iOS Forensics with Open-Source Tools

Andrey Belenko

VIAFORENSICS

black hat REGIONAL SUMMIT

AGENDA

- Basics
- iOS Security
- iOS Data Protection
- Hands-On!

FORENSICS 101

Acquisition ➜ Analysis ➜ Reporting

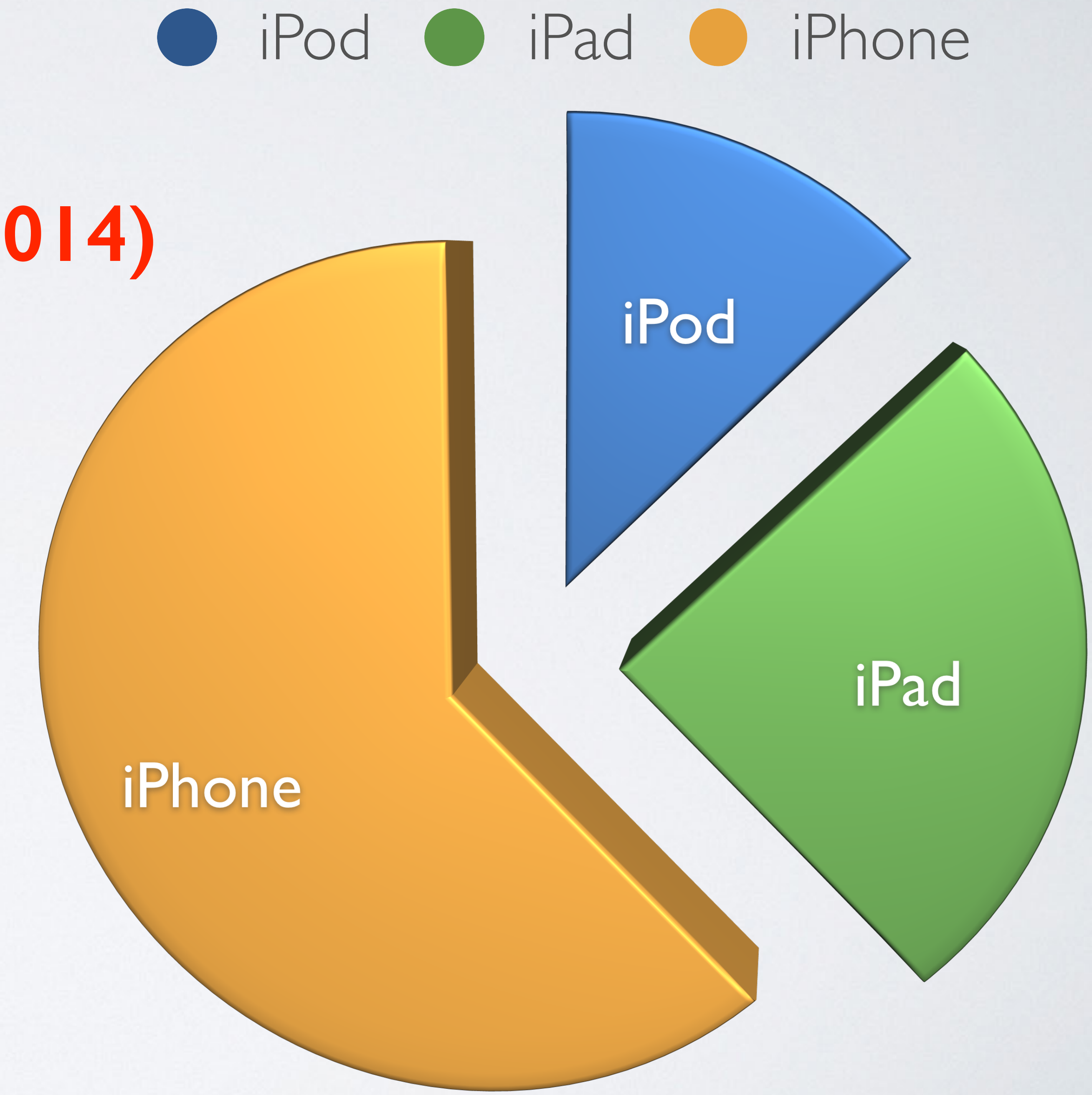
GOALS:

1. Assuming physical access to the device extract as much information as practical

2. Leave as little traces/artifacts as practical

WHY BOTHER?

More than 800M devices (Jun 2014)



IOS FORENSICS 101

- Passcode
  - Protects device from unauthorised access
  - Cryptographically protects some data
- Keychain
  - System-wide storage for passwords and other sensitive data
  - Encrypted
- Disk/Files
  - Encrypted

IOS FORENSICS 101

- Logical
  - Uses external logical interfaces
  - iTunes Backup
  - “Backdoor” services: file_relay and house_arrest
- Physical
  - Extract disk image
  - Bruteforce passcode
  - Needs code execution on the device

IOS FORENSICS 101

- iCloud Backup
  - Downloads backup from the iCloud
  - No encryption
  - Needs Apple ID and password
- NAND
  - “Extension” of physical
  - Potentially allows recovery of deleted files

IOS SECURITY

Chain of trust:

- BootROM (programmed at the factory; read-only)
- iBoot (signature checked and loaded by BootROM)
- Kernel (signature checked and loaded by iBoot)
- Applications (verified and run by kernel)

Applications must be signed

- $99/yr for Developer certificate or $399/yr for an Enterprise one

Applications are sandboxed

JAILBREAK

- Circumvents iOS security to run custom (=unsigned) apps
- Does this by breaking chain of trust
- Can break it at any level from BootROM to kernel
- Can be tethered or untethered

JAILBREAK

Boot-level JB

- Exploits BootROM or iBoot
- Loads custom (patched) kernel
- BootROM exploits cannot be patched!

User-level JB

- Exploits running kernel
- Usually subject to more limitations
  - No passcode, no backup password, etc

JAILBREAK

Tethered JB

- Connection to host is required to JB
- Host sends exploits
- JB doesn’t persist across reboots
- May leave very few traces (esp. boot-level tethered JB)

Untethered JB

- Device is modified to JB itself on each boot
- JB persists across reboots
- Leaves permanent traces

IOS SECURITY

iPhone 4 + iOS 4

- Proper passcode protection
- Proper data encryption
- Common name: iOS Data Protection
- Challenge for iOS forensics

iPhone 4S, 5, 5c have minor changes

iOS 5-8 introduce incremental changes to Data Protection

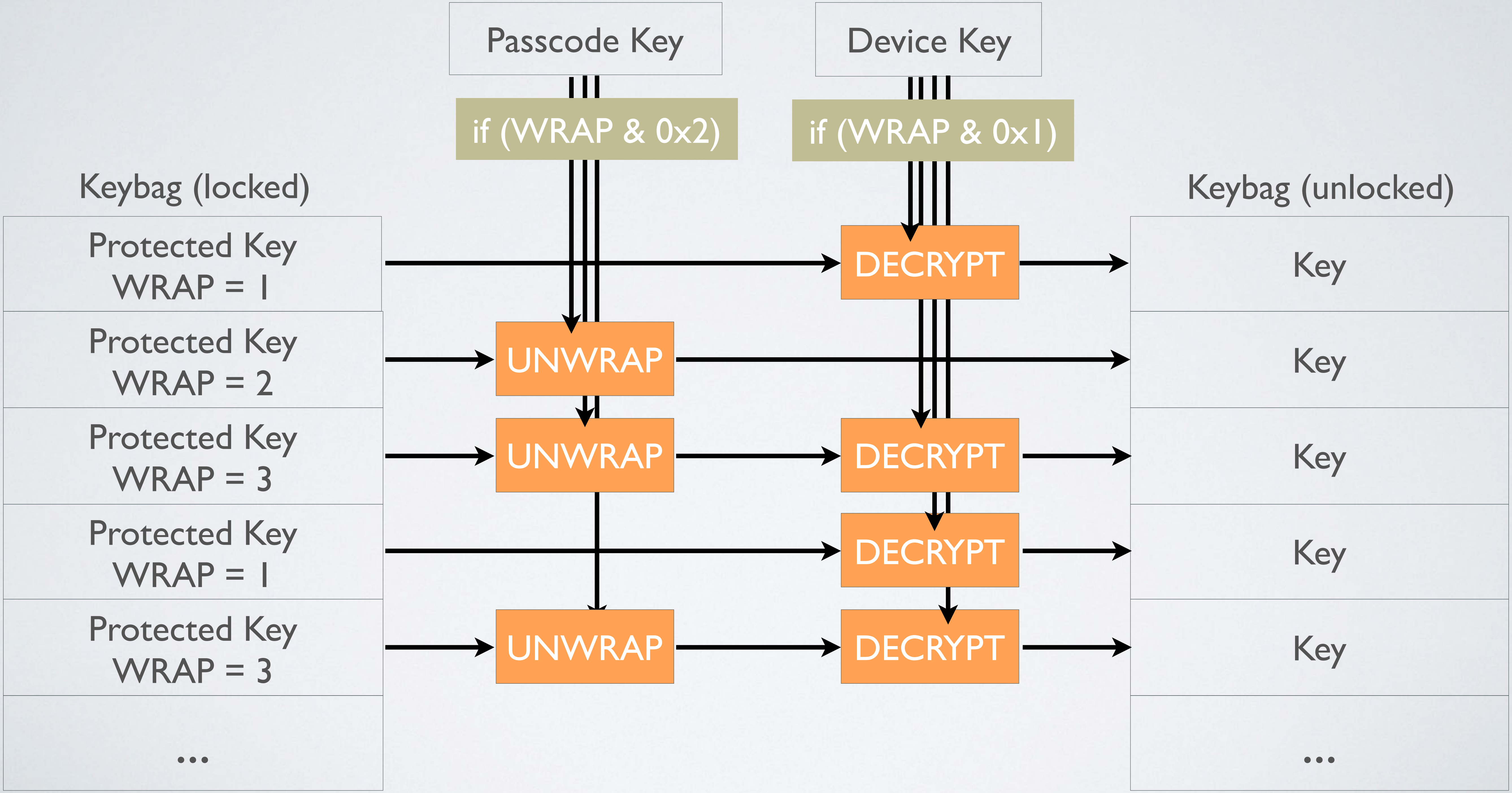
DATA PROTECTION

- More robust passcode protection
  - Passcode participates in data encryption
  - Offline bruteforce not possible
- Better disk encryption
  - Per-file encryption key
- Better keychain encryption
  - Per-item encryption key
- New iTunes backup format
  - Slower password recovery

PROTECTION CLASSES

- Content grouped by accessibility requirements
  - Available at all times
  - Available only when device is unlocked
  - Available after device has been unlocked at least once after boot
- Random master key (class key) for each protection class
- Each class key encrypted with device key and optionally passcode key
- Class keys for all protection classes are stored in System Keybag
  - /var/keybags/systembag.kb
  - New keybag is generated on device restore/wipe

KEYBAG PROTECTION



Passcode Key

Device Key

if (WRAP & 0x2)

if (WRAP & 0x1)

Keybag (locked)

Protected Key
WRAP = 1

Protected Key
WRAP = 2

Protected Key
WRAP = 3

Protected Key
WRAP = 1

Protected Key
WRAP = 3

...

Keybag (unlocked)

Key

Key

Key

Key

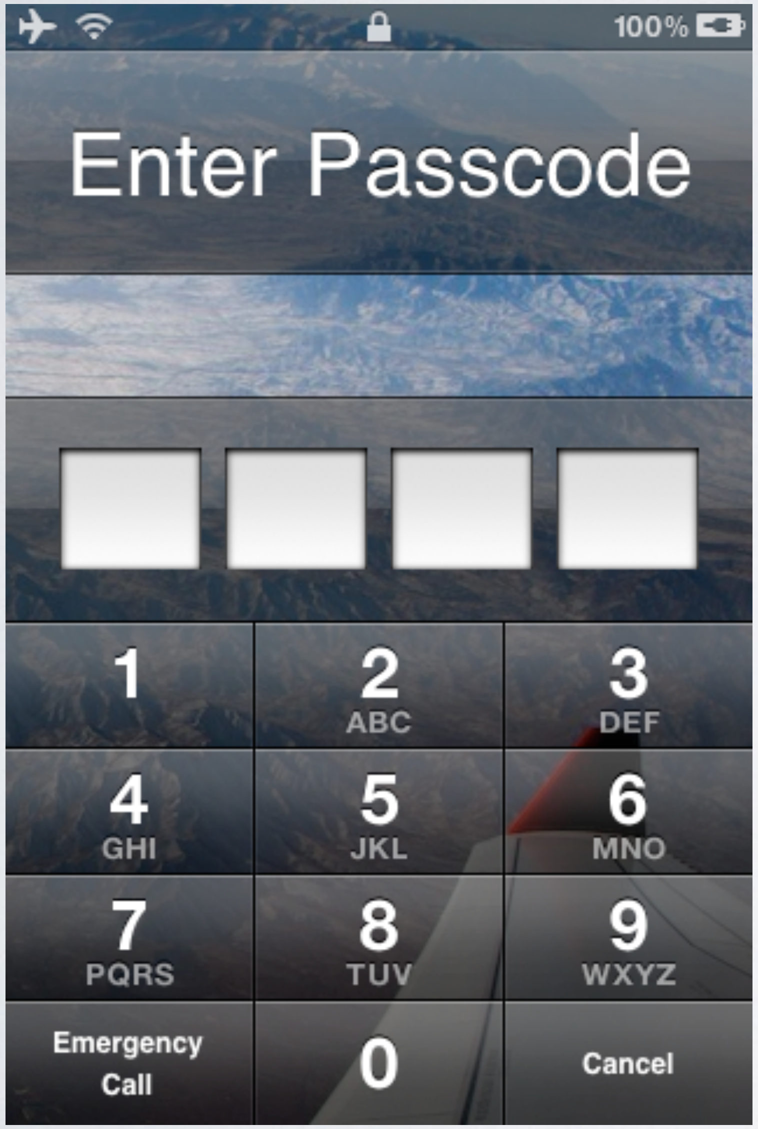
Key

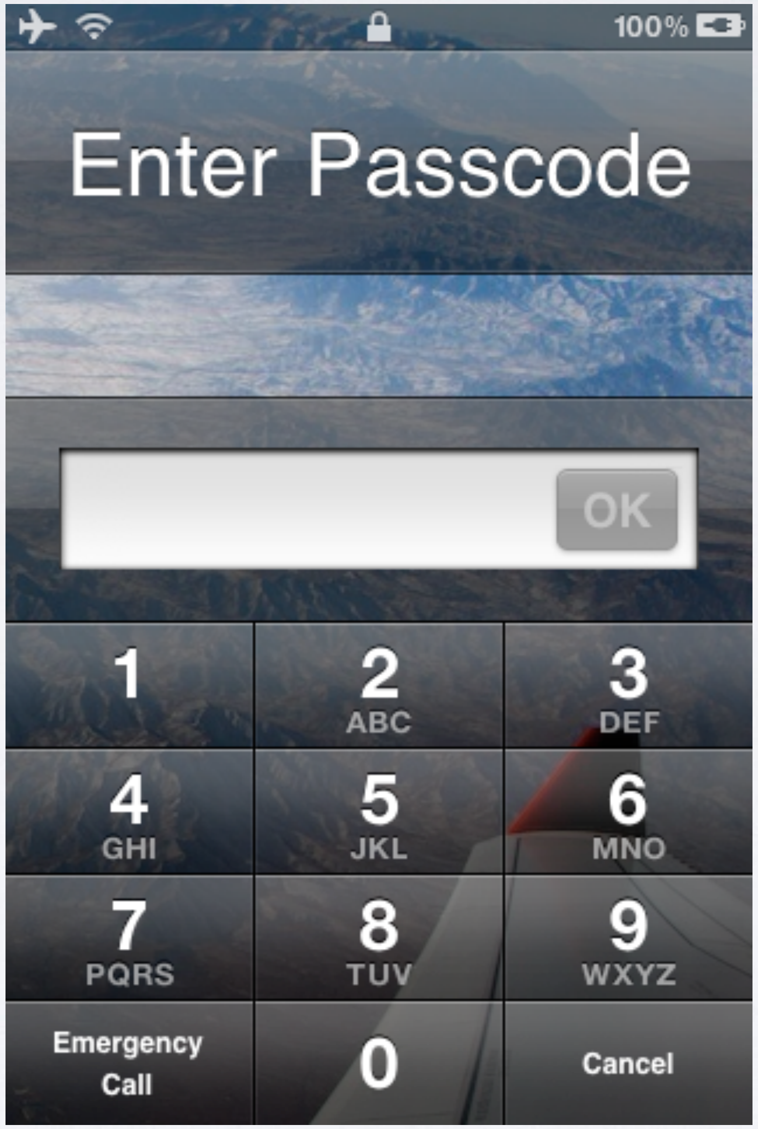
...

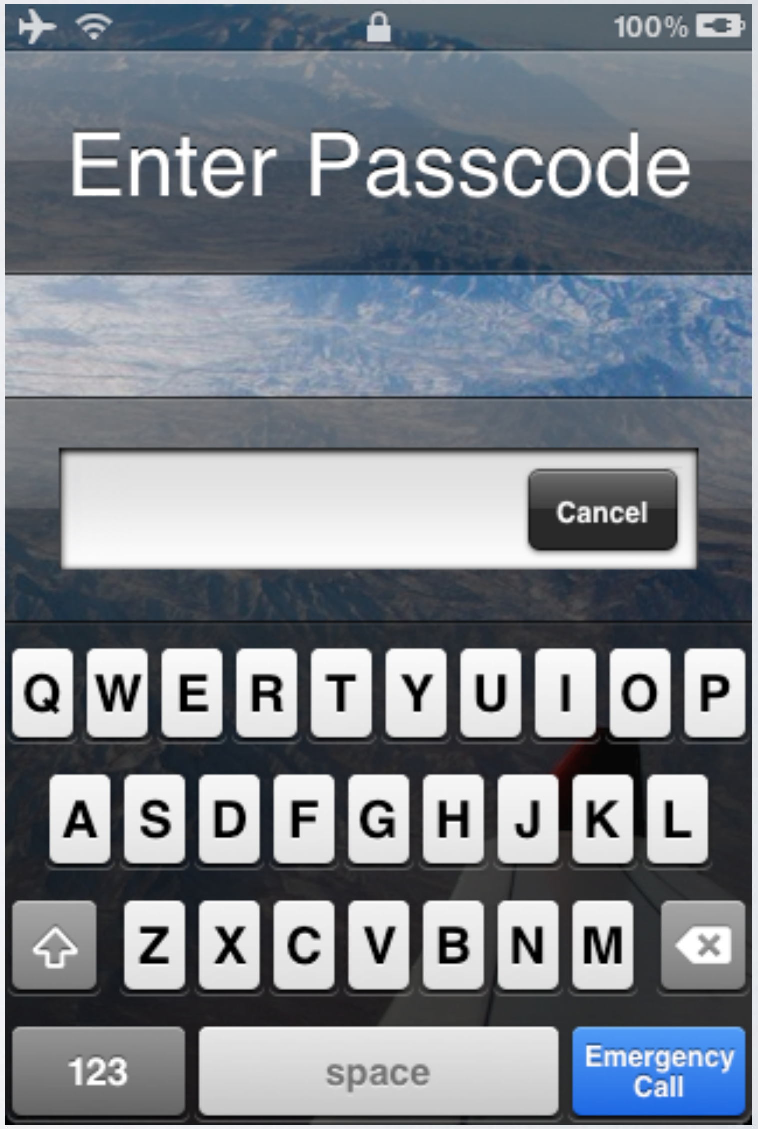
PASSCODE

- Passcode key protects most class keys
- Passcode key is computed from passcode
  - Computation depends on device-specific UID (UID+ on newer hardware) key
  - Must be done on device; cannot bruteforce offline
- System keybag contains hint on passcode complexity

PASSCODE


100%
Enter Passcode
1
2 ABC
3 DEF
4 GHI
5 JKL
6 MNO
7 PQRS
8 TUV
9 WXYZ
Emergency Call
0
Cancel


100%
Enter Passcode
OK
1
2 ABC
3 DEF
4 GHI
5 JKL
6 MNO
7 PQRS
8 TUV
9 WXYZ
Emergency Call
0
Cancel


100%
Enter Passcode
Cancel
Q W E R T Y U I O P
A S D F G H J K L
Z X C V B N M
123
space
Emergency Call

KEYCHAIN

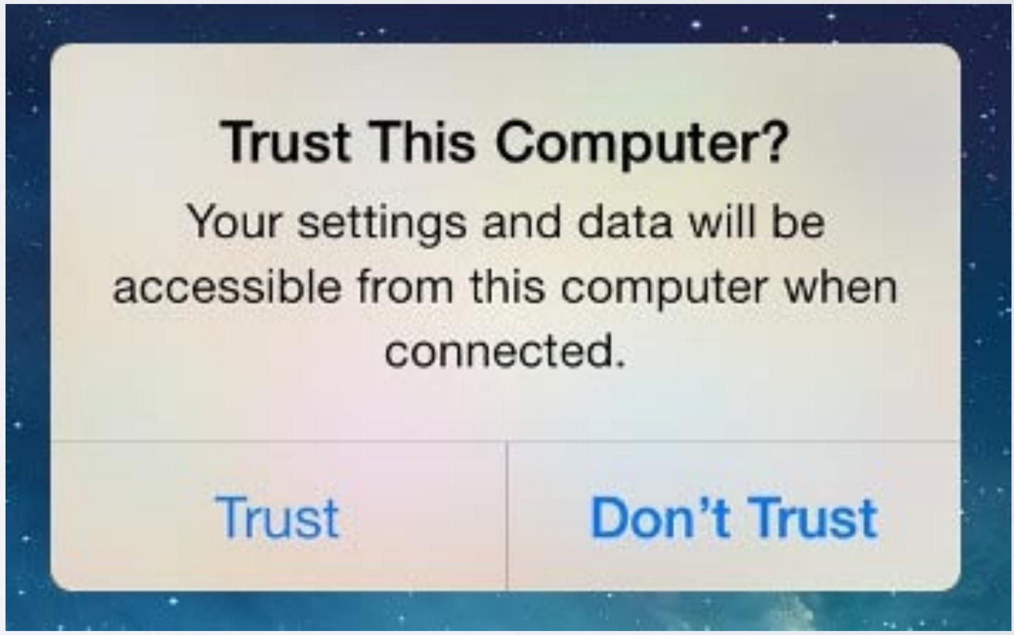
- SQLite3 DB
- iOS 4: only passwords are encrypted (metadata in clear)
- iOS 5+: passwords and metadata are encrypted
- iOS 4: AES-CBC
- iOS 5+: AES-GCM
- Random key for each item/password
- Item key is encrypted with corresponding class key

DISK ENCRYPTION

- Only Data (User) partition is encrypted
- Not a full-disk encryption but per-file encryption, more like EFS
- File key, encrypted with class key, is stored in com.apple.system.cprotect extended attribute
- Protection classes:
  - NSFileProtectionNone
  - NSFileProtectionComplete
  - NSFileProtectionCompleteAfterFirstAuthentication (iOS 5+)
  - NSFileProtectionCompleteUnlessOpen (iOS 5+)

PAIRING

- Key negotiation/generation
- Device must be unlocked
- Since iOS 7 user must confirm pairing
- Pairing record gives same powers as knowing the passcode



Trust This Computer?

Your settings and data will be accessible from this computer when connected.

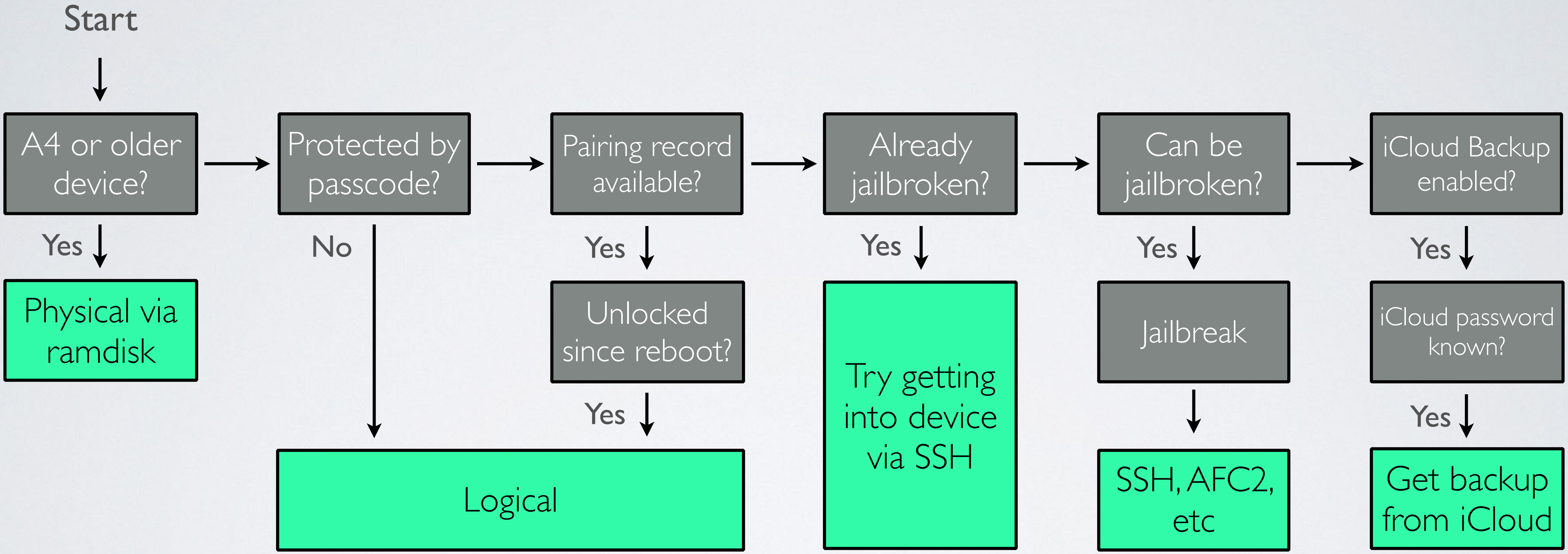
IOS SECURITY

iPhone 5s

- 64-bit
- Secure Enclave (SEP)
- Touch ID
  - More passcode-protected devices
- Yet another challenge for (physical) iOS forensics

iPhone 6, 6 Plus have minor changes

WORKFLOW



Start

A4 or older device?

Yes

Physical via ramdisk

Protected by passcode?

No

Pairing record available?

Yes

Unlocked since reboot?

Yes

Logical

Already jailbroken?

Yes

Try getting into device via SSH

Can be jailbroken?

Yes

Jailbreak

SSH, AFC2, etc

iCloud Backup enabled?

Yes

iCloud password known?

Yes

Get backup from iCloud

QUESTIONS SO FAR?

HANDS-ON

Let’s Get Hacking!

TOOLS OF THE TRADE

- Physical
  - iphone-dataprotection from Sogeti
- Logical
  - libimobiledevice
- Environment
  - Santoku Linux 0.5 (VM guest)
  - OS X (VM host) with VMware Fusion
  - Windows and/or VirtualBox may also work

IPHONE-DATAPROTECTION

- https://code.google.com/p/iphone-dataprotection/
- OS X to build ramdisk and modified kernel
- OS X or Windows to boot device
- Doesn't reliably work from within VM because of USB

SANTOKU

- We'll be using Santoku Linux 0.5 as our base
    - Based off Lubuntu 14.04
- Not a strict requirement at all – can use any Linux distribution

- User/pwd for workshop VM:

santoku/santoku

LOGICAL

libimobiledevice

http://www.libimobiledevice.org

https://github.com/libimobiledevice/

LIBIMOBILEDEVICE – BUILDING

- https://github.com/libimobiledevice/libplist/archive/1.12.tar.gz
  - ./autogen.sh && make && sudo make install
- https://github.com/libimobiledevice/libusbmuxd/archive/1.0.10.tar.gz
  - ./autogen.sh && make && sudo make install
- https://github.com/libimobiledevice/libimobiledevice/archive/1.1.7.tar.gz
  - ./autogen.sh --enable-dev-tools
  - make && sudo make install
- https://github.com/libimobiledevice/usbmuxd/archive/1.1.0.tar.gz
  - ./autogen.sh --without-systemd (at least on Santoku 0.5)
  - make && sudo make install

LIBIMOBILEDEVICE – BUILDING ADDITIONAL TOOLS

- https://github.com/libimobiledevice/ideviceinstaller/archive/1.1.0.tar.gz
    - ./autogen.sh
    - make
    - sudo make install
- https://github.com/libimobiledevice/ifuse/archive/1.1.3.tar.gz
    - ./autogen.sh
    - make
    - sudo make install

LIBIMOBILEDEVICE

List connected devices

```
idevice_id -l
```

LIBIMOBILEDEVICE

Get device info

```
ideviceinfo -s
ideviceinfo [-q <domain>] [-x > out.plist]
```

LIBIMOBILEDEVICE

List installed applications

```
ideviceinstaller -l
ideviceinstaller -l [-o ]
```

LIBIMOBILEDEVICE

Create full device backup

```
idevicebackup2 backup  --full <location>
```

LIBIMOBILEDEVICE – HIDDEN GEM

com.apple.mobile_file_relay client

filerelaytest

FILE RELAY – SOURCES

AppleTV
Baseband
Bluetooth
Caches
CoreLocation
CrashReporter
CLTM
demod
Keyboard
Lockdown
MobileBackup
MobileInstallation
MobileMusicPlayer
Network

Photos
SafeHarbor
SystemConfiguration
Ubiquity
UserDatabases
AppSupport
Voicemail
VPN
WiFi
WirelessAutomation
MapsLogs
NANDDebugInfo
IORegUSBDevice
VARFS
HFSMeta

tmp
MobileAsset
GameKitLogs
Device-O-Matic
MobileDelete
itunesstored
Accounts
AddressBook
FindMyiPhone
DataAccess
DataMigrator
EmbeddedSocial
MobileCal
MobileNotes

FILE RELAY – CPIO.GZ

```
gunzip <file.cpio.gz>
cpio -imdv <file.cpio>
```

FILE RELAY – IOS 8

- Guarded in iOS 8
- /Library/Managed Preferences/mobile/com.apple.mobile_file_relay.plist
- Set "Enabled" = true

HOUSE ARREST

Access application’s sandbox

```
ifuse --container <bundle.id> <location>
```

Unmount

```
fusermount -u <location>
```

ICLOUD BACKUP

iLoot

https://github.com/hackappcom/iloot

THANKS!

✉ ABelenko@viaforensics.com

@abelenko

VIAFORENSICS

black hat®
REGIONAL SUMMIT