
MMSSMS	MMS and SMS database parts
Mail	EMail database parts
Project	AndroidProcessor.exe project in Visual C++
SDCard	SDCard files

5 Conclusion

Norby's phone images contain the dialog that has several contradictions between the work and the payment (the dialog with Mr. E). Also we understand he had a very valuable information that could be a real motive of the murder.

But, there was a single bullet to the head. The professional homicide (from "organized criminal group called KRYPTIX") should make even one control shot but he didn't.