# iPhone Forensics with F/OSS

*A HOWTO for iPhone Forensics with free and/or open source tools*

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) |

# Presentation Goals

iPhone Forensics with F/OSS tools

- Commercial Tools exist but there are a growing number of F/OSS tools
- A Mac (OSX) or Linux workstation is used for many of these programs
- Focus on step-by-step examples

# iPhone Backup Analyzer

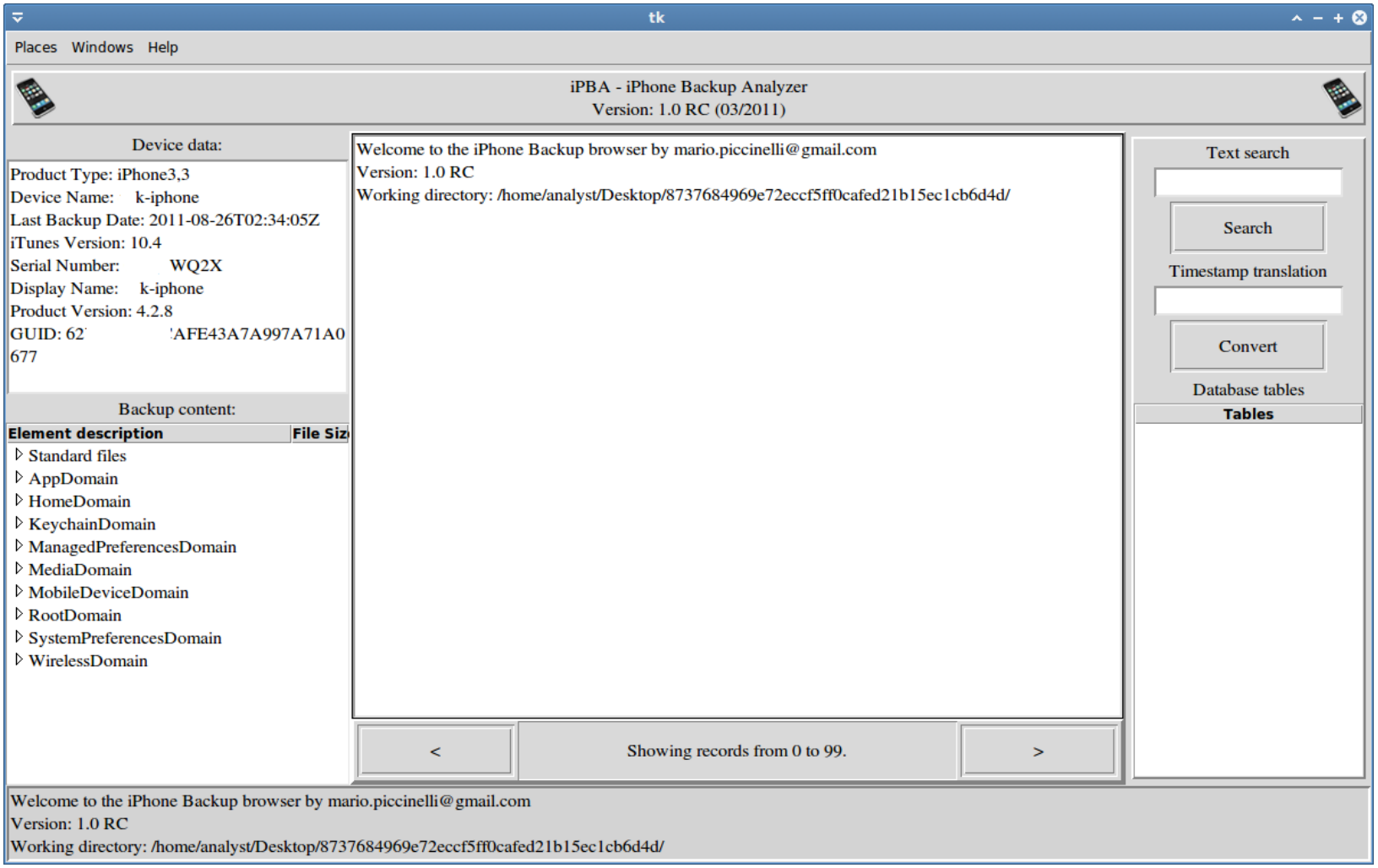
Open source (MIT) iPhone backup analyzer by Mario Picci (http://ipbackupanalyzer.com/)

- Decodes files, presents in a hierarchical view, has some search and conversions
- Plist files are shown (binary plist files are automatically converted in ascii format)
- Image files are shown
- SQLite files are shown with the list of the tables they contain. By clicking on the tables list the selected table's content is dumped in the main UI
- Unknown data files are shown as hex/ASCII data

**iTunes Backup Directories**

**Mac Os X**: ~/Library/Application Support/MobileSync/Backup/
**Windows XP**: \Documents and Settings\(username)\Application Data\Apple Computer\MobileSync\Backup\
**Windows Vista, Windows 7**: \Users\(username)\AppData\Roaming\Apple Computer\MobileSync\Backup\

# iPhone Backup Analyzer



# iPhone Backup Analyzer – Linux Install

On Ubuntu Workstation

---

sudo apt-get update

sudo apt-get install python-tk python-imaging python-imaging-tk git

Install pyttk

- Download: http://pypi.python.org/pypi/pyttk/
- Extract: tar xzvf pyttk-0.3.2.tar.gz
- cd pyttk-0.3.2/
- Install: sudo python setup.py install

git clone git://github.com/PicciMario/iPhone-Backup-Analyzer

cd iPhone-Backup-Analyzer/

./main.py -d ~/Desktop/8737684969e72eccf5ff0cafed21b15ec1cb6d4d/

# Zdziarski's iOS forensic tools

Free for qualified law enforcement and government agencies

- Based on F/OSS software and research (Cyanide, etc)
- Physical acquisition
- Logical acquisition
- PIN bypass
- Decrypts the encrypted files / slice
  - iOS 3.x: fully decrypt slice, gets unallocated
  - iOS 4.x: decrypts files, not unallocated (mostly)
- Decrypt Keychain
- Working on recovering deleted keys

# iOS 4 Encryption defeated with F/OSS

- @0naj iphone-dataprotection tools (Python and C)
  - Brute force PIN code on device
  - Recover device encryption keys
  - Decrypt the keychain, all dataprotection encrypted files
  - Scrape the HFS journal for deleted content
  - Decrypt the entire raw disk
  - Included with Jonathan Zdziarski's toolset, or available separately to developers:
    - http://code.google.com/p/iphone-dataprotection/

# Mount the dmg image read-only (Linux)

- Determine file system offset in dd image:

```
ahoog@linux-wks-003:~$ mmls item001.dc3dd
GUID Partition Table (EFI)
Offset Sector: 0
Units are in 512-byte sectors

     Slot    Start        End          Length       Description
00:  Meta    0000000000   0000000000   0000000001   Safety Table
01:  -----   0000000000   0000000039   0000000040   Unallocated
02:  Meta    0000000001   0000000001   0000000001   GPT Header
03:  Meta    0000000002   0000000033   0000000032   Partition Table
04:  00      0000000040   0000409639   0000409600   EFI system partition
05:  01      0000409640   0488134983   0487725344   Customer
06:  -----   0488134984   0488397167   0000262184   Unallocated

Then take 409640 * 512 to get offset of 209735680.
```

- Mount HFS partition read only:

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

ahoog@ubuntu:~$ sudo mount -t hfsplus -o ro,loop,offset=209735680 item001.dc3dd ~/mnt/hfs/

If iPhone from Zdziarski's toolset:
ahoog@ubuntu:~$ sudo mount -t hfsplus -o ro,loop iPhone-3g-313.dmg ~/mnt/hfs/
```

# Mount the dmg image read-only (Linux)

- Make sure file system was mounted

```
analyst@ubuntu:~$ mount
/dev/sda1 on / type ext4 (rw,errors=remount-ro,commit=0)
proc on /proc type proc (rw,noexec,nosuid,nodev)
none on /sys type sysfs (rw,noexec,nosuid,nodev)
fusectl on /sys/fs/fuse/connections type fusectl (rw)
none on /sys/kernel/debug type debugfs (rw)
none on /sys/kernel/security type securityfs (rw)
none on /dev type devtmpfs (rw,mode=0755)
none on /dev/pts type devpts (rw,noexec,nosuid,gid=5,mode=0620)
none on /dev/shm type tmpfs (rw,nosuid,nodev)
none on /var/run type tmpfs (rw,nosuid,mode=0755)
none on /var/lock type tmpfs (rw,noexec,nosuid,nodev)
none on /proc/fs/vmblock/mountPoint type vmblock (rw)
/dev/loop0 on /home/analyst/mnt/hfs type hfsplus (ro)
```

- Can check disk usage

```
analyst@ubuntu:~$ df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/sda1              19G  5.2G   13G  29% /
none                  243M  200K  243M   1% /dev
none                  249M  148K  249M   1% /dev/shm
none                  249M  100K  249M   1% /var/run
none                  249M     0  249M   0% /var/lock
.host:/               931G  682G  249G  74% /mnt/hgfs
/dev/loop0            7.1G  629M  6.5G   9% /home/analyst/mnt/hfs
```

# 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/
- Install:

```
Download sleuthkit:
http://sleuthkit.org/sleuthkit/download.php
tar xzvf sleuthkit-3.2.2.tar.gz
cd sleuthkit-3.2.2/
./configure
make
sudo make install
```

# TSK – Linux install

- 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 info

```
analyst@ubuntu:/mnt/hgfs/Desktop$ fsstat iPhone-3g-313.dmg
FILE SYSTEM INFORMATION
--------------------------------------------
File System Type: HFSX
File System Version: HFSX
Case Sensitive: yes

Volume Name: Data
Volume Identifier: f2aaasa2a44e9

Last Mounted By: Mac OS X
Volume Unmounted Improperly
Mount Count: 13328222

Creation Date:              Sun Feb  7 12:13:16 2010
Last Written Date:          Tue Sep 13 10:45:59 2011
Last Backup Date:           Wed Dec 31 18:00:00 1969
Last Checked Date:          Sun Feb  7 06:13:16 2010

METADATA INFORMATION
--------------------------------------------
Range: 2 - 2377984
Bootable Folder ID: 0
Startup App ID: 0
Startup Open Folder ID: 0
Mac OS 8/9 Blessed System Folder ID: 0
Mac OS X Blessed System Folder ID: 0
Number of files: 1535
Number of folders: 260

CONTENT INFORMATION
--------------------------------------------
Block Range: 0 - 1854341
Total Range in Image: 0 - 1854340
Allocation Block Size: 4096
Number of Free Blocks: 1693385
```

# TSK – forensic 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
  - analyst@ubuntu:/mnt/hgfs/Desktop$ fls -z CST6CDT -s 0 -m '/' -f hfs -r -i raw iPhone-3g-313.dmg > ~/iPhone-timeline.body

```
0|/$ExtentsFile|3|r/r---------|0|0|4194304|0|0|0|0
0|/$CatalogFile|4|r/r---------|0|0|8388608|0|0|0|0
0|/$BadBlockFile|5|r/r---------|0|0|0|0|0|0|0
0|/$AllocationFile|6|r/r---------|0|0|233472|0|0|0|0
0|/$StartupFile|7|r/r---------|0|0|0|0|0|0|0
0|/$AttributesFile|8|r/r---------|0|0|8388608|0|0|0|0
0|/^^^^HFS+ Private Data|16|d/d---------|0|0|0|1265544796|1265544796|1265544796|1265544796
0|/^^^^HFS+ Private Data/temp2377964|2377964|r/rrw-------|501|0|512|1315928734|1315928759|1315928759|1315928734
0|/^^^^HFS+ Private Data/temp2377965|2377965|r/rrw-------|501|0|4096|1315928734|1315928740|1315928740|1315928734
0|/.HFS+ Private Directory Data^|17|d/dr-xr-xr-t|0|0|0|1265544796|1265544796|1265544796|1265544796
0|/CommCenter|575|d/drwx------|0|0|0|1269656622|1269656622|1269656622|1269656622
0|/CommCenter/spool|576|d/drwx------|0|0|0|1269656622|1315928721|1315928721|1269656622
0|/CommCenter/spool/MobileOriginated|577|d/drwx------|0|0|0|1269656622|1315257710|1315257710|1269656622
0|/CommCenter/spool/MobileTerminated|578|d/drwx------|0|0|0|1269656622|1315407199|1315407199|1269656622
0|/Keychains|18|d/drwxr-xr-x|64|0|0|1261418823|1315428289|1315428289|1261125197
0|/Keychains/TrustStore.sqlite3|318|r/rrw-------|64|0|8192|1265545345|1265545345|1265545345|1265545345
0|/Keychains/keychain-2.db|209|r/rrw-------|64|0|53248|1265545280|1315058674|1315058674|1265545280
0|/Keychains/ocspcache.sqlite3|6945|r/rrw-------|64|0|100352|1270345718|1314266964|1314266964|1270345718
```

# mactime – make body file human friendly

- analyst@ubuntu:/mnt/hgfs/Desktop$ mactime -b ~/iPhone-timeline.body -z CST6CDT -d > ~/iPhone-timeline.csv
  - Takes body file and turns into CSV or other format

```
Date,Size,Type,Mode,UID,GID,Meta,File Name
Date,Size,Type,Mode,UID,GID,Meta,File Name
Sun Dec 31 2000 18:00:00,15872,ma.b,r/rrw-r--r--,501,501,241,"/mobile/Library/Caches/SpringBoardIconCache/com.apple.WebSheet"
Sun Dec 31 2000 18:00:00,3712,ma.b,r/rrw-r--r--,501,501,242,"/mobile/Library/Caches/SpringBoardIconCache-small/com.apple.WebSheet
Sun Dec 31 2000 18:00:00,15872,ma.b,r/rrw-r--r--,501,501,255,"/mobile/Library/Caches/SpringBoardIconCache/com.apple.DemoApp"
Sun Dec 31 2000 18:00:00,3712,ma.b,r/rrw-r--r--,501,501,256,"/mobile/Library/Caches/SpringBoardIconCache-small/com.apple.DemoApp"
Sun Dec 31 2000 18:00:00,15872,ma.b,r/rrw-r--r--,501,501,259,"/mobile/Library/Caches/SpringBoardIconCache/com.apple.fieldtest"
Sun Dec 31 2000 18:00:00,3712,ma.b,r/rrw-r--r--,501,501,260,"/mobile/Library/Caches/SpringBoardIconCache-small/com.apple.fieldtes
Sun Dec 31 2000 18:00:00,15872,ma.b,r/rrw-r--r--,501,501,277,"/mobile/Library/Caches/SpringBoardIconCache/com.apple.springboard"
Sun Dec 31 2000 18:00:00,3712,ma.b,r/rrw-r--r--,501,501,278,"/mobile/Library/Caches/SpringBoardIconCache-small/com.apple.springbo
Thu Aug 21 2008 02:36:52,92,m..b,r/rrw-r--r--,0,0,60,"/root/Library/Preferences/.GlobalPreferences.plist"
Fri Dec 18 2009 02:21:47,0,...b,d/drwxrwxrwt,0,0,64,"/tmp"
Fri Dec 18 2009 02:21:48,0,...b,d/drwxr-xr-x,0,0,35,"/logs/Baseband"
Fri Dec 18 2009 02:21:49,0,m..b,d/drwxr-x---,0,0,57,"/root"
Fri Dec 18 2009 02:21:49,0,...b,d/drwxr-x---,0,0,58,"/root/Library"
Fri Dec 18 2009 02:33:13,0,m..b,d/drwxr-xr-x,0,3,28,"/empty"
Fri Dec 18 2009 02:33:13,0,m..b,d/drwxr-xr-x,0,0,29,"/folders"
Fri Dec 18 2009 02:33:13,0,m..b,d/drwxr-xr-x,0,0,54,"/msgs"
Fri Dec 18 2009 02:33:13,0,ma.b,r/rrw-r--r--,0,0,55,"/msgs/bounds"
Fri Dec 18 2009 02:33:14,0,...b,d/drwxrwxr-x,0,1,62,"/run"
Fri Dec 18 2009 02:33:14,0,m..b,d/drwxr-xr-x,0,0,65,"/vm"
Fri Dec 18 2009 02:33:16,0,m..b,d/drwxr-xr-x,501,501,27,"/ea"
Fri Dec 18 2009 02:33:17,0,...b,d/drwxr-xr-x,64,0,18,"/Keychains"
Fri Dec 18 2009 02:33:17,0,m..b,d/drwxr-xr-x,0,0,19,"/Managed Preferences"
Fri Dec 18 2009 02:33:17,0,...b,d/drwx------,501,501,20,"/Managed Preferences/mobile"
Fri Dec 18 2009 02:33:17,0,...b,d/drwxr-xr-x,0,0,21,"/MobileDevice"
```

# 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/

- Install log2timeline on Ubuntu 10.10 (lucid)
  - sudo add-apt-repository "deb http://log2timeline.net/pub/ lucid main"
  - wget -q http://log2timeline.net/gpg.asc -O- | sudo apt-key add -
  - sudo apt-get update
  - sudo apt-get install log2timeline-perl

# Log2timeline

- sudo timescanner -d /home/analyst/mnt/hfs/ -z CST6CDT -w ~/iPhone-log2timeline.csv
  - 218 artifacts (either files or directories).
  - Run time of the script 24 seconds.

- If you output in body format, can combine with TSK's fls output and generate full timeline of file system and file metadata (sometimes referred to as a "Super Timeline"

# 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/
  - sudo apt-get install libtre-dev libtre5
  - ./configure; make
  - sudo cp scalpel /usr/local/bin
- Run scalpel

  $ scalpel -c ~/scalpel.conf iPhone-3g-313.dmg
- Examine data in “scalpel-output” directory

# Sample scalpel.conf

```
# Author: Andrew Hoog [ahoog at viaforensics dot com]
# http://viaforensics.com/products/tools
# Name: scalpel-android.conf
#
# (c) Copyright 2011 viaForensics. All rights reserved.
# All software is provided as is and without warranty. License for personal and educational use is granted however
# commercial use is prohibited without further permission from viaForensics.

#              case    size    header                  footer
#extension   sensitive

        gif     y       5000000         \x47\x49\x46\x38\x37\x61        \x00\x3b
        gif     y       5000000         \x47\x49\x46\x38\x39\x61        \x00\x3b
        jpg     y       200000000       \xff\xd8\xff\xe0\x00\x10        \xff\xd9
        jpg     y       5000000         \xff\xd8\xff\xe1        \x7f\xff\xd9

        png     y       102400          \x50\x4e\x47?   \xff\xfc\xfd\xfe
        png     y       102400          \x89PNG

        sqlitedb        y       409600  SQLite\x20format

        email   y       10240   From:

        doc     y       10000000  \xd0\xcf\x11\xe0\xa1\xb1\x1a\xe1\x00\x00 \xd0\xcf\x11\xe0\xa1\xb1\x1a\xe1\x00\x00 NEXT
        doc     y       10000000  \xd0\xcf\x11\xe0\xa1\xb1

        htm     n       50000   <html                   </html>

        pdf     y       5000000 %PDF  %EOF\x0d  REVERSE
        pdf     y       5000000 %PDF  %EOF\x0a  REVERSE

        wav     y       200000  RIFF????WAVE
        amr     y       200000  #!AMR

        zip     y       10000000        PK\x03\x04      \x3c\xac

        java    y       1000000 \xca\xfe\xba\xbe
```

# Sample scalpel.conf

```
analyst@ubuntu:~$ scalpel -p /mnt/hgfs/Desktop/iPhone-3g-313.dmg -c ~/scalpel.conf
Scalpel version 2.0
Written by Golden G. Richard III and Lodovico Marziale.
Multi-core CPU threading model enabled.
Initializing thread group data structures.
Creating threads...
Thread creation completed.

Opening target "/mnt/hgfs/Desktop/iPhone-3g-313.dmg"

Image file pass 1/2.
/mnt/hgfs/Desktop/iPhone-3g-313.dmg: 100.0% |*********************************|    7.1 GB    00:00 Allocating work queues...
Work queues allocation complete. Building work queues...
Work queues built.  Workload:
gif with header "\x47\x49\x46\x38\x37\x61" and footer "\x00\x3b" --> 0 files
gif with header "\x47\x49\x46\x38\x39\x61" and footer "\x00\x3b" --> 15 files
jpg with header "\xff\xd8\xff\xe0\x00\x10" and footer "\xff\xd9" --> 303 files
jpg with header "\xff\xd8\xff\xe1" and footer "\x7f\xff\xd9" --> 107 files
png with header "\x50\x4e\x47?" and footer "\xff\xfc\xfd\xfe" --> 0 files
png with header "\x89PNG" and footer "" --> 1126 files
sqlitedb with header "SQLite\x20format" and footer "" --> 226 files
email with header "From:" and footer "" --> 491 files
doc with header "\xd0\xcf\x11\xe0\xa1\xb1\x1a\xe1\x00\x00" and footer "\xd0\xcf\x11\xe0\xa1\xb1\x1a\xe1\x00\x00" --> 0 files
doc with header "\xd0\xcf\x11\xe0\xa1\xb1" and footer "" --> 0 files
htm with header "<html" and footer "</html>" --> 117 files
pdf with header "%PDF" and footer "%EOF\x0d" --> 0 files
pdf with header "%PDF" and footer "%EOF\x0a" --> 0 files
wav with header "RIFF????WAVE" and footer "" --> 1 files
amr with header "#!AMR" and footer "" --> 14 files
zip with header "PK\x03\x04" and footer "\x3c\xac" --> 0 files
java with header "\xca\xfe\xba\xbe" and footer "" --> 0 files
** PREVIEW MODE: GENERATING AUDIT LOG ONLY **
** NO CARVED FILES WILL BE WRITTEN **
Carving files from image.
Image file pass 2/2.
/mnt/hgfs/Desktop/iPhone-3g-313.dmg: 100.0% |**********************|    7.1 GB    00:00 ETA Processing of image file complete.
Cleaning up...
Done.
Scalpel is done, files carved = 2400, elapsed  = 357 secs.
```

# Viewing image with streaming hex viewer

- Usage:

```
$ xxd iPhone-3g-313.dmg | less
```

- To auto skip 0's:

```
$ xxd -a iPhone-3g-313.dmg | less
```

```
0000000: 0000 0000 0000 0000 0000 0000 0000 0000  ................
*
0000400: 4858 0005 0000 0000 3130 2e30 0000 0000  HX......10.0....
0000410: c794 5edc ca95 2b37 0000 0000 c794 5edc  ..^...+7......^.
0000420: 0000 05ff 0000 0104 0000 1000 001c 4b86  ..............K.
0000430: 0019 d6c9 0000 6e30 0001 0000 0001 0000  ......n0........
0000440: 0024 4901 00cb 5f5e 0000 0000 0000 0001  .$I..._^........
0000450: 0000 0000 0000 0000 0000 0000 0000 0000  ................
0000460: 0000 0000 0000 0000 f2d4 1487 aa2a 44e9  .............*D.
0000470: 0000 0000 0003 9000 0003 9000 0000 0039  ...............9
0000480: 0000 0001 0000 0039 0000 0000 0000 0000  .......9........
```

# Hex editor

- Usage:

```
$ hexedit iPhone-3g-313.dmg
```

- Once in hex editor:
  - "/" = search hex/ASCII string (in "hexedit" use tab to change between ASCII and hex searches)
  - q = exit hex editor
  - h = help
- Can quickly locate potential evidence
- Other tools also available (hexeditor and many others)

# Grep Command

- Searches through a file (or many files/folders) for a specified keyword(s)

- Grep is case sensitive by default

```
$ grep amr iPhone-3g-313.dmg
```

- To do case-insensitive (more time consuming):

```
$ grep –i AmR iPhone-3g-313.dmg
```

- Can search for a phrase in quotes

```
$ grep “Trace File” iPhone-3g-313.dmg
$ grep -a "Trace File" iPhone-3g-313.dmg
$ grep -a -A 1 -B 1 "Trace File" iPhone-3g-313.dmg
```

# Grep Command (continued)

- Can also be used to search through many files

- Grep through all files in a user's home directory for "viaF":

```
analyst@ubuntu:~$ grep -R 312493 *
Binary file scalpel-output/sqlitedb-9-0/00001.db matches
Binary file scalpel-output/sqlitedb-9-0/00017.db matches
```

**Find all sms database files from iPhone (after scalpel)**

```
analyst@ubuntu:~$ grep -R svc_center sqlite*
```

# “Strings” Command

- Strings is a powerful utility to extract ASCII or Unicode strings from binary data

- Can be run against a file or a full disk image

```
$ strings iPhone-3g-313.dmg > iPhone.str
$ strings iPhone-3g-313.dmg | less
```

- Can also search for Unicode

```
$ strings -e b iPhone-3g-313.dmg | less
```

# “Strings” does more than ASCII

- Strings is designed to extract ASCII and Unicode
  - 7-bit ASCII, 8-bit ASCII
  - 16-bit big-endian and little-endian
  - 32-bit big-endian and little-endian

- From the strings manual page:

--encoding=encoding

Select the character encoding of the strings that are to be found. Possible values for encoding are: s = single-7-bit-byte characters (ASCII, ISO 8859, etc., default), S = single-8-bit-byte characters, b = 16-bit bigendian, l = 16-bit littleendian, B = 32-bit bigendian, L = 32-bit littleendian. Useful for finding wide character strings. (l and b apply to, for example, Unicode UTF-16/UCS-2 encodings).

# Decrypting data – step 1

- Scenario: imaged iPhone and application has encrypted data which you need to view.

- Our solution (but other approaches may work)
  - Noted app data was encrypted
  - Analyzed symbol table for app, saw entries such as:
    - 00091033 t -[NSData(AESAdditions) AES256DecryptWithKey:]
    - 00092015 t -[NSData(AESAdditions) AES256EncryptWithKey:]
    - 0009aA07e t -[NSData(AESAdditions) keyBytes:]
    - 00034261 t +[NSData(Base64) dataFromBase64String:]
    - 00034410 t -[NSData(Base64) base64EncodedString]

# Decrypting data – step 2

- Determined app stored key in Keychain so cracked the key chain, found an entry with a Base64 encoded key
- Decoded Base64 key
- Wrote quick program that used “AES256DecryptWithKey” API, encrypted file and decode AES encryption key to access data

- F/OSS Tools used:
  - Zdziarski’s techniques to physically image device, crack keychain
  - Strings to determine encryption technique
  - XCode from Apple to write decrypt program

# 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