6.1) Généralités

version : 12-11-2014 13:30

Sommaire :

Pourquoi analyser un smartphone ?

version : 12-01-2015 21:05

1. Un smartphone est utilisé pour attaquer :
   - un réseau d'entreprise
   - un réseau télécom
   - un réseau domestique
2. Un smartphone a été attaqué par :
   - un virus / cheval de troie / ...
   - une backdoor
   - l'installation d'un logiciel espion
3. Un smartphone a été retrouvé près :
   - d'une victime de meurtre
   - caché chez un particulier
   - dans un repère terroriste

Ces équipements sont intéressants par rapport aux caractéristiques suivantes :

- Ultra-Connecté : WIFI / GSM / NFC / Bluetooth / ...
- Petit
- Puissant

Difficultés dans l'analyse des smartphones

version : 14-10-2014 09:25

Pas de disque dur à analyser :

- Souvent chiffré
- Stockage non standardisé
- Interfaces de connexion non standards

Et le principe de non modification ?

Et la copie des media type disque dur ?

Et les données contenues dans le cloud ?

Exemple :

Lorsque la police saisit un smartphone, certains propriétaires utilisent la fonctionnalité d'effacement à distance pour effacer les données du téléphone. http://www.theinquirer.net/inquirer/news/2375230/criminals-remotely-wiping-mobile-devices-seized-as-police-evidence

Méthodologie idéale

version : 12-10-2014 16:33

Pour l'analyse du disque interne :

- démontage de l'ordiphone
- débrasage du composant
- lecture du composant et copie des données
- reconstruction du système de fichiers
- analyse des données

Pour l'analyse de la RAM :

- démontage à chaud de la coque
- connexion via un outil adéquate aux pins de la RAM
- copie de la RAM
- reconstruction de la RAM

- analyse des informations

Méthodologie réelle

version : 09-12-2015 18:34

Préparation

- préparer son analyse (gants, ...)
- prendre des photos
- bloquer l'émission et la réception

Sécurisation

Comment bloquer l'émission et la réception ?

- cage de Faraday,
- canette,
- mode avion,
- ...

Analyse

2 cas se présentent :

- Que faire si le téléphone est éteint ?
- Que faire si le téléphone est allumé ?

Téléphone éteint :

- Enlever la batterie
- Récupérer les numéros de série
- Démarrer le smartphone sur une ROM custom

Téléphone allumé :

Si le smartphone est **allumé/déverrouillé** :

- vérifier si il y a un mot de passe / PIN
- effectuer une acquisition logique

Si le smartphone est **allumé/verrouillé** :

- si le verrou est le code PIN opérateur
  - faire une demande à l'opérateur
- si le verrou vient de l'OS
  - "smudge attack"
  - force brute

Éléments essentiels à récupérer dans un smartphone

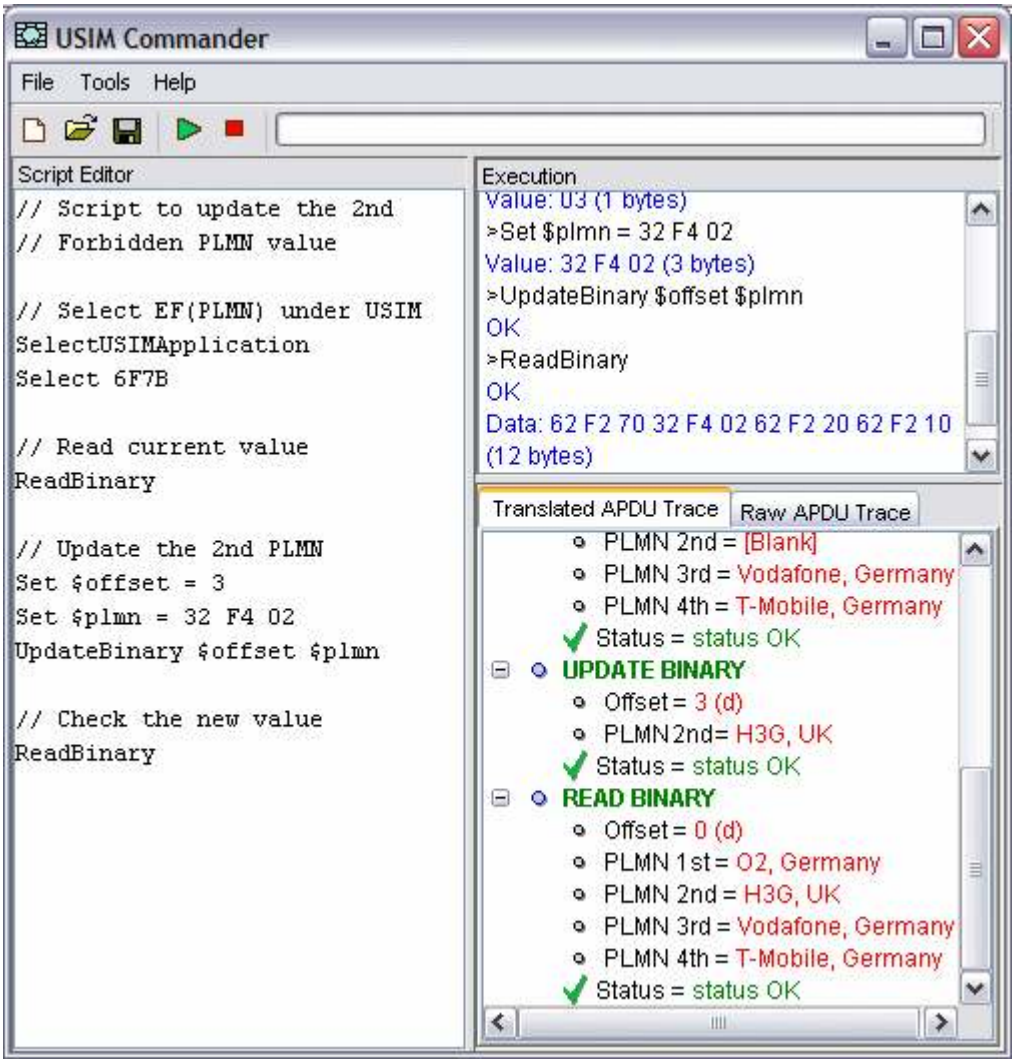
version : 15-01-2015 22:12

- Information général et configuration
- Contacts et leurs photos
- Labels personnalisés du répertoire
- Événements du calendrier
- Journal des appels
- Messages
- Photos avec coordonnées géographiques
- Clips vidéo
- Système de fichiers
- Cache et historique Google maps
- Données des comptes de messagerie et mails
- Configuration des applications tierces
- Positionnement géographique
  - Google maps history,
  - locations,
  - GPS data,
  - Wi-Fi networks
- Dictionnaire de l'utilisateur

Analyse des cartes (U)SIM

version : 12-10-2014 16:35

http://www.quantaq.com/usimcommander.htm



6.2) Android

version : 11-01-2015 13:44

Présentation

Android Présentation

version : 13-01-2015 21:34

Description

L'OS se base sur les **permissions d'un système Linux standard** (permissions « rwx » sur le système de fichiers). À cela, il faut ajouter un système de gestion des **droits transverses** qui se fait pour chaque application en appliquant à cette application un UID et GID particulier. Le cœur du système ainsi que les librairies internes sont sur une **partition en lecture seule**. Les applications tierces sont sur une partition en lecture/écriture séparées les unes des autres par le système UID/GID. La SDCard est une partition en lecture/écriture accessible à tout le monde sans aucune contrainte.

Architecture

- Noyau Linux
- Bac à sable
- IPC sécurisés
- Capacités
- Exe signés
- Chiffrement
- Code PIN et mot de passe

Android Debug Bridge : adb

ADB ne fonctionne que si le "Débogage USB" est sélectionné dans les options du smartphone

Récupération du numéro de série :

```
$ adb get-serialno
HT9ACL901794
```

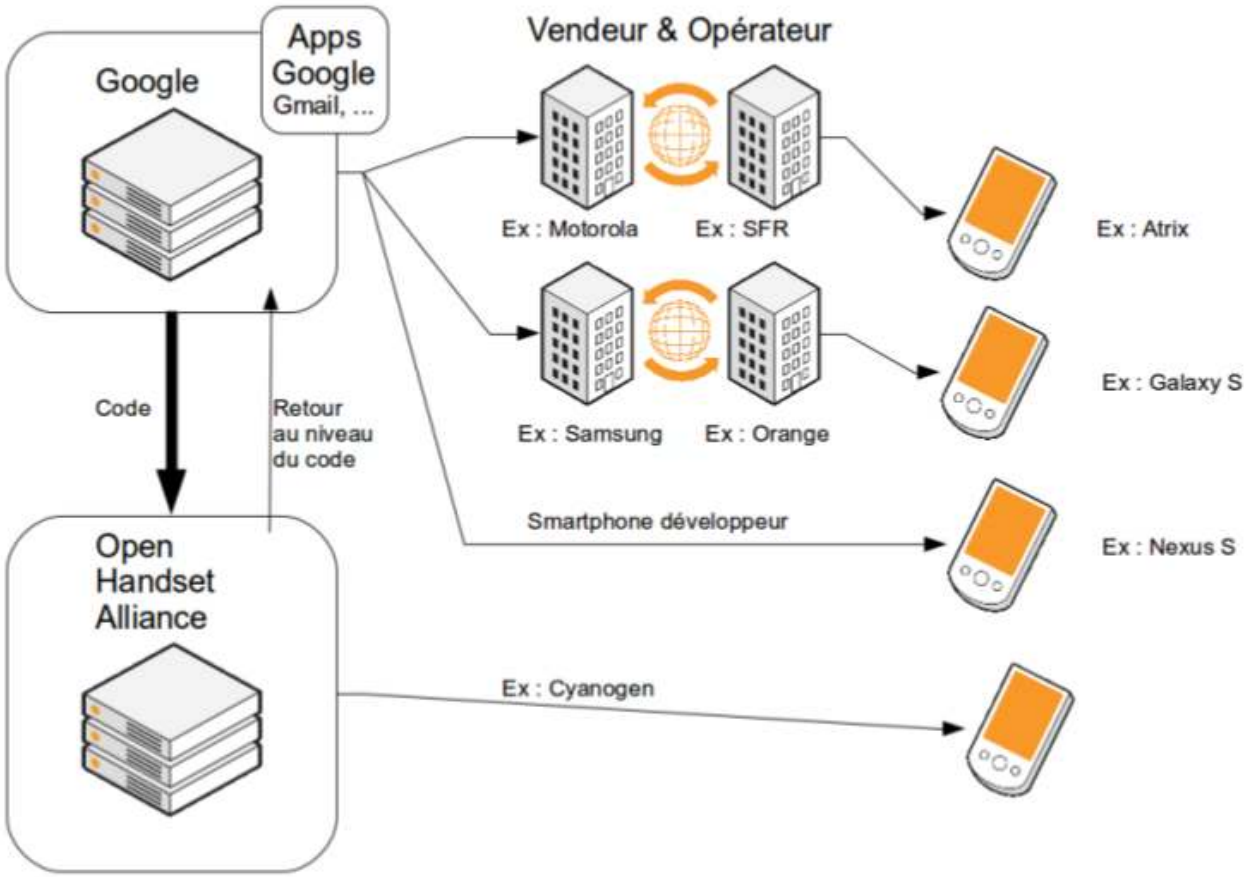
Récupération des logs :

```
$ adb logcat
V/WifiMonitor(  157): Event [CTRL-EVENT-CONNECTED -
Connection to 00:1d:6a:84:31:d2 completed (reauth) [id=0
id_str=]]
V/WifiStateTracker(  157): New network state is
CONNECTED
D/WifiStateTracker(  157): DHCP request started
D/WifiStateTracker(  157): DHCP succeeded with lease:
86400
V/WifiStateTracker(  157): IP configuration: ipaddr
192.168.1.19 gateway 192.168.1.1 netmask 255.255.255.0
dns1 192.168.1.1 dns2 192.168.1.1 DHCP server
192.168.1.1 lease 86400 seconds
D/WifiWatchdogService(  157):
(android.server.ServerThread) Hubble (00:1d:6a:84:31:d2)
does not require the watchdog
D/Tethering(  157): MasterInitialState.processMessage
what=3
D/CMStats (  341): CONNECTIVITY_ACTION: noConnectivity =
false
D/dalvikvm(  382): GC_EXPLICIT freed 103K, 49% free
2759K/5379K, external 0K/0K, paused 121ms
D/dalvikvm(  444): GC_EXPLICIT freed 26K, 51% free
2802K/5639K, external 0K/0K, paused 86ms
D/dalvikvm(  341): GC_EXPLICIT freed 10K, 52% free
2701K/5575K, external 0K/0K, paused 85ms
D/dalvikvm(  350): GC_EXPLICIT freed 20K, 50% free
2805K/5511K, external 0K/0K, paused 110ms
D/dalvikvm(  361): GC_EXPLICIT freed 273K, 52% free
2700K/5511K, external 0K/0K, paused 103ms
E/libEGL  (  157): called unimplemented OpenGL ES API
E/libEGL  (  157): called unimplemented OpenGL ES API
E/libEGL  (  157): called unimplemented OpenGL ES API
E/libEGL  (  157): called unimplemented OpenGL ES API
E/libEGL  (  157): called unimplemented OpenGL ES API
W/SharedBufferStack(  157):
```

```
waitForCondition(LockCondition) timed out (identity=13,
status=0). CPU may be pegged. trying again.
W/SharedBufferStack(  216):
waitForCondition(LockCondition) timed out (identity=3,
status=0). CPU may be pegged. trying again.
I/power   (  157): *** set_screen_state 0
D/LockPatternKeyguardView(  157): onScreenTurnedOff()
D/SurfaceFlinger(  157): About to give-up screen,
flinger = 0xcf2b0
D/AK8973  (  120): Compass CLOSE
```

Il existe plein d'autres commandes ADB

Distribution des mises à jour



Déploiement de l'OS Android depuis les serveurs de Google

Android Dead Forensics

version : 09-12-2015 18:35

Voici quelques méthodes de récupération :

- Quand on a accès au smartphone :
  - logique (synchronisation via une solution logiciel)
    - `adb backup...`
    - `adb shell ls -l /dev/block/platform/msm_sddc.1/by-name/` puis `adb shell dd if=... of=...`
  - physique, càd une copie bit-à-bit
  - insertion d'un module dans le noyau de l'Android
- Quand on n'a pas accès au smartphone :
  - utilisation des sauvegardes du smartphone (si l'utilisateur en fait)
- Quand le smartphone est verrouillé par un code PIN
  - Brute forcing du code PIN
    - Exemple de Brute forcing avec Santoku Linux
  - Récupération de la clé de chiffrement dans la RAM
    - Exemple FROST : http://www1.cs.fau.de/filepool/projects/frost/frost.pdf
  - suppression des fichiers (Android <= 4.2.2, à condition que le smartphone soit rooté) :
    - `/data/system/gesture.key`
    - `/data/system/password.key`

Quelques outils :

- Android-Forensics
- Oxygen Forensic Suite
- Santoku Linux
- MPE+
- Magnet IEF

Mots de passe

Le problème des mots de passe :

- Comparaisons entre mots de passe Android
- http://linuxsleuthing.blogspot.fr/2013/01/cracking-android-passwords-need-for_19.html

Android Live Forensics

version : 09-12-2015 22:48

L'acquisition est très compliquée, il faut être root, il faut donc rooter l'équipement !!!

Pour transférer l'exploit, on peut utiliser adb, fastboot, ou un outil du fabricant.

En étant sur le système :

```
mount -o remount,rw -t yaffs2 /dev/block/mtdblock3
/system
cd system
cd bin
cat sh > su
chmod 4755 su
```

puis copier la RAM via un module noyau :

- dmd (Droid Memory Dumper)
- https://github.com/504ensicsLabs/LiME

```
$ adb push lime.ko /sdcard/lime.ko
```

```
$ adb forward tcp:4444 tcp:4444
$ adb shell
$ su
#
# insmod lime path=tcp:4444
$ nc localhost 4444 > evo.dump
```

L'analyse peut ensuite s'effectuer via Volatility

```
# python volatility.py -f /mnt/data/volimgs/androidmem --profile=android li-
nux_mount

Volatile Systems Volatility Framework 1.4_rc1
/dev/block/mtdblock4  /system     yaffs2      ro,relatime
sysfs                 /sys        sysfs       rw,relatime
devpts                /dev/pts    devpts      rw,relatime
/dev/block/dm-1       /mnt/asec/com.rovio.angrybirds-1
                                  Vfat        ro,relatime,nosuid,nodev,noexec
proc                  /proc       proc        rw,relatime
none                  /dev/cpuctl cgroup      rw,relatime
tmpfs                 /mnt/sdcard/.android_secure
                                  tmpfs       ro,relatime
tmpfs                 /dev        tmpfs       rw,relatime
/dev/block/mtdblock6  /data       yaffs2      rw,relatime,nosuid,nodev
tmpfs                 /app-cache  tmpfs       rw,relatime
/dev/block/vold/179:1 /mnt/sdcard vfat        rw,relatime,nosuid,nodev,noexec
none                  /acct       cgroup      rw,relatime
tmpfs                 /mnt/asec   tmpfs       rw,relatime
/dev/block/vold/179:1 /mnt/secure/asec/.android_secure
                                  Vfat        rw,relatime,nosuid,nodev,noexec
/dev/block/mtdblock5  /cache      yaffs2      rw,relatime,nosuid,nodev
/dev/block/dm-0       /mnt/asec/com.com2us.sliceit-1
                                  Vfat        ro,relatime,nosuid,nodev,noexec
```

```
# python volatility.py -f /mnt/data/volimgs/androidmem --profile=android
                       linux_task_list_psaux -p 1

python volatility.py --profile=android -f /mnt/data/volimgs/android-full
linux_task_list_psaux
Volatile Systems Volatility Framework 1.4_rc1
Arguments                                                    Pid
init                                                         1
[kthreadd]                                                   2
[ksoftirqd/0]                                                3
[watchdog/0]                                                 4
```

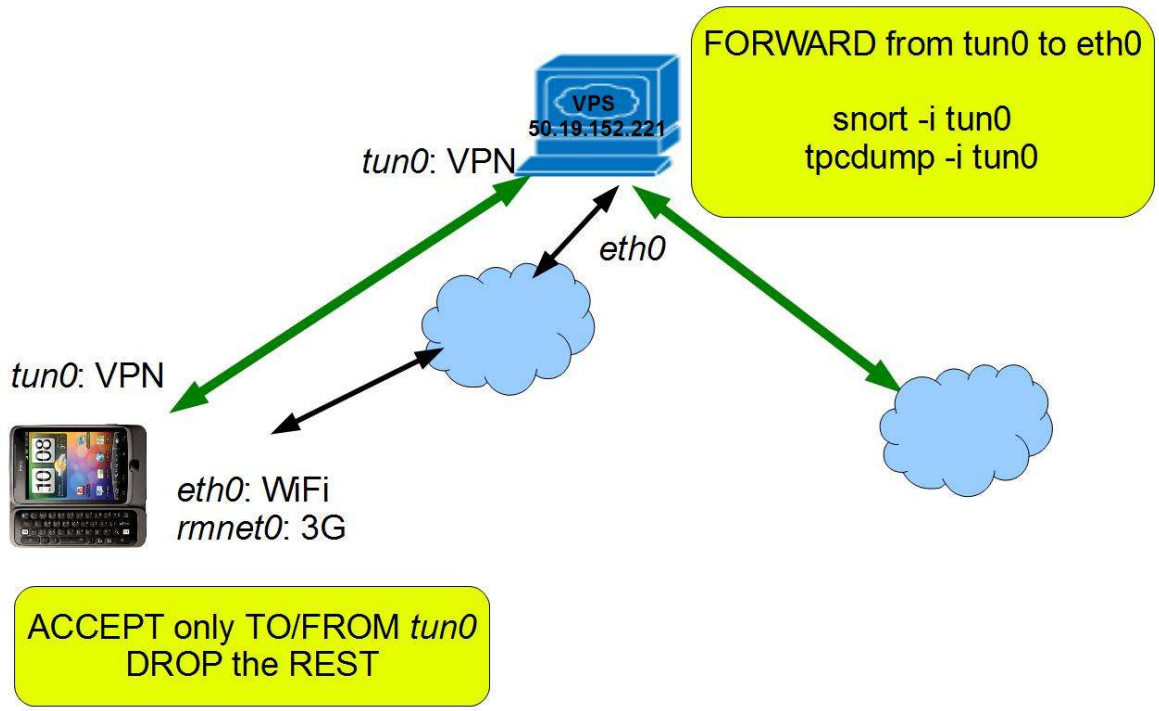
TODO: http://josemilagre.com.br/blog/wp-content/uploads/2014/03/Forensic-Analysis-of-WhatsApp-on-Android-Smartphones.pdf

Android Network Forensics

version : 12-10-2014 16:56

On peut utiliser

- TCPDump sur l'équipement (ou un paquet de type "Network Diagnostics Utility Pro", "pirni")
- un scanner Wifi sur un système distant style Ubuntu
- une Femtocells
- un VPN
  - http://www.sans.org/reading_room/whitepapers/detection/monitoring-network-traffic-android-devices_34097



Android Social Forensics

version : 12-10-2014 16:56

On retrouve toutes les connexions aux réseaux sociaux dans ces équipements

Le problème, c'est de les trouver...

Exemple des mots de passe Wifi :

- [Desire Z] /data/misc/wifi/wpa_supplicant.conf
- [Galaxy S] /data/wifi/bcm_supp.conf
- [Streakroid ROM] /data/misc/wifi/wpa.conf
- /data/data/databases

6.3) iOS

version : 14-01-2015 22:11

iOS Présentation

version : 12-10-2014 17:00

Description

iOS est un système BSD au départ. Il utilise un noyau Mack, un système de fichier Unix et un framework de développement Cocoa. Il y a très peu de changement par rapport à un système Mac OS X, les changements les plus importants sont au niveau du desktop.

Il y a 2 partitions dans le système :

- la partition système qui contient le « kernelcache » (le noyau et ses extensions) ainsi que l'ensemble des librairies, des daemons et des programmes du cœur de l'OS. Cette partition est en lecture seule.
- la partition utilisateur qui contient l'ensemble des applications pour l'utilisateur ainsi que leurs préférences et données associées.

Le boot du système est une version ultra sécurisée du bootloader Mac OS X.

Chaque maillon est signé numériquement et chaque maillon doit vérifier la signature du maillon qui le suit pour éviter toute tentative de modification.

Les applications peuvent stocker leurs mots de passe ou des données confidentielles dans la « keychain ». C'est une base de données chiffrée avec une clé unique au téléphone. L'accès à cette base est contrôlée par le « Security server ».

Architecture

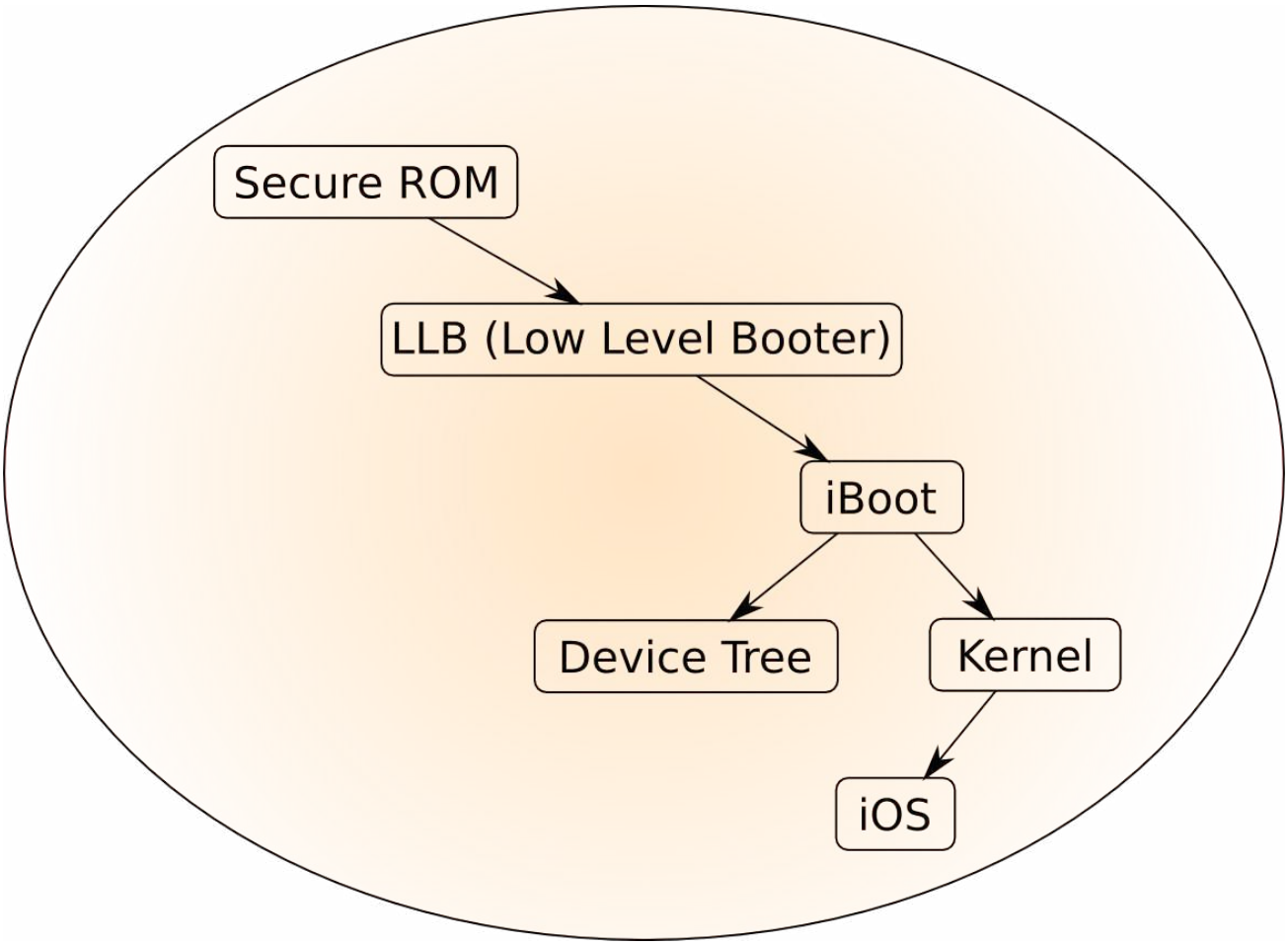
- Chiffrement des fichiers
- Tout exe doit être signé
- Data Execution Prevention
- Bac à sable pour les exe
- Bootloader sécurisé
- Code PIN

Couches logiciels

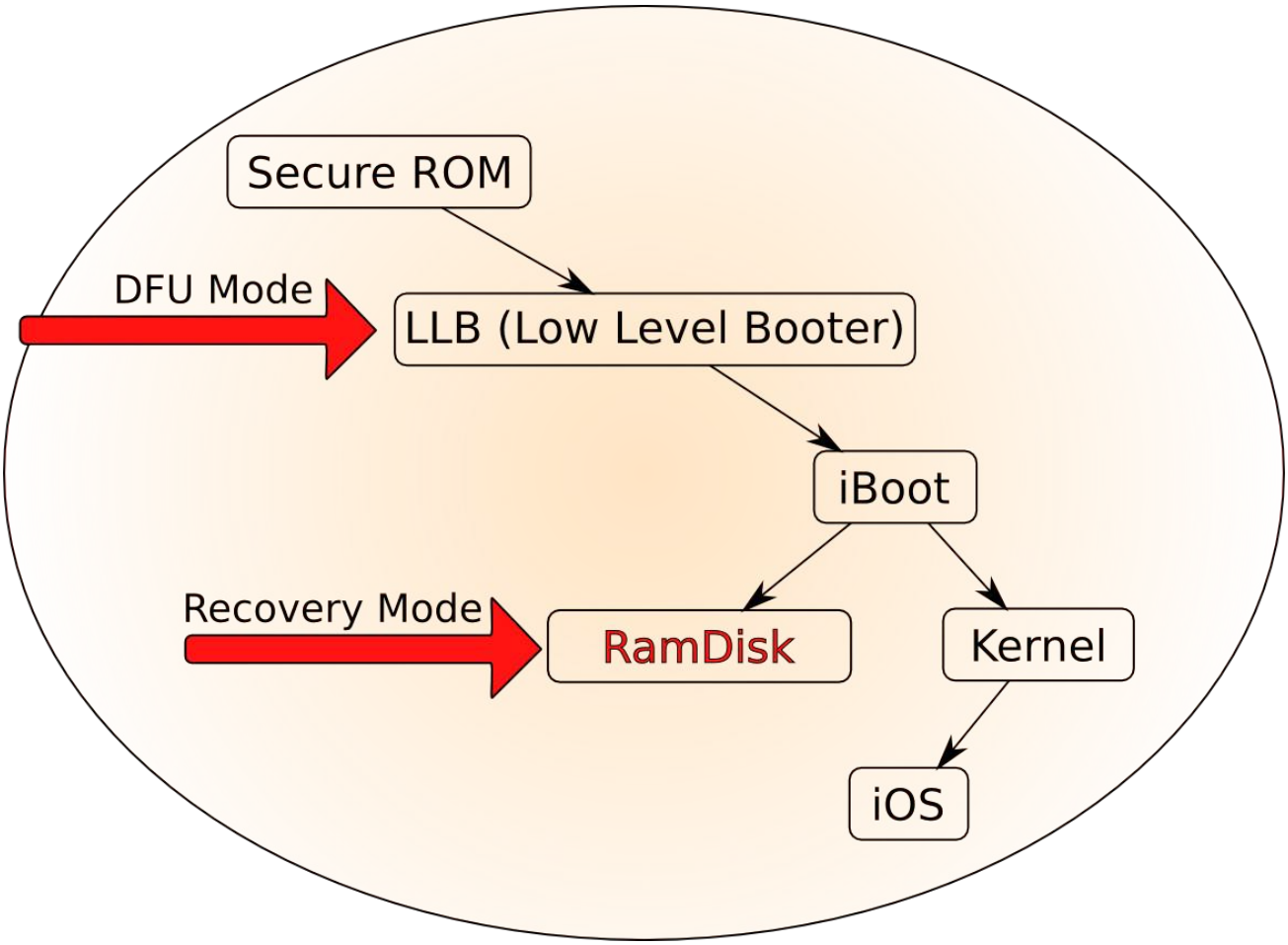
1. Applications
2. Cocoa Touch
3. Media
4. Core Services (Security services)
5. Core OS

Boot

Boot en mode nominal



Boot en mode recovery ou DFU (Device Firmware Upgrade)



iOS Dead Forensics

version : 14-01-2015 22:46

via http://www.sans.org/reading-room/whitepapers/forensics/forensic-analysis-ios-devices-34092 et http://www.policeone.com/police-products/police-technology/articles/7598849-What-cops-need-to-know-about-Apple-s-iOS-8-lockout/

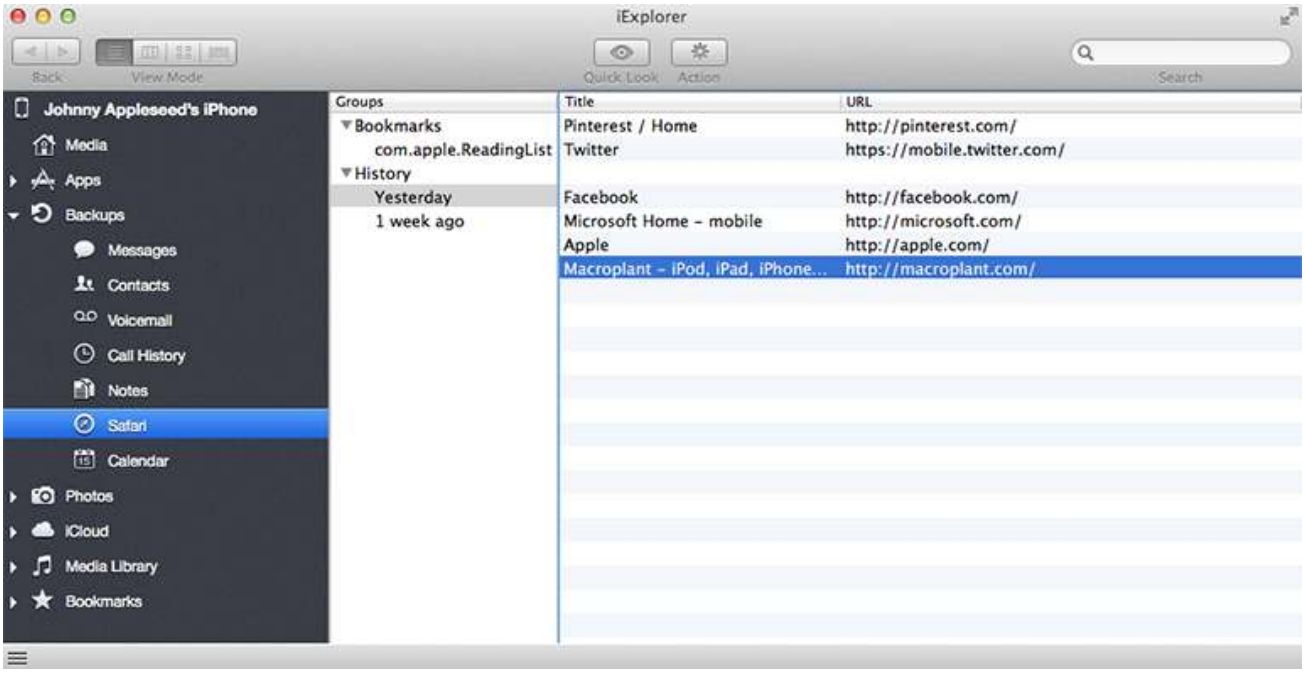
Il y a 3 méthodes de récupération :

- Quand on a accès au smartphone :
  - logique (synchronisation via une solution logiciel)
    - soit en effectuant un jailbreak
    - soit en utilisant les services iOS utilisés par iTunes
  - physique, càd une copie bit-à-bit
    - via le démarrage d'un système alternatif [http://www.limera1n.fr/]
    - https://code.google.com/p/iphone-dataprotection/
- Quand on n'a pas accès au smartphone :
  - via le backup iTunes

À partir d'iOS 8, le téléphone est chiffré et la clé de déchiffrement se récupère à partir du code PIN de l'utilisateur. Apple n'est plus détenteur des clés de déchiffrement.

Il existe une connexion de "confiance" entre le téléphone et le dernier ordinateur qui a été utilisé pour faire une backup. On peut donc récupérer une connexion avec l'iPhone si on récupère cet ordinateur. Mais attention, cette relation de confiance se casse dès que l'iPhone a été arrêté.

Logique



On peut utiliser des logiciels comme "iPhone Explorer" [http://www.macroplant.com/iexplorer/] pour récupérer des données comme :

- historique d'appel
- SMS
- photos
- contacts
- bookmarks
- ...

Backup

Emplacements :

- Windows XP:
  - `C:\Documents and Settings\(username)\Application Data\Apple Computer\MobileSync\Backup\`
- Windows Vista:
  - `C:\Users\(username)\AppData\Roaming\Apple Computer\MobileSync\Backup\`
- Mac OS X:
  - `/Users/(username)/Library/Application Support/MobileSync/Backup/`

Fichiers et répertoires :

- `Status.plist` – état de la dernière synchronisation
- `Manifest.plist` – liste de tous les fichiers sauvegardés (ainsi que la date de modification et le hash)
- `Info.plist` – information à propos de l'iPhone
- `*.mdbackup` – le nom du fichier est le hash SHA1 au moment de la sauvegarde, les données sont sérialisées

Logiciels analysant le backup :

- Device Seizure – Paraben
- Mac Lock Pick – SubRosaSoft
- MDBackupExtract – BlackBag Tech
- iPhone Analyzer

Physique

Très difficile à effectuer, on peut néanmoins utiliser les logiciels suivants :

- Lantern 3 [http://katanaforensics.com/]
- iXam [http://www.ixam-forensics.com]

Listing + ou - exhaustive des logiciels pour analyse les iPhone :

- UFED – Cellebrite
- iXam – FTS
- Oxygen Forensics for iPhone

- .XRY – MicroSystemation
- Lantern – Kantana Forensics
- MacLockPick – SubRosaSoft
- Mobilyze – BlackBag Tech
- Physical DD – Jonathan Zdziarski
- Device Seizure – Paraben
- MobileSyncBrowser – Vaughn S. Cordero
- CellDEK – Logicube
- EnCase Neutrino – Guidance Software
- iPhone Analyzer - http://www.crypticbit.com/zen/products/iphoneanalyzer#!
- MPE+

SQL Carving

Attention à l'ordre des champs SQL lors d'une recherche, pour 2 iPhone identiques (modèle et version), les champs SQL ne sont pas forcément dans le même ordre :

- http://linuxsleuthing.blogspot.fr/2013/01/rotten-apples-watch-out-for-worms.html

Données supprimées dans les bases SQL

Il y a deux types de messages supprimés dans les bases de données SQL :

- les dossiers marqués comme supprimés dans la base de données (donc pas vraiment supprimé à tous) et
- les enregistrements supprimés de la base de données elle-même

Plus d'info :

- http://linuxsleuthing.blogspot.fr/2011/02/parsing-iphone-sms-database.html
- http://linuxsleuthing.blogspot.fr/2011/02/recovering-data-from-deleted-sql.html

iOS Live Forensics

version : 14-01-2015 21:05

Le plus simple est de passer par les services iOS utilisés par iTunes

iOS Network Forensics

version : 14-01-2015 21:24

On peut utiliser ~TCPDump sur l'équipement ou un paquet de type

- "Network Diagnostics Utility Pro"
- "pirni"

On peut utiliser un scanner Wifi sur un système distant style Ubuntu

On peut utiliser une Femtocells

On peut utiliser le service réseau com.apple.pcapd accessible via iTunes qui permet de créer un TAP virtuel, cela permet d'écouter les trames réseau échangées par des programmes malicieux.

iOS Social Forensics

version : 12-10-2014 17:08

On retrouve toutes les connexions aux réseaux sociaux dans ces équipements Le problème, c'est de les trouver...

6.4) Windows Phone

version : 14-01-2015 22:47

Produit encore très "jeune"

Très peu de recherche encore dessus Mais on peut effectuer :

- du Dead Forensics
  - Windows Phone Device Manager

- du Live Forensics
  - pas de copie mais utilisation d'une apps
- du Network Forensics
  - TCPDump
- du Social Forensics
  - il suffit de trouver les mots de passe

6.5) Blackberry

version : 14-01-2015 22:54

Produit très cloisonné

Très peu de recherche encore dessus

Mais on peut effectuer :

- du Dead Forensics
  - le SDK permet la copie bit-à-bit
- pas de Live Forensics
- du Network Forensics
- du Social Forensics
  - il suffit de trouver les mots de passe

Logiciels :

- Chip-Off – BlackBerry® devices

À lire : http://www.nist.gov/forensics/upload/5-Punja-nist-2014-bb-forensics-FULL.pdf

6.6) Et les autres...

version : 12-10-2014 17:09

Pour certains, il y a peu d'informations à récupérer.

- les contacts,
- les SMS,
- le journal d'appel,
- les photos (quand il y en a),
- le calendrier et les mémos

Logiciels :

- Bitpim

Bitpim

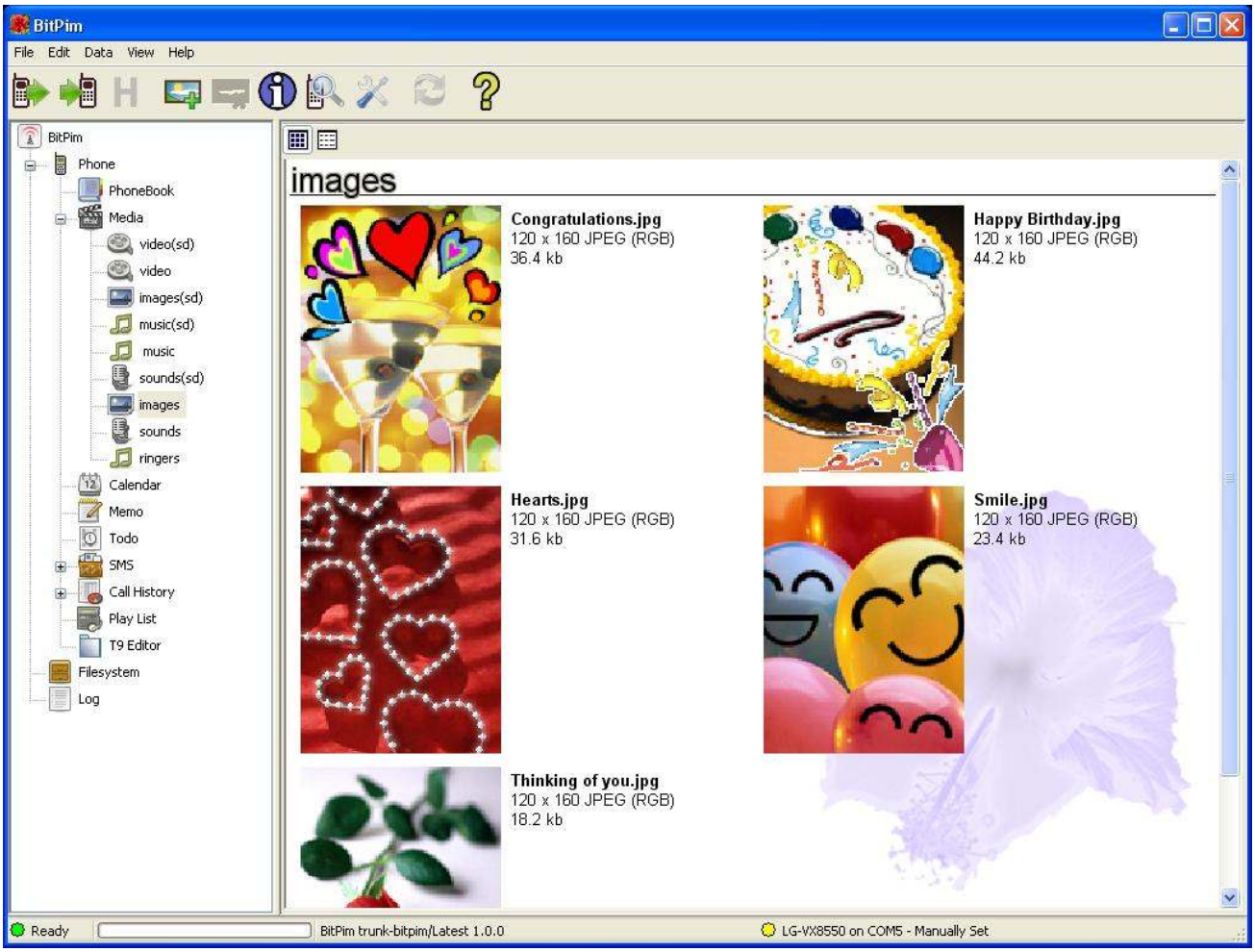
version : 12-10-2014 17:10

http://www.bitpim.org/

Manage data on CDMA phones from LG, Samsung, Sanyo and others

Téléphones supportés officiellement :

- LG ~AX-8600
- LG C2000
- LG G4015
- LG ~LX570
- LG ~PM225
- LG ~UX-5000
- LG ~VX-*
- Motorola Phones
- Samsung Phones
- Sanyo ~SCP-6600 (Katana)
- Other Sanyo Phones
- Toshiba ~VM-4050



7) Conclusions

version : 16-12-2015 09:34

TableOfContents

Pour aller plus loin

De nombreux exemples sont disponibles ici :

- http://www.forensicfocus.com/images-and-challenges
- Spy Hunter Holiday Challenge 2014 http://blog.mywarwithentropy.com/2014/11/spy-hunter-holiday-challenge-2014.html

Pense-bête : https://blogs.sans.org/computer-forensics/files/2012/06/SANS-Digital-Forensics-and-Incident-Response-Poster-2012.pdf

Analyse de malware : http://postmodernsecurity.com/2015/09/11/malware-analysis-and-incident-response-tools-for-the-frugal-and-lazy/

http://digital-forensics.sans.org/community/summits