# A Geeks Guide to Digital Forensics

*or: How I learned to stop worrying and love the hex editor*

VIAFORENSICS

innovative digital forensics and security

# Qualifications

| | |
|---|---|
| **Background** | Computer scientist, prev CIO, co-founder of viaForensics |
| **Author** | Two books on mobile forensics and security |
| **Researcher** | Two patents pending in security and forensics |
| **Forensics:** | Multiple certifications, expert in Federal and State courts |
| Geek | Avid Linux user since 1995 (e.g. compile kernel for Soundblaster) |

# What is Digital Forensics?

- Branch of forensic science – uses scientific method

- The preservation, recovery, analysis and reporting of digital artifacts including information stored on:
  - Computer/laptop systems (hard drives)
  - Storage media (USBs, CDs, DVDs, cameras, etc.)
  - Mobile phones
  - Electronic documents

- Typically used reactively, move toward proactive
  - Reactive: court cases, incident response
  - Proactive: mobile app security audits, continuous forensic monitoring

# Storage Devices

There are 3 main types of storage devices used today:

1. Hard-disk drive (HDD) – Contains a spinning magnetic drive used to store non-volatile data.
2. Solid-state drive (SSD) – Contains internal microchips for the purpose of storing non-volatile data.
3. NAND Flash memory
    1. Typically found in smart phones, USB thumb drivers and other portable devices
    2. Not removable like typical HDD or SSD
    3. Very unique characteristics from standard HDD (limited writes/erase)
    4. In constant state of change (FTL)

# Acquisition strategies

Forensics Analysts can acquire/receive data 3 different ways

- Backup Files
  - Backup files are provided from the “custodian”. This could include backup software from corporations, PST file, iTunes backup, etc.
- Logical Acquisition
  - A copy of the file system is created (i.e. tar.gz of / or recursive copy that preserves date/time)
- Physical Acquisition
  - Creates an exact digital replica of the storage medium
  - Can recover deleted data
  - This process requires specialized analysis tools and techniques
  - Drive management firmware may still affect acquisition (FTL, bad blocks, etc.)

# Image Verification

- Hash value – A calculated hex signature based on a set of data.
  - A hash value can be used to verify forensic image integrity. One slight change in source will cause "avalanche" effect in hash value
  - In order to prove that two data sets are identical, their hash values must match.
  - In some instances, hash values are not stable (NAND Flash) so a hash of the data as it's extracted is taken but won't necessarily match if source is imaged again
- Common hash techniques
  - md5 (128-bit value)
  - sha256 (256-bit value)
- md5 of "Andrew Hoog" = 9bdbad9aecd74fce6e6bb48ee18100b8

# How to acquire a forensic image

- If possible, connect drive to a physical write blocker
  - This prevents any writes to the drive
  - There are software techniques but not as effective
  - Generally, impossible with NAND Flash devices

- Forensically acquire device with software
  - Open source: dd, dcfldd and dc3dd (we use the later)
  - Free: FTK Imager and many others
  - Commercial: FTK, EnCase, etc.

- Perform verification of source and image with hash signature and record in Chain of Custody

# Example imaging with dc3dd

- Department of Defense's Cyber Crime Center dc3dd
  - Patched version of GNU dd
  - includes a number of features useful for forensics
  - Free and open source
- Command:
  - dc3dd if=<source device> of=drive001.dc3dd verb=on hash=sha256 hlog=drive001.hashlog log=drive001.log rec=off
  - rec=off determines how to handle I/O errors (recover=off)
  - Full details: http://dc3dd.sourceforge.net/
    - ./configure; make; sudo make install

```
[3610506.275511] sd 7:0:0:0: [sde] 976773168 512-byte logical blocks: (500 GB/465 GiB)
[3610506.276398] sd 7:0:0:0: [sde] Write Protect is on
[3610506.276404] sd 7:0:0:0: [sde] Mode Sense: 10 00 80 00
[3610506.276408] sd 7:0:0:0: [sde] Assuming drive cache: write through
[3610506.279379] sd 7:0:0:0: [sde] Assuming drive cache: write through
[3610506.279386]  sde: sde1 sde2
[3610506.331605] sd 7:0:0:0: [sde] Assuming drive cache: write through
[3610506.331611] sd 7:0:0:0: [sde] Attached SCSI disk
```

# Handling failing drives

- May run into drive issues, have to decide how to handle
  - Stop on error
  - Continue, fill with NULLs (0x00)
  - Skip (would result in smaller dd image, not recommended)

- Example of errors:

```
[3698052.155258] sd 7:0:0:0: [sde] Result: hostbyte=DID_OK driverbyte=DRIVER_SENSE
[3698052.155266] sd 7:0:0:0: [sde] Sense Key : Aborted Command [current]
[3698052.155272] sd 7:0:0:0: [sde] Add. Sense: No additional sense information
[3698052.155278] end_request: I/O error, dev sde, sector 720361992
[3698052.155285] Buffer I/O error on device sde, logical block 90045249
```

- Potential workaround
  - GNU ddresuce – very powerful alternative, install from source
  - Will rescue blocks, read drive backwards, restart where last left off
  - http://www.gnu.org/software/ddrescue/ddrescue.html

# “Typical” forensic analysis steps

1. Create timeline of events
    1. File system modified, accessed, changed and created
    2. Metadata from files (images, documents, flash cookies, etc)
2. Mount dd image read-only
3. Generate list of all files (allocated and deleted)
4. Analyze key files
    1. Windows: Registry, LNK files, user profile, web history, etc.
    2. Linux: Bash history, .recently-used.xbel, gvfs-metadata, etc.
5. Recover deleted files
6. File carving (handles unallocated)
7. Search files, dd image, etc.
8. Many specialized techniques

# Analyzing forensic image (F/OSS)

- The Sleuth Kit by Brian Carrier
  - Brain author of excellent book File System Forensics Analysis (FSFA)
  - Actively maintained, just released 3.2.2 (06/13/2011)
  - Supports NTFS, FAT, UFS 1, UFS 2, EXT2FS, EXT3FS, and ISO 9660
  - http://sleuthkit.org/
- Programs to start with:
  - mmls – Media Management ls, generally partition info:

```
ahoog@linux-wks-002:~$ sudo mmls /dev/sdb
DOS Partition Table
Offset Sector: 0
Units are in 512-byte sectors

     Slot    Start        End          Length       Description
00:  Meta    0000000000   0000000000   0000000001   Primary Table (#0)
01:  -----   0000000000   0000000062   0000000063   Unallocated
02:  00:00   0000000063   0562307129   0562307067   Linux (0x83)
03:  Meta    0562307130   0586067264   0023760135   DOS Extended (0x05)
04:  Meta    0562307130   0562307130   0000000001   Extended Table (#1)
05:  -----   0562307130   0562307192   0000000063   Unallocated
06:  01:00   0562307193   0586067264   0023760072   Linux Swap / Solaris x86 (0x82)
07:  -----   0586067265   0586072367   0000005103   Unallocated
```

# TSK – File system info

- fsstat – File system information:

```
ahoog@linux-wks-002:~$ fsstat ./webos-secure-erase/var.dd
FILE SYSTEM INFORMATION
--------------------------------------------
File System Type: Ext3
Volume Name: /var
Volume ID: f967a0c0313889b736486f05d6ba4fbe

Last Written at: Wed Jun 24 06:38:28 2009
Last Checked at: Wed Jun 24 03:47:30 2009

Last Mounted at: Wed Jun 24 06:38:28 2009
Unmounted properly
Last mounted on:

Source OS: Linux
Dynamic Structure
Compat Features: Journal, Ext Attributes, Resize Inode, Dir Index
InCompat Features: Filetype, Needs Recovery,
Read Only Compat Features: Sparse Super, Has Large Files,

Journal ID: 00
Journal Inode: 8

METADATA INFORMATION
--------------------------------------------
Inode Range: 1 - 65537
Root Directory: 2
Free Inodes: 65314

<snip>
```

# TSK – listing (all) files

- fls – Forensic list
  - Power utility which can list allocated/deleted files
  - Provides offset so recovery is possible
  - Build MACB for timeline analysis
  - fls -z CST6CDT -s 0 -m '/' -f ext3 -r -o 63 -i raw file.dd > body

```
ahoog@linux-wks-002:~$ fls -m 'C:/' -r -i raw disk001.dd | less
0|C:/$AttrDef|4-128-4|r/rr-xr-xr-x|48|0|2560|1287497006|1287497006|1287497006|1287497006
0|C:/$BadClus|8-128-2|r/rr-xr-xr-x|0|0|0|1287497006|1287497006|1287497006|1287497006
0|C:/$BadClus:$Bad|8-128-1|r/rr-xr-xr-x|0|0|20957978624|1287497006|1287497006|1287497006|1287497006
0|C:/$Bitmap|6-128-1|r/rr-xr-xr-x|0|0|639592|1287497006|1287497006|1287497006|1287497006
0|C:/$Boot|7-128-1|r/rr-xr-xr-x|48|0|8192|1287497006|1287497006|1287497006|1287497006
0|C:/$Extend|11-144-4|d/dr-xr-xr-x|0|0|344|1287497006|1287497006|1287497006|1287497006
0|C:/$Extend/$ObjId:$O|25-144-5|r/rr-xr-xr-x|0|0|152|1287515747|1287497007|1287497007|1287497007
0|C:/$Extend/$Quota:$O|24-144-3|r/rr-xr-xr-x|0|0|88|1287497007|1287497007|1287497007|1287497007
0|C:/$Extend/$Quota:$Q|24-144-2|r/rr-xr-xr-x|0|0|208|1287497007|1287497007|1287497007|1287497007
0|C:/$Extend/$Reparse:$R|26-144-2|r/rr-xr-xr-x|0|0|48|1287515734|1287497007|1287497007|1287497007
0|C:/$LogFile|2-128-1|r/rr-xr-xr-x|0|0|67108864|1287497006|1287497006|1287497006|1287497006
0|C:/$MFT|0-128-1|r/rr-xr-xr-x|0|0|11911168|1287497006|1287497006|1287497006|1287497006
0|C:/$MFTMirr|1-128-1|r/rr-xr-xr-x|0|0|4096|1287497006|1287497006|1287497006|1287497006
0|C:/$Secure:$SDS|9-128-8|r/rr-xr-xr-x|0|0|295364|1287497006|1287497006|1287497006|1287497006
0|C:/$Secure:$SDH|9-144-11|r/rr-xr-xr-x|0|0|168|1287497006|1287497006|1287497006|1287497006
0|C:/$Secure:$SII|9-144-14|r/rr-xr-xr-x|0|0|152|1287497006|1287497006|1287497006|1287497006
0|C:/$UpCase|10-128-1|r/rr-xr-xr-x|0|0|131072|1287497006|1287497006|1287497006|1287497006
0|C:/$Volume|3-128-3|r/rr-xr-xr-x|48|0|0|1287497006|1287497006|1287497006|1287497006
0|C:/ADFS|10921-144-1|d/drwxrwxrwx|0|0|48|1287516828|1287516828|1287516828|1287516828
0|C:/AUTOEXEC.BAT|7357-128-1|r/rrwxrwxrwx|0|0|0|1287515644|1287515644|1287515644|1287515644
0|C:/Documents and Settings|3660-144-6|d/drwxrwxrwx|0|0|56|1287522035|1287515927|1287515927|1287497094
```

# mactime – make body file human friendly

- mactime -b body -z CST6CDT -d > timeline.csv
  - Takes body file and turns into CSV or other format

```
Date,Size,Type,Mode,UID,GID,Meta,File Name
Tue Oct 19 2010 09:03:26,11911168,macb,r/rr-xr-xr-x,0,0,0-128-1,C:/$MFT
Tue Oct 19 2010 09:03:26,4096,macb,r/rr-xr-xr-x,0,0,1-128-1,C:/$MFTMirr
Tue Oct 19 2010 09:03:26,131072,macb,r/rr-xr-xr-x,0,0,10-128-1,C:/$UpCase
Tue Oct 19 2010 09:03:26,344,macb,d/dr-xr-xr-x,0,0,11-144-4,C:/$Extend
Tue Oct 19 2010 09:03:26,67108864,macb,r/rr-xr-xr-x,0,0,2-128-1,C:/$LogFile
Tue Oct 19 2010 09:03:26,0,macb,r/rr-xr-xr-x,48,0,3-128-3,C:/$Volume
Tue Oct 19 2010 09:03:26,2560,macb,r/rr-xr-xr-x,48,0,4-128-4,C:/$AttrDef
Tue Oct 19 2010 09:03:26,639592,macb,r/rr-xr-xr-x,0,0,6-128-1,C:/$Bitmap
Tue Oct 19 2010 09:03:26,8192,macb,r/rr-xr-xr-x,48,0,7-128-1,C:/$Boot
Tue Oct 19 2010 09:03:26,20957978624,macb,r/rr-xr-xr-x,0,0,8-128-1,C:/$BadClus:$Bad
Tue Oct 19 2010 09:03:26,0,macb,r/rr-xr-xr-x,0,0,8-128-2,C:/$BadClus
Tue Oct 19 2010 09:03:26,295364,macb,r/rr-xr-xr-x,0,0,9-128-8,C:/$Secure:$SDS
Tue Oct 19 2010 09:03:26,168,macb,r/rr-xr-xr-x,0,0,9-144-11,C:/$Secure:$SDH
Tue Oct 19 2010 09:03:26,152,macb,r/rr-xr-xr-x,0,0,9-144-14,C:/$Secure:$SII
Tue Oct 19 2010 09:03:27,208,macb,r/rr-xr-xr-x,0,0,24-144-2,C:/$Extend/$Quota:$Q
Tue Oct 19 2010 09:03:27,88,macb,r/rr-xr-xr-x,0,0,24-144-3,C:/$Extend/$Quota:$O
Tue Oct 19 2010 09:03:27,152,m.cb,r/rr-xr-xr-x,0,0,25-144-5,C:/$Extend/$ObjId:$O
Tue Oct 19 2010 09:03:27,48,m.cb,r/rr-xr-xr-x,0,0,26-144-2,C:/$Extend/$Reparse:$R
Tue Oct 19 2010 09:04:21,208,...b,r/rr-xr-xr-x,0,0,3653-128-3,C:/boot.ini
Tue Oct 19 2010 09:04:54,56,...b,d/drwxrwxrwx,0,0,3660-144-6,C:/Documents and Settings
Tue Oct 19 2010 14:10:34,208,m...,r/rr-xr-xr-x,0,0,3653-128-3,C:/boot.ini
Tue Oct 19 2010 14:14:04,0,macb,r/rrwxrwxrwx,0,0,7354-128-1,C:/CONFIG.SYS
Tue Oct 19 2010 14:14:04,0,macb,r/rrwxrwxrwx,0,0,7357-128-1,C:/AUTOEXEC.BAT
Tue Oct 19 2010 14:14:12,208,..c.,r/rr-xr-xr-x,0,0,3653-128-3,C:/boot.ini
Tue Oct 19 2010 14:15:34,48,.a..,r/rr-xr-xr-x,0,0,26-144-2,C:/$Extend/$Reparse:$R
Tue Oct 19 2010 14:15:47,152,.a..,r/rr-xr-xr-x,0,0,25-144-5,C:/$Extend/$ObjId:$O
Tue Oct 19 2010 14:18:47,56,m.c.,d/drwxrwxrwx,0,0,3660-144-6,C:/Documents and Settings
Tue Oct 19 2010 14:33:48,48,macb,d/drwxrwxrwx,0,0,10921-144-1,C:/ADFS
Tue Oct 19 2010 14:56:30,0,...b,r/rrwxrwxrwx,0,0,11602-128-3,C:/body
Tue Oct 19 2010 15:24:20,208,.a..,r/rr-xr-xr-x,0,0,3653-128-3,C:/boot.ini
Tue Oct 19 2010 16:00:35,56,.a..,d/drwxrwxrwx,0,0,3660-144-6,C:/Documents and Settings
```

# Mount dd image read-only

- Determine file system offset in dd image:

```
ahoog@ubuntu:~/sd-emmc/$ mmls sdcard2-113serialno.dc3dd
DOS Partition Table
Offset Sector: 0
Units are in 512-byte sectors

     Slot    Start        End          Length       Description
00:  Meta    0000000000   0000000000   0000000001   Primary Table (#0)
01:  -----   0000000000   0000000128   0000000129   Unallocated
02:  00:00   0000000129   0003911679   0003911551   DOS FAT16 (0x06)
```

- Mount FAT16 (and many others f/s) partition read only:

```
ahoog@ubuntu:~$ mkdir -p ~/mnt/sdcard

ahoog@ubuntu:~$ sudo mount -t vfat -o loop,ro,offset=66048 sdcard2-113serialno.dc3dd ~/mnt/sdcard

ahoog@ubuntu:~$ mount | grep vfat
/dev/loop0 on /home/ahoog/mnt/sdcard type vfat (ro,offset=66048)

ahoog@ubuntu:~$ df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/sda1              19G   18G  570M  97% /
<snip>
/dev/mtdblock0         64M  1.2M   63M   2% /home/ahoog/mnt/yaffs2
/dev/loop0            1.9G  244M  1.7G  13% /home/ahoog/mnt/sdcard
```

- Perform additional analysis on files

# Log2timeline

- Kristinn Gudjonsson developed this software
  - Written in Perl (trying to convince him to move to Python)
  - Extracts timeline artifacts from many file types including
    - Evt/extx, registry, $MFT, prefetch, browser history, etc. (46 and climbing)
  - 10+ export formats
  - http://log2timeline.net/

- timescanner -d ~/mnt/sdcard -z CST6CDT -w body.ts

- If you output in body format, can combine with TSK's fls output and generate full timeline of file system and file metadata

# Regripper

- Harlan Carvey developed this software
  - Written in Perl
  - Windows is primary platform, there is a Linux port
  - Parses Windows registry files
    - Support hives: NTUSER.dat, system, software, sam, security, etc.
  - http://regripper.wordpress.com/regripper/

```
ahoog@linux-wks-002:/mnt/wip/via/src/regripper$ /mnt/wip/via/src/regripper/rip.pl
Rip v.20080419 - CLI RegRipper tool - Linux version

Rip [-r Reg hive file] [-f plugin file] [-p plugin module] [-l] [-h]

Parse Windows Registry files, using either a single module, or a plugins file.
All plugins must be located in the "plugins" directory; default plugins file
used if no other filename given is "plugins\plugins".

  -r Reg hive file...Registry hive file to parse
  -g .................Guess the hive file (experimental)
  -f [plugin file]...use the plugin file (default: plugins\plugins)
  -p plugin module...use only this module
  -l .................list all plugins
  -c .................Output list in CSV format (use with -l)
copyright 2008 H. Carvey [This linux version modified by Daniele Murrau
```

# Scalpel

- Download scalpel src at:
  - wget http://www.digitalforensicssolutions.com/Scalpel/scalpel-2.0.tar.gz
- Compile
  - tar xzvf scalpel-2.0.tar.gz
  - cd scalpel-2.0/
  - ./configure; make
  - sudo cp scalpel /usr/local/bin
- Run scalpel

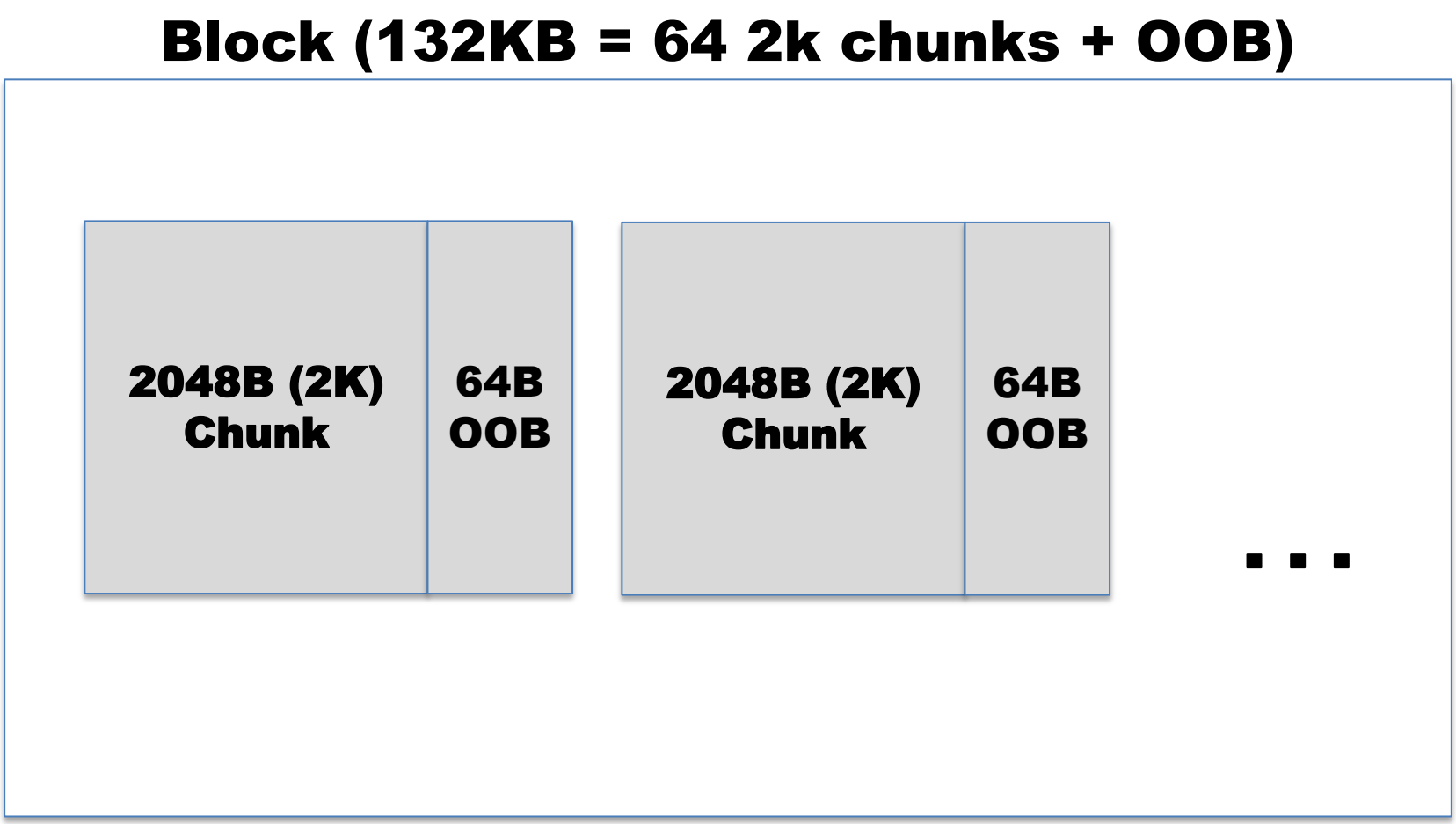
  $ scalpel -c scalpel.conf ~/Desktop/image.dd

  $ scalpel -c android-scalpel.conf ~/Desktop/android-image.nanddump
- Examine data in “scalpel-output” directory

# Android Flash Memory

- Android devices use a raw flash device, and therefore need a Flash Transition Layer (FTL)
  - FTL provides basic block interface to developers
  - Handles wear leveling, bad block management, metadata, etc.
- FTL is provided by Memory Technology Device (MTD)
  - MTD is open source
  - Newer Android devices are moving to eMMC where FTL controller is embedded with the memory (similar to thumb drives and SSD)
- MTD divides memory into blocks, each of which is 128K with a 64 byte Out-of-Band (OOB) area
  - OOB houses YAFFS2 tags, meta data, bad blocks and ECC

# YAFFS2 – Block/Chunk/OOB diagram



# Android Forensics

- Logical recovery can be achieved through Content Providers
  - We've developed free tool for law enforcement: AFLogical
  - Commercial: viaExtract - http://viaforensics.com/products/viaextract/
- Beyond CPros
  - To extract more data, we first need to escalate privileges on the device.
    - This presentation is not intended to cover these techniques (a.k.a. get a Google Dev phone or go read XDA)
- Logical Acquisition
  - With escalated privileges, we can simply connect to the device using the Android Debug Bridge (adb) and execute an adb pull command on the files that we wish to acquire. (i.e. /data/data)

# Android Forensics – Physical acquisition

- Physical Acquisition
  - –Android dd image
    - The dd utility on Android devices is only capable of reading the non-OOB data from the YAFFS2 MTD partition
  - – Full NAND image
    - Includes OOB
    - We use an in-house developed nanddump utility capable of reading and extracting all data from the YAFFS2 partition (and dealing with bad blocks)
    - Allows an examiner to take full advantage of the YAFFS2 features, primarily artifacts from being a log-structure file system

# YAFFS2 Timeline

```
nanddump -c /dev/mtd0ro | grep -v "00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00" |
grep -v "ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff" | less

0x00006800: 10 00 00 00 10 10 00 00 ff ff 66 96 c6 56 13 e2  |.........file1.|
0x00006810: 47 87 47 00 00 00 00 00 00 00 00 00 00 00 00 00  |txt.............|
0x00006900: 00 00 00 00 00 00 00 00 00 00 ff ff ff 18 00 00  |................|
0x00006910: d6 00 00 00 57 00 00 00 63 99 d5 d4 d7 99 d5 d4  |m...u...6.]M}.]M|
0x00006920: 42 a9 d5 d4 51 00 00 00 ff ff ff ff ff ff ff ff  |$.]M............|
0x000069c0: ff ff ff ff ff ff ff ff ff ff ff ff 00 00 00 00  |................|
0x000069e0: ff ff ff ff ff ff ff ff 00 00 00 00 ff ff ff ff  |................|
0x000069f0: ff ff ff ff ff ff ff ff 00 00 00 00 00 00 00 00  |................|
  OOB Data: ff ff 10 01 00 00 20 10 00 01 10 10 00 08 51 00  |................|
  OOB Data: 00 00 51 af e2 e2 10 00 00 00 ef ff ff ff ff ff  |................|
  OOB Data: ff ff ff ff ff ff ff ff ff 00 3c ff 3c ff ff ff  |................|
  OOB Data: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff  |................|
```

- Number as written to NAND flash: 63 99 d5 d4 (0x6399d5d4)
- Converted from little endian to big endian: 4d 5d 99 36 (0x4d5d9936 which is the hex read from right to left)
- Converting 0x4d5d9936 (hex) to base 10 is 1297979702
- Unix time stamp 1297979702 in human date time format is Thu Feb 17 15:55:02 CST 2011 (date -d @1297979702)

# YAFFS2 Timeline

- Using this information, we can isolate a number of important artifacts
  - atime (accessed time) for a directory along with mtime and ctime
  - Object ID to the directory within the OOB
  - Object ID for files and cross-reference to make sure it is consistent with debug data.

- Additional analysis would allow us to create the MAC times for each file and directory on the NAND.

- It is also possible to gather additional meta data information from ObjectHeaders found on the NAND.

# Proactive forensics

- Forensics has typically been used reactively
- By moving forensic techniques into proactive security services, excellent results are achieved
  - appWatchdog: basic security testing for mobile apps
    - http://viaforensics.com/appwatchdog/
  - Mobile app security: see online presentation
    - http://viaforensics.com/computer-forensics/mobile-app-security-presentation-andrew-hoog.html
  - liveForensics: continuous forensic monitoring of key assets
    - http://viaforensics.com/services/security/liveforensics/

# Contact viaForensics

Andrew Hoog

Chief Investigative Officer

ahoog@viaforensics.com


http://viaforensics.com

Main Office:

1000 Lake St, Suite 203

Oak Park, IL 60301

Tel: 312-878-1100 | Fax: 312-268-7281

VIAFORENSICS

innovative digital forensics and security