HACKITO ERGO SUM

GOTO: H[a]CK

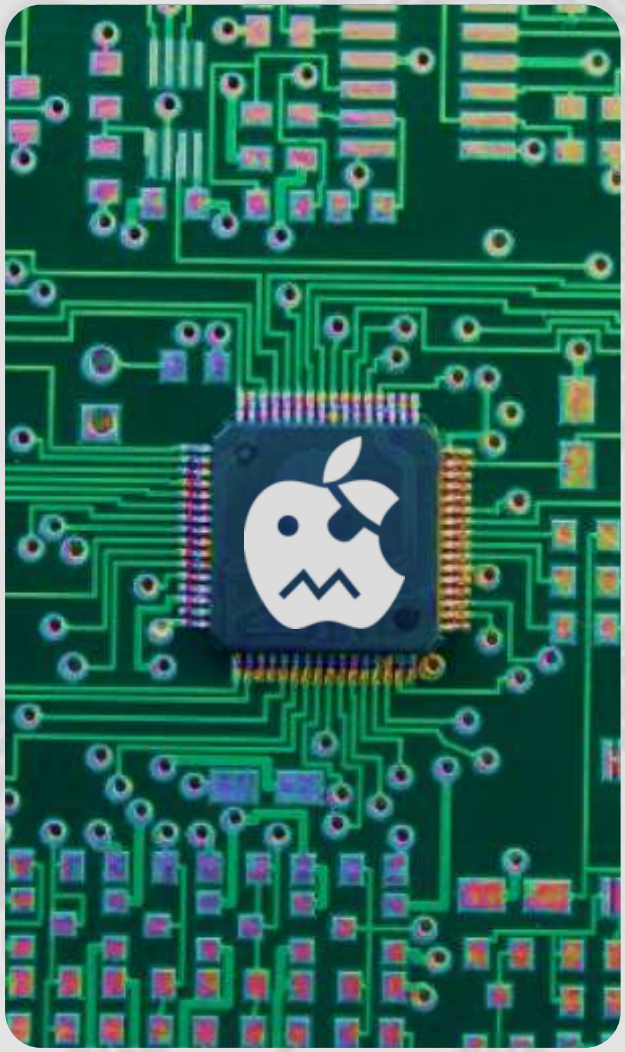
SOGETI

Hacking apple accessories to pown iDevices

Wake up Neo! Your phone got pown!

Mathieu RENARD - @GOTOHACK
mathieu.renard[-at-]gotohack.org

Who am I ?



@GotoHack
- DAY: Pentester & Team Leader
- NIGHT: Security Researcher

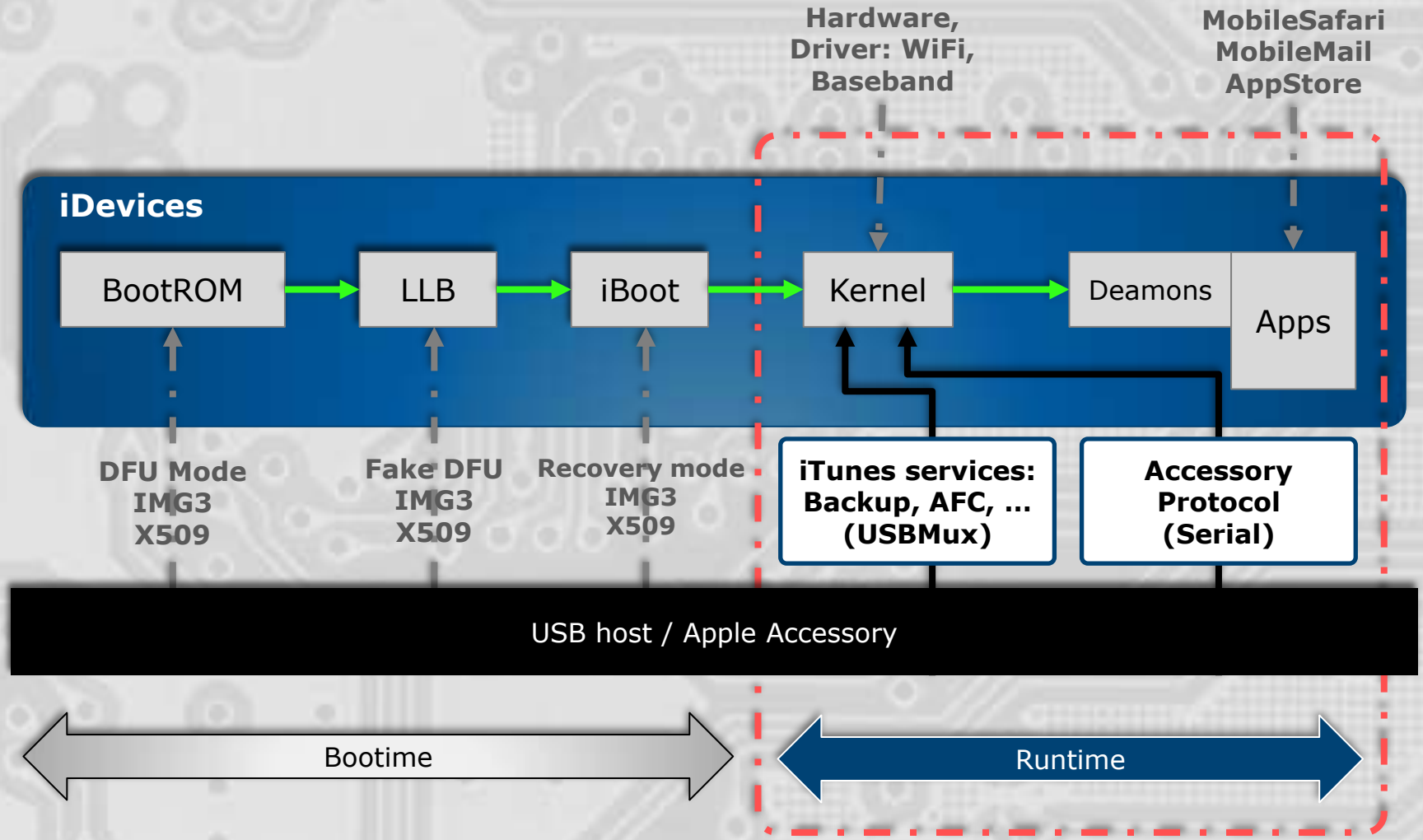
Area of expertise
- Mobility / BYOD
- Web application
- Embedded systems
- Hardware Hacking

Publications
- GreHack 2012
  - Practical iOS Application Hacking
- HACK.lu 2012
  - Hacking iOS Application

MFI Devices Invasion

Made for iPod iPhone iPad

iDevices Attack surface

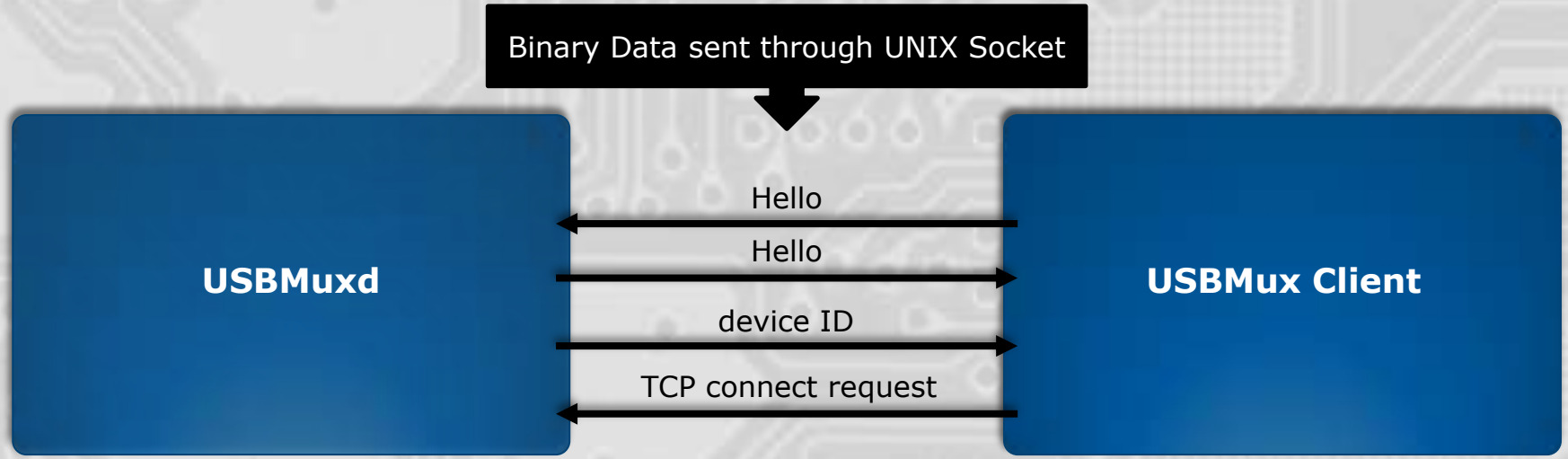


iTunes Services

USBmuxd and USBmux protocol
Client side communication

SOGETI

USBmuxd
- Daemon is started at system launch (on user system).
- Creates a listening UNIX Domain Socket at /var/run/usbmuxd.
- Wait for iDevice connections via USB
- Allows multiplexing of TCP connection over one USB pipe

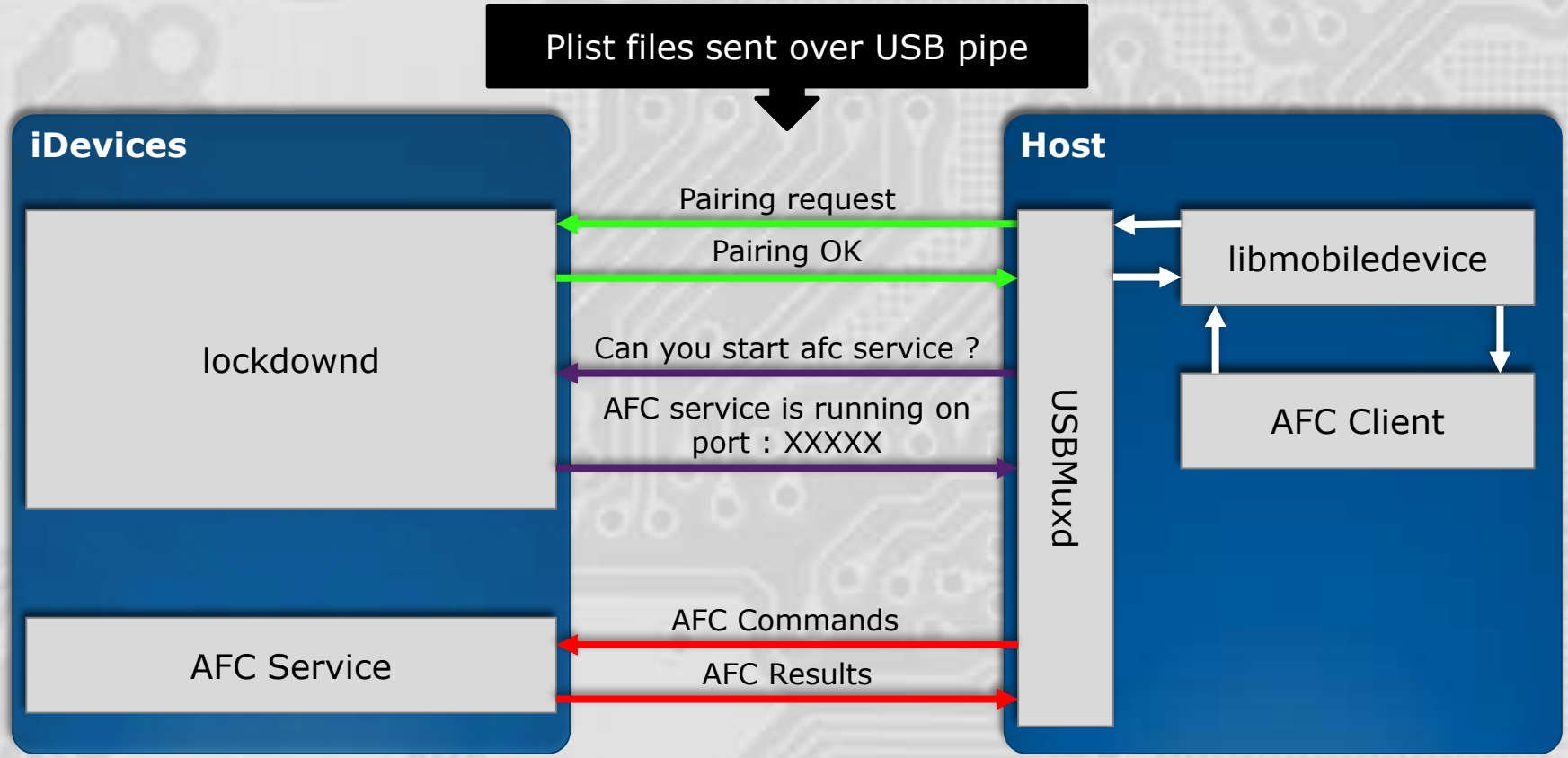


Lockdownd

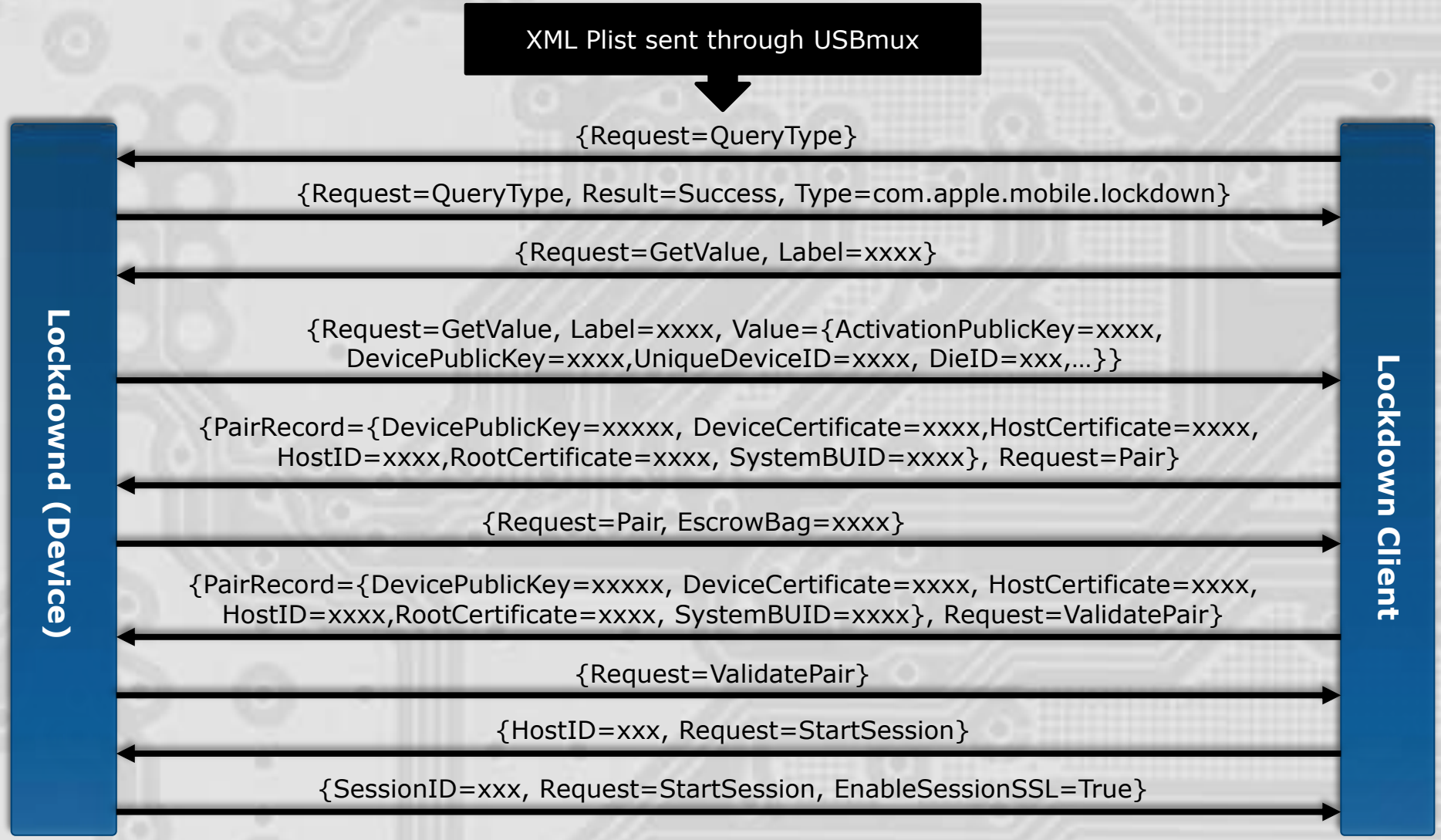
Lockdownd binary

- Responsible for several tasks
  - Pairing,
  - Activation,
  - Unlocking FairPlay certificates,
  - Delegating communications to other services
  - …
- Listening on port 62078
  - Accessed through the usbmux protocol.
  - Packets
    - Data length : 32bits big endian word
    - Data : XML plist
- **Only available after pairing.**
  - **First pairing require the device to be unlocked**

iTunes' service communication overview



Lockdown protocol & Pairing



libImobiledevice / pymobiledevice

Libimobiledevice
  - Cross-platform software library
  - Developed by Nikias Bassen
  - Handles the protocols to support iDevices.
  - Based on the open source implementation of usbmuxd

Pymobiledevice
  - Lite python implementation
  - Handles only most important protocols to support iDevices
  - Based on the open source implementation of usbmuxd

Allows other software to easily interact with the services hosted on the device.

com.apple.mobilebackup & com.apple.mobilebackup2

Mobilebackup services
  - Used by iTunes to backup the device

iDevice backup
  - Permit a user to restore personal data and settings
  - Abusing this service may allow an attacker
    - Retrieving personal and confidential data
      - SMS
      - Call Logs
      - application data
      - default preferences
      - data stored in the keychain (WiFi password, VPN Certificate Passwords).
    - Inject data to the device.
  - Can be password protected

com.apple.afc

- AFC (Apple File Connection)
  - Service running on all iDevices
  - Handled by /usr/libexec/afcd
  - Used by iTunes to exchange files
  - AFC clients can access certain files only
    - Files located in the Media folder

com.apple.mobile.house_arrest

- House_arrest
  - allows accessing to AppStore applications folders and their content.

- Using an AFC client, a user/attacker can download the application resources and data (documents, photos…).
  - Including "default preferences"
    - File where credentials are sometimes stored.

com.apple.mobile.installation_proxy

Installation proxy
- Manages applications on a device
  - List installed applications.
  - Install an application on the device.
  - Upgrade an application on the device.
  - Uninstall an application from the device.
  - List archived applications.
  - Archive an application on the device
    - Creating a ZIP archive in the "ApplicationArchives" directory and uninstalling the application
  - Removes a previously archived application from the device
- Used by the com.apple.mobile.house_arrest
  - Enumerate and dump installed applications.

com.apple.mobile.diagnostics_relay

Diagnostics relay
- Allows requesting iOS diagnostic information.
- Handles the following actions:
  - Puts the device into deep sleep mode and disconnects from host.
  - Restart the device and optionally show a user notification.
  - Shutdown of the device and optionally show a user notification.
- Used by evasi0n to update some caches by rebooting the device.

com.apple.mobile.file_relay

File_Relay

- Allow paired devices to launch the following commands
  - AppleSupport,
  - Network,
  - WiFi,
  - SystemConfiguration,
  - VPN,
  - UserDatabases,
  - CrashReporter,
  - Tmp,
  - Caches
- All the files returned are stored in clear text in a CPIO archive
- Asking for UserDatabases allow retrieving
  - SMS, Contacts, Calendar and Email from databases in clear text.

Summary

- Pairing is initiated on the USB Host side
  - Unlocking the device is mandatory
  - This implementation may allow malicious dock station to
    - Retrieve & Inject
      - SMS
      - Call Logs
      - application data
      - default preferences and data stored in the keychain (using backup)

HACKITO ERGO SUM

GOTO: H[a]CK

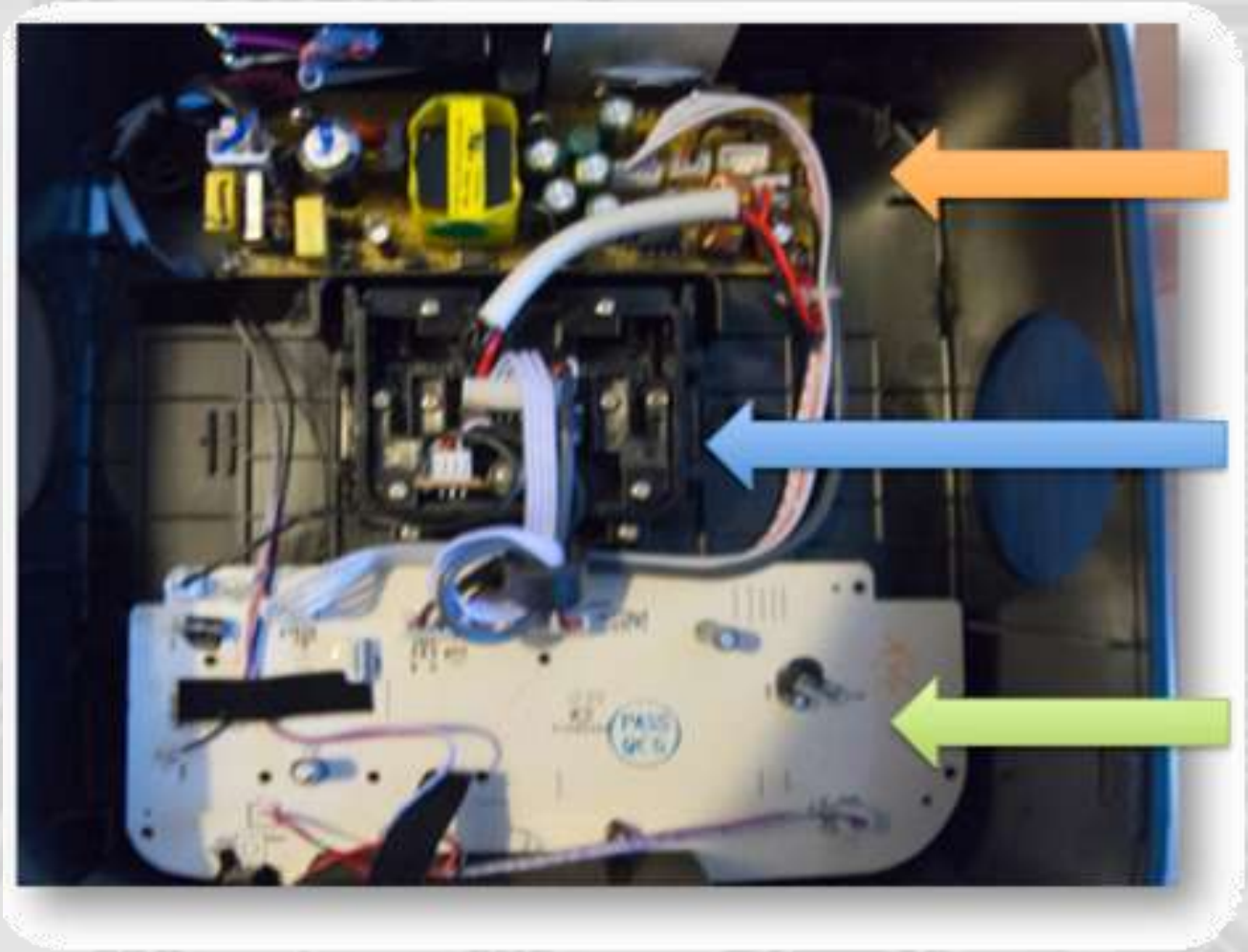
SOGETI

Reversing an Apple MFI accessory

Anatomy of an Accessory

- MFI Alarm clock
  - Apple dock connector
  - Features :
    - Compatible with all iPods
    - Wake up to iPod
    - Full-function remote control
    - Charges iPod whilst connected

Opening the box…

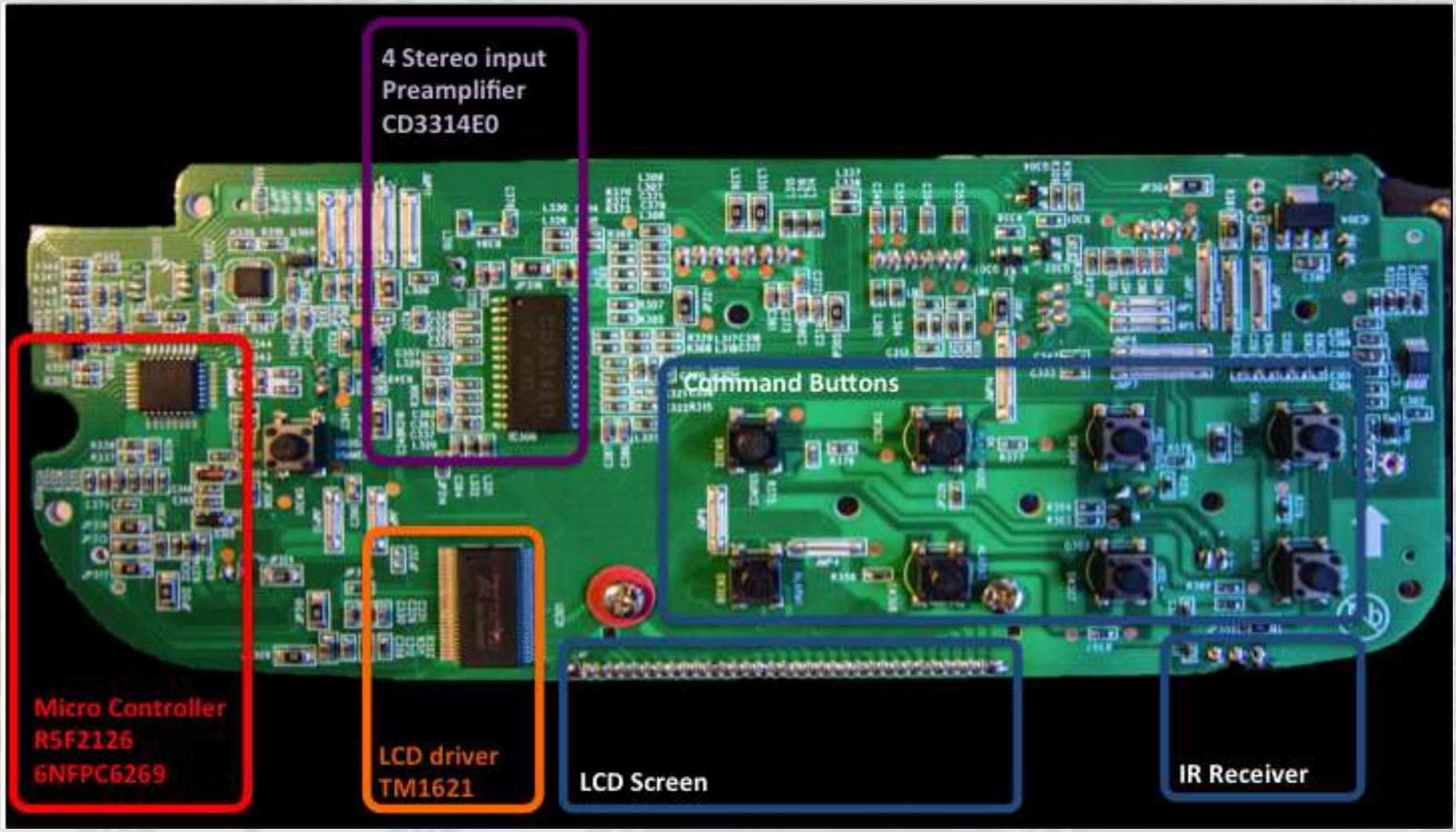


Power supply & Audio Amplifier

iDevice interface

Mother board

Mother board analysis



R5F2126

RENESAS

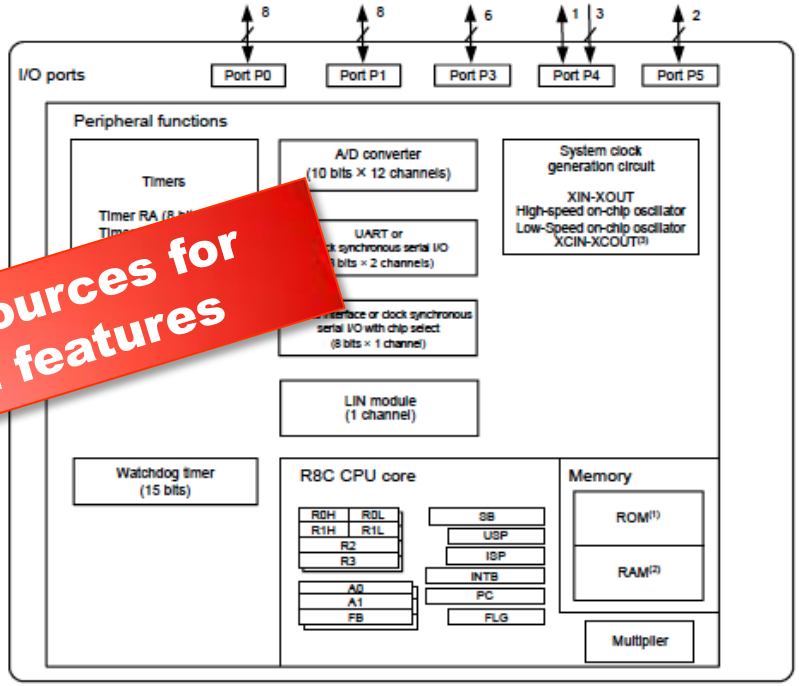
R8C/26 Group, R8C/27 Group

SINGLE-CHIP 16-BIT CMOS MCU

REJ03B0168-0210
Rev.2.10
Sep 26, 2008

In-System Programming
On-chip data flash (1Kbytes)
Internal ROM (32 Kbytes)

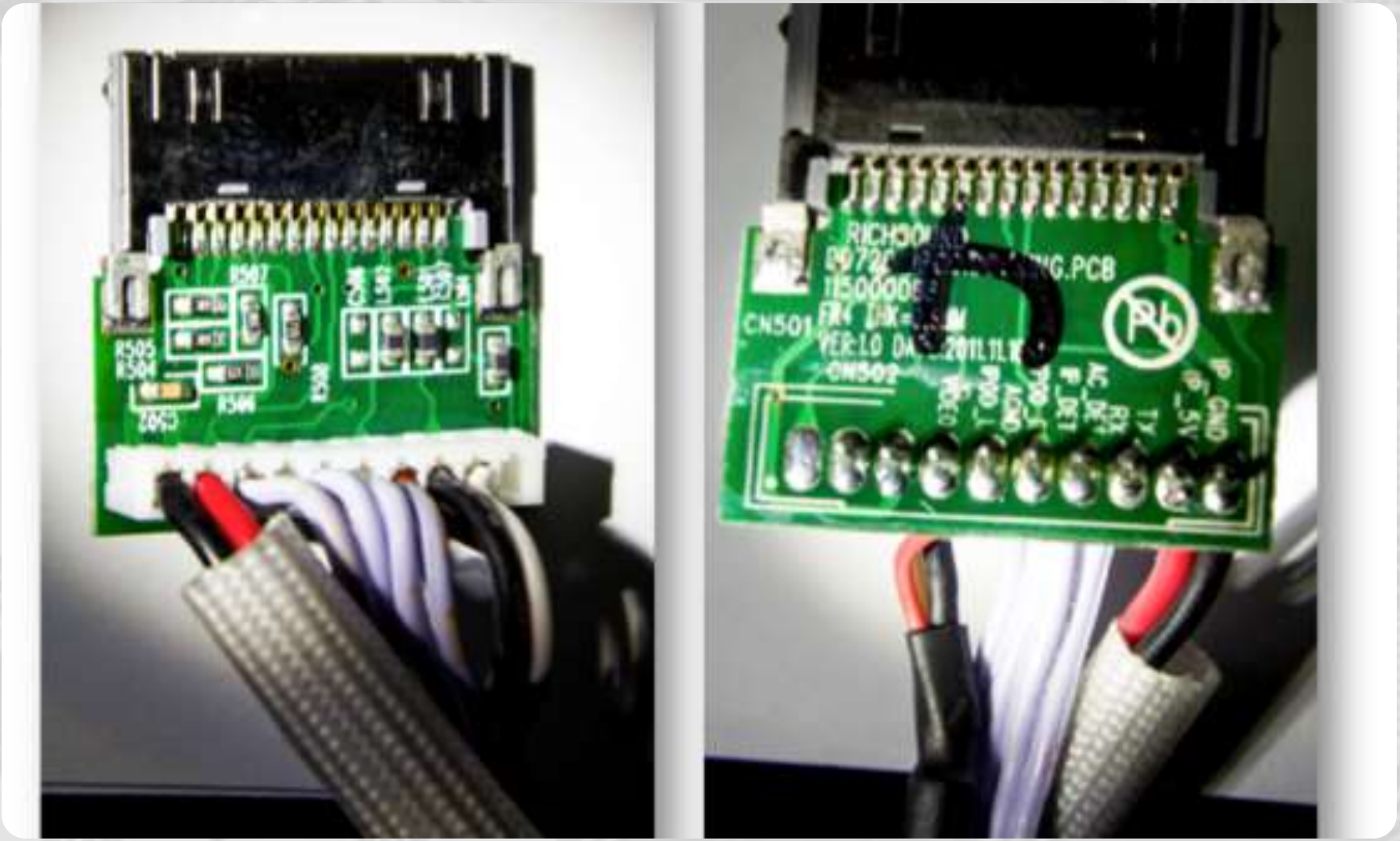
NOTES:
1. P4_7 is an input-only port.
2. Can be assigned to the pin in parentheses by a program.
3. XCIN, XCOUT can be used only for N or D version.
4. Confirm the pin 1 position on the package by referring to the package dimensions.



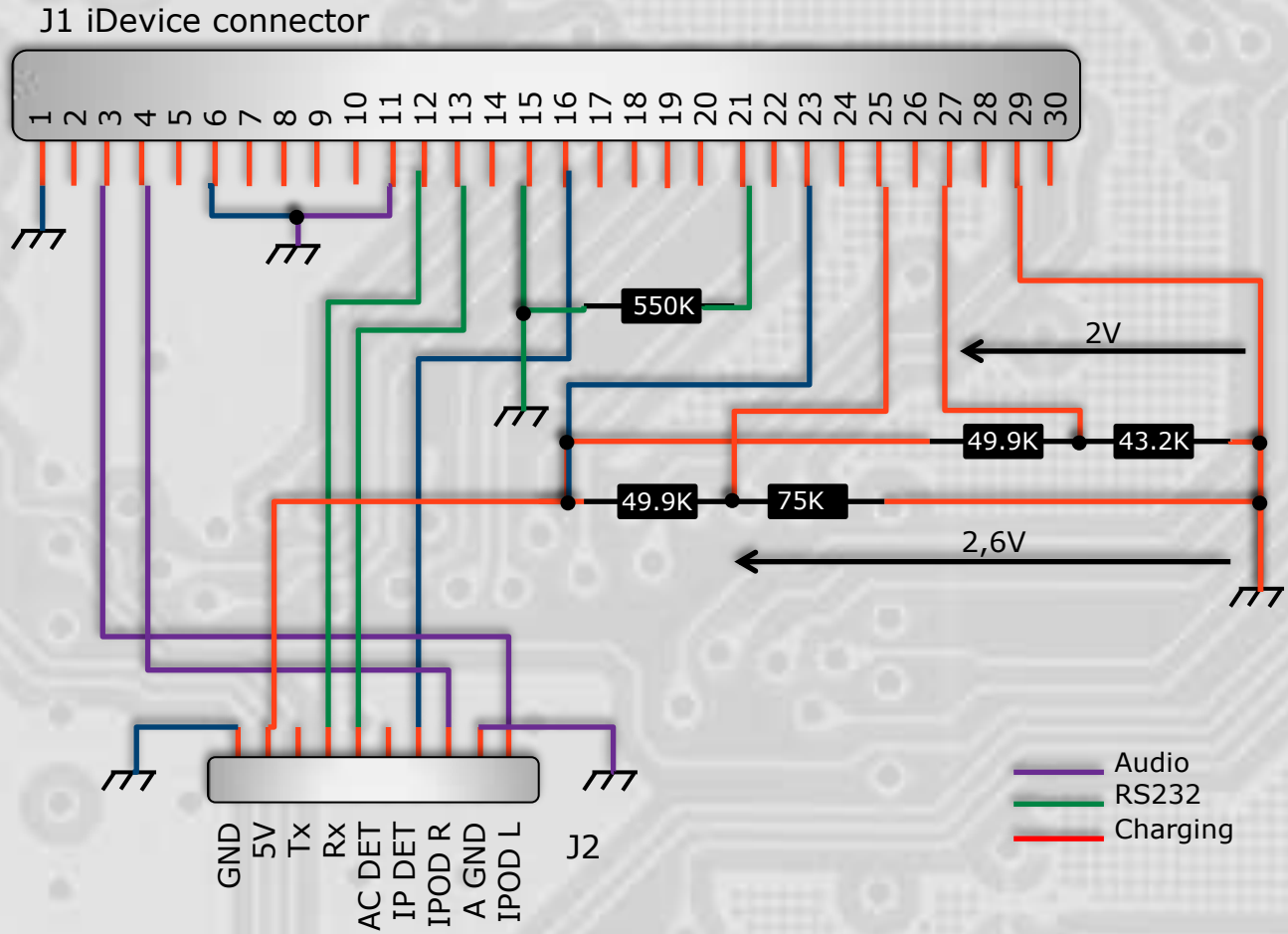
NOTES:
1. ROM size varies with MCU type.
2. RAM size varies with MCU type.
3. XCIN, XCOUT can be used only for N or D version.

Not enough resources for hosting iPown features

iDevice interface



Reversing the circuit

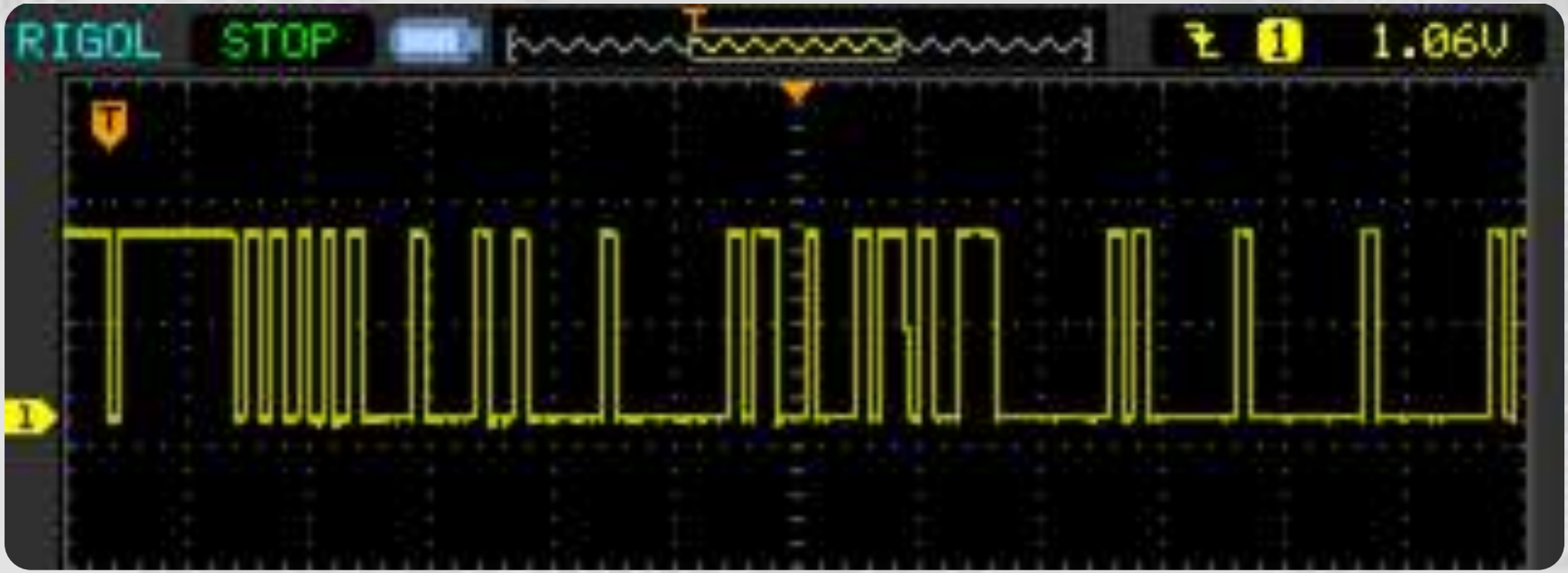


What about the Lightning connector ?

- In October 2012 Apple released the Lightning
  - Apple proprietary bus and power connector
  - Replace its previous proprietary 30-pin dock connector.
  - Using 8 pins instead of 30
  - Significantly more compact than the 30-pin dock connector
  - Can be inserted with either face up.
  - Embeds an authentication chip inside the cable.
    - Analyzing the Lightning connector will not be so easy.

- *30 pins adapters*
  - *Allows to connect 30-pin accessories to devices featuring the Lightning connector.*
  - *Successfully tested on the dock station used for our analysis*

Sniffing the communications



- *Standard 8N1 serial protocol.*
- *Data are sent @ 19200 bauds*.

Request/Response Structure

Request & Response Structure		
Field	**Size**	**Value**
Header	2	0xff 0x55
Length	1	Size of Mode + Command + Parameter
Mode	1	The mode the command is referring to.
Command	2	The two bytes command.
Parameter	0..n	Optional parameter, depending on the command.
Checksum	1	0x100 - (sum of all length / mode / command / parameter bytes) & 0xFF

```
FF 55 03 02 00 80 7B 00 00
```

Play

Summary

SOGETI

Hacking the firmware of the µC ?
  - Not relevant regarding our goal
    - We need some space to store user data…

Developing a custom dock ?
  - Challenging but too much time consuming regarding this study.
  - USB pins are not used
    - Allows connecting another device that share the same power supply

Hacking the dock and adding some hardware
  - The raspberry PI is meeting all our requirements
  - At least two USB ports
    - 1 to communicate with the connected device
    - 1 for a 3G / Wi-Fi adapter
  - 1 Ethernet port for debugging
  - GPIO (simulating user action on the dock)
  - Accepting 5V power supply.

HACKITO ERGO SUM

GOTO: HI[a]CK

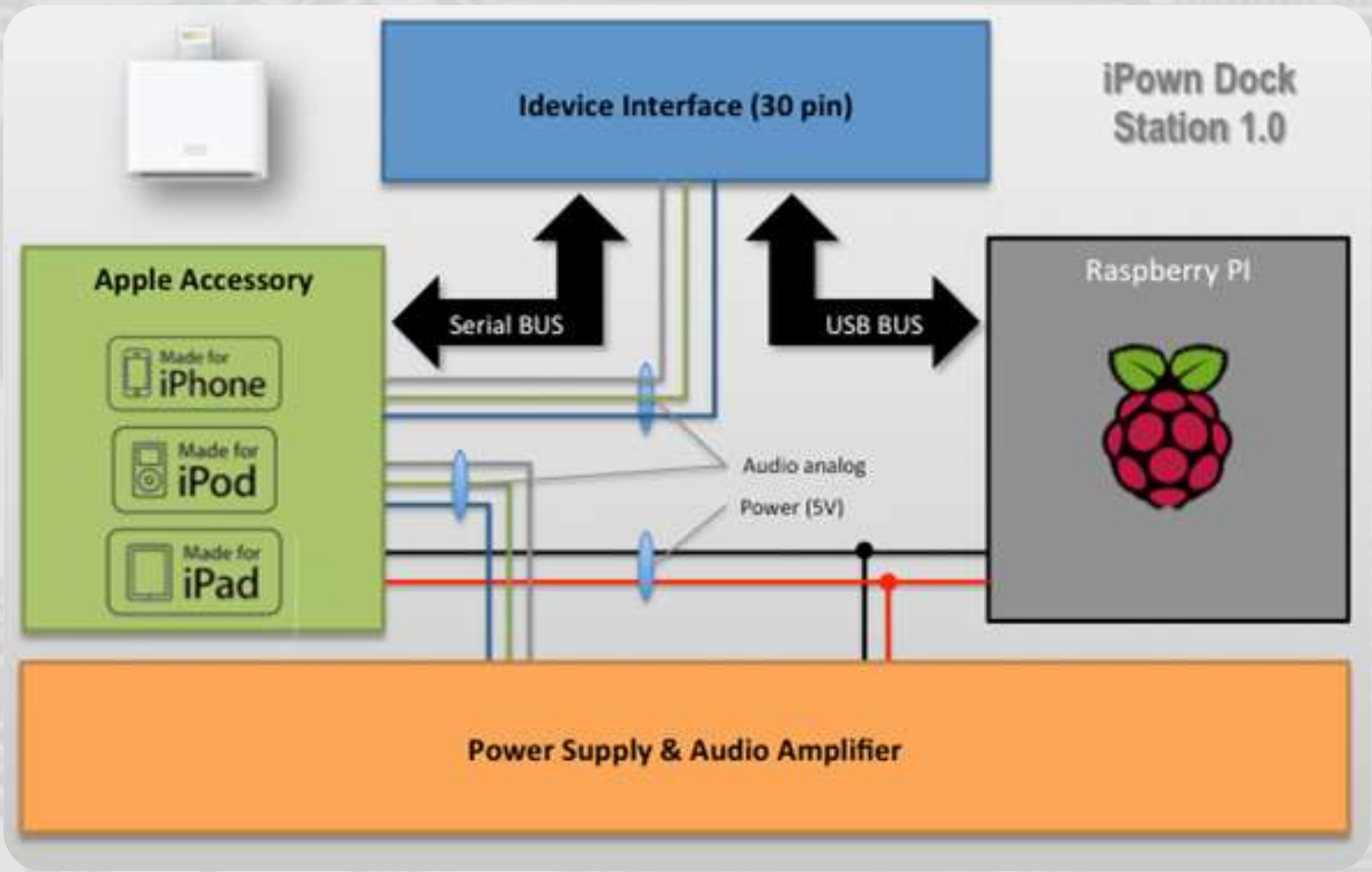
SOGETI

Weaponizing an Apple MFI accessory

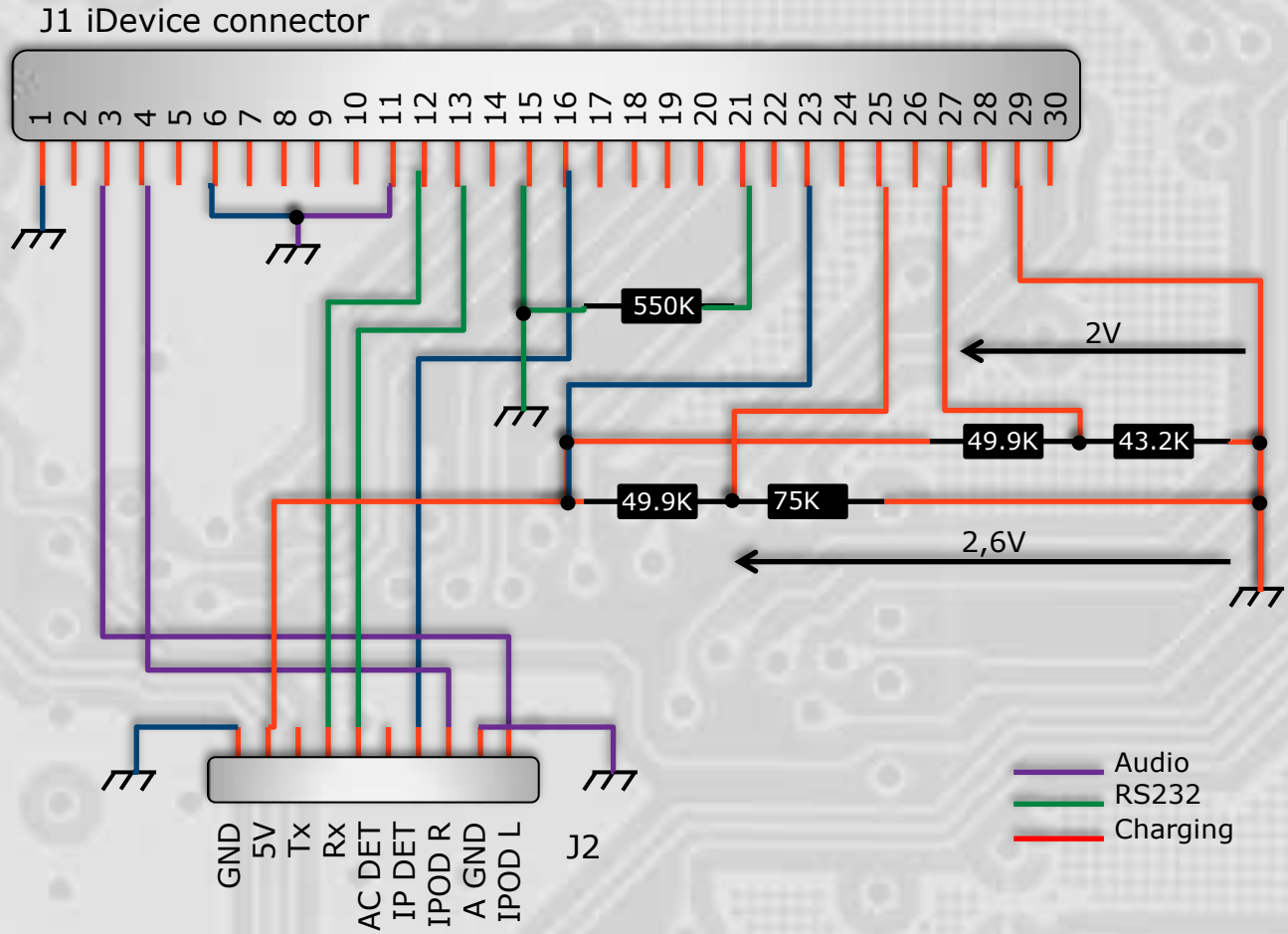
iPown Bill of materials

- # 1 Raspberry pi
- # 1 PodSocket
- # 1 PodBreakout
- # 1 USB Connector
- # 1 mini USB Connector
- # 1 WiFi USB Key
- # 1 SDcard

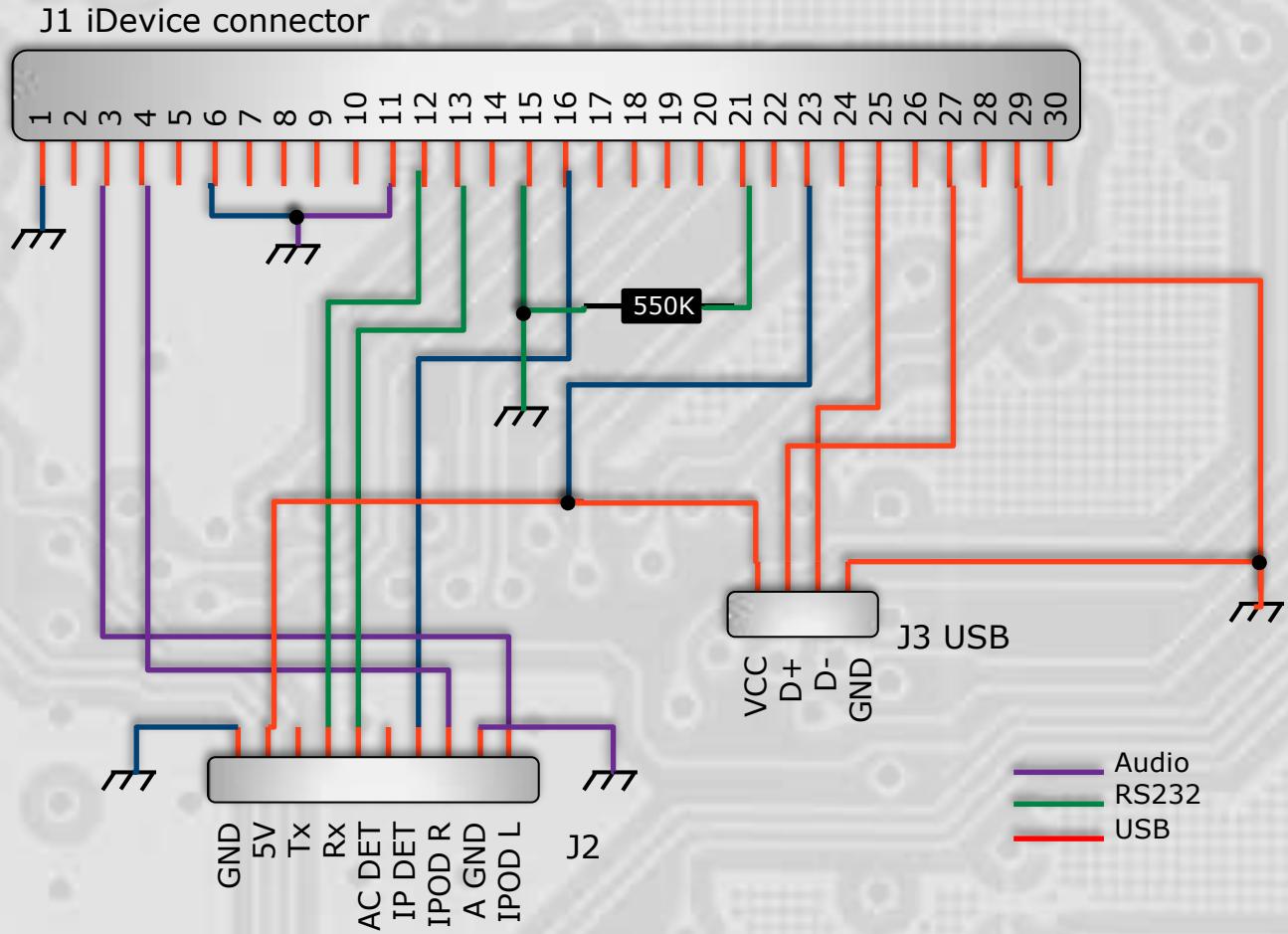
Hardware Hacking



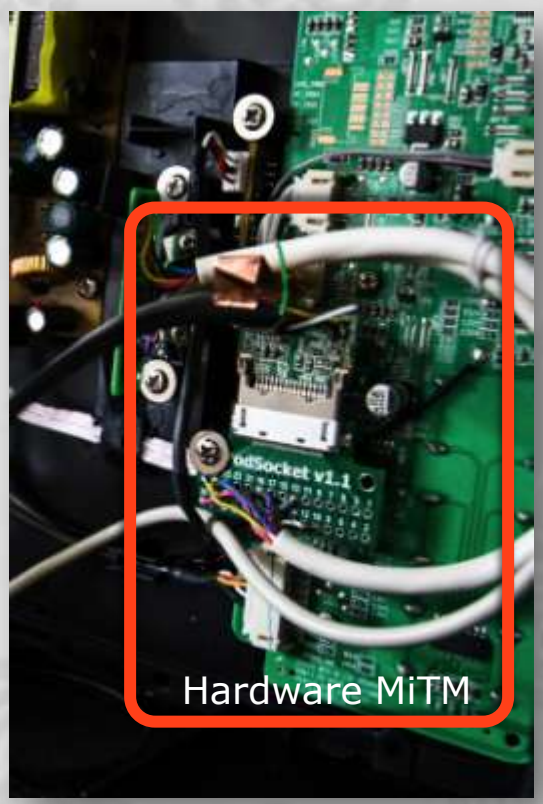
Reversing the circuit

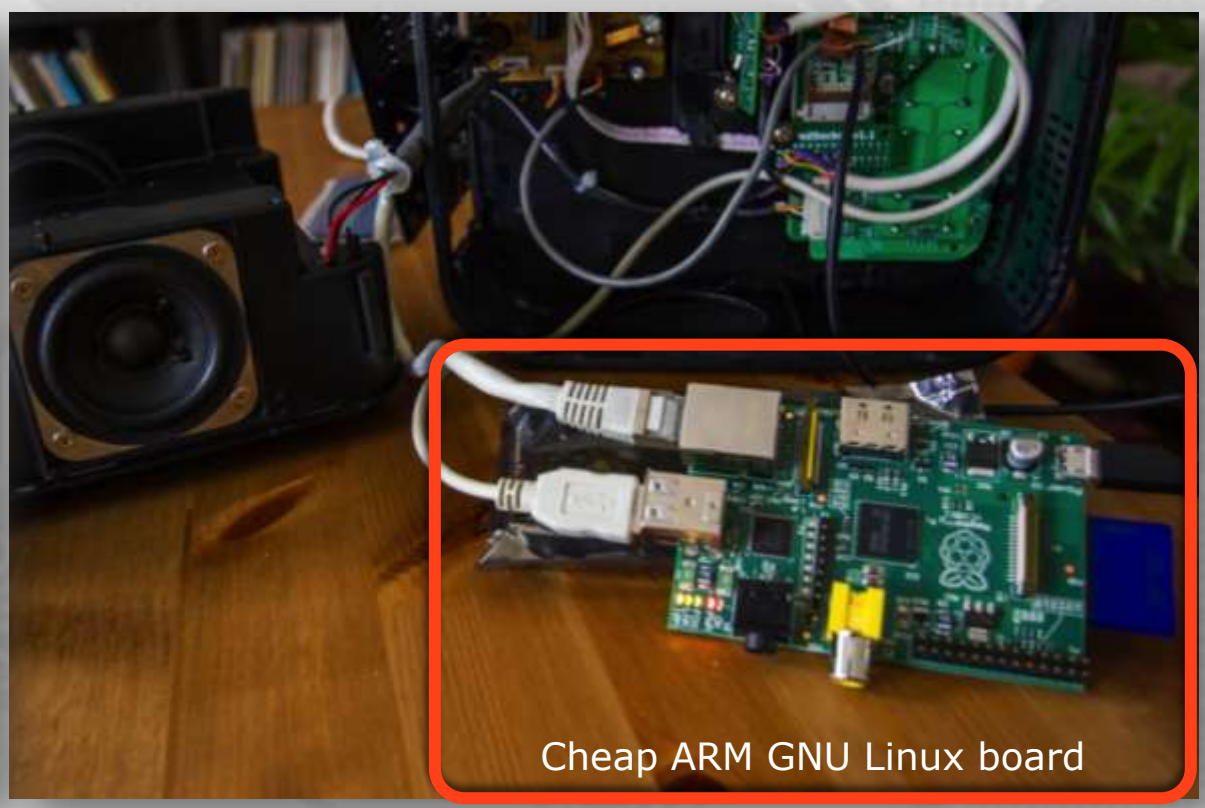


Enabling USB



Hacked MFI accessory





This dock station is now powered by

Demo

iPown

Personal Data dumper…

iPown
Personal Data dumper…

What if out alarm clock could silently jailbreak our device in our sleep when we are dreaming ?

HACKITO ERGO SUM

GOTO: H[a]CK

SOGETI

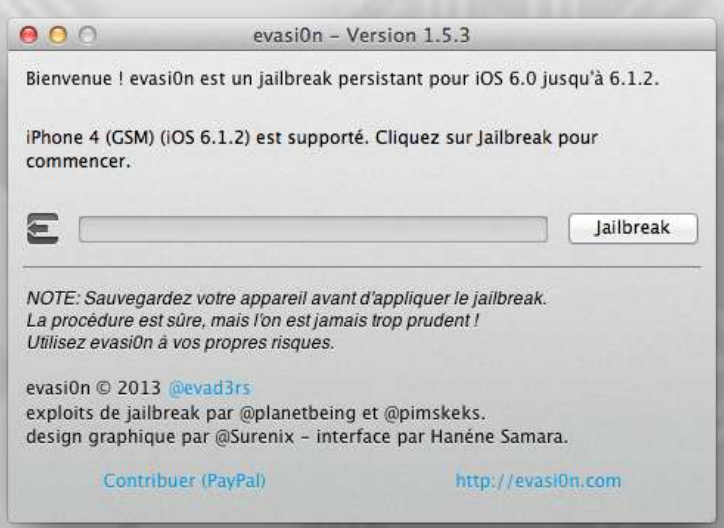
iPown 2.0

Automating Jailbreak

Public Jailbreaks

- jailbreakme.com
  - Exploits by comex, Grant Paul (chpwn), Jay Freeman (saurik) & MuscleNerd
  - Targeting MobileSafari
  - Could be used against an unwitting victim
  - Only working on old devices
- Others other recent jailbreaks (absinthe 1&2, evasi0n)
  - Require USB tethering.
  - Require User interaction
    - iDevices refuse to communicate over USB if they are locked unless they have previously paired.
    - Lower security impact
    - only useful to the phone's owner
- Are we really safe ?

evasi0n...


evasi0n – Version 1.5.3
Bienvenue ! evasi0n est un jailbreak persistant pour iOS 6.0 jusqu'à 6.1.2.
iPhone 4 (GSM) (iOS 6.1.2) est supporté. Cliquez sur Jailbreak pour commencer.
Jailbreak
NOTE: Sauvegardez votre appareil avant d'appliquer le jailbreak.
La procédure est sûre, mais l'on est jamais trop prudent !
Utilisez evasi0n à vos propres risques.
evasi0n © 2013 @evad3rs
exploits de jailbreak par @planetbeing et @pimskeks.
design graphique par @Surenix – interface par Hanéne Samara.
Contribuer (PayPal)
http://evasi0n.com


No Service
16:14
Stocks
Utilities
Facebook
JBChecker
Jailbreak
Phone
Mail
Safari
Music

Evasi0n
Initialization & Stage 1

Evasi0n Stage 1
- Pairing with the device
- Starting com.apple.mobile.file_relay service
- Retrieving the com.apple.mobile.installation.plist
  - plist file
  - caches the list of installed applications
- Activating the apple “DemoApp.app”
- Restoring Hijacked “DemoApp.app” in /var/mobile
  - Using old mobilebackup simlink trick
- Updating the caches / Rebooting the device
  - “DemoApp.app” will show up on SpringBoard after restart”

Evasi0n
Stage 2: Overview

Evasi0n Stage 2
- Chmod 777 /var/tmp/launchd
  - Injecting symbolic link 1/2
    - /var/db/timezone -> /var/tmp/launchd
  - Crarshing lockdonwd 1/2
    - Chmod 777 /var/db/timezone
- Chmod 777 /var/tmp/launchd/sock
  - Injecting symbolic link 2/2
    - /var/db/timezone -> /var/tmp/launchd/sock
  - Crashing lockdonwd 2/2
    - Chmod 777 /var/tmp/launchd/sock
- Waiting for user to launch the “DemoApp.app”
- Injecting the remount payload
- Uploading Cydia files

Evasi0n
Stage 2: Remout payload

SOGETI

Executing "DemoApp.app" => Executing the remount script

```
#!/bin/launchctl submit -l remount -o
/var/mobile/Media/mount.stdout -e
/var/mobile/Media/mount.stderr -- /sbin/mount -v -t
hfs -o rw /dev/disk0s1s1
```

```
<key>EnvironmentVariables</key>
<dict>
    <key>LAUNCHD SOCKET</key>
    <string>/private/var/tmp/launchd/sock</string>
</dict>
```

- Launchctl interfaces with launchd to load, unload daemons/agents
- launchd's IPC mechanism operates through Unix domain sockets.
- LAUNCHD SOCKET
  - Informs launchctl how to find the correct launchd socket
- Launchd runs as root and here launchctl runs as mobile
  - The socket and the demaon launchctl have been chmoded 777
  - Our mobile now able to communicate with the root user's launchd

Evasi0n
Stage 2: Remouting the file system in R/W

- launchd (runing as root) execute the remount script
  - No mount point is specified in the script
  - The kernel use the script name as mount point
    - Generating errors messages on stderr
    - The size of mount.stderr growing up
- Evasion detects the “DemoApp.app” was launched
  - Checking the size of mount.stderr
- Evasion inject another set of files unsing backup
  - Restoring timezone directory
  - Replacing “DemoApp.app” binary by a symbolink link pointing to /
    - The kernel use the script name as mount point
    - The file system is successfully remounted in RW

Evasi0n
Stage 3: Injecting final payload

- Evasi0n Stage 3
  - Creating a directory at /var/evasi0n containing 4 files
    - launchd.conf.
      - List of subcommands to run via launchctl when launchd starts
        - Remouting the filesytem in RW
        - Loading amfi.dylib library
        - Executing the evasi0n binary
    - amfi.dylib
      - Loaded with DYLD_INSERT_LIBRARY
      - Contains only lazy bindings and no TEXT section
        - No TEXT/text section means that there is nothing to sign
        - Overriding MISValidateSignature in order to always return 0
        - Allowing unsigned code execution
    - Evasi0n Binary :
      - Executed with root privilege in the early boot environment.
        - Launches the kernel exploit
    - Udid
      - Contains the UDID of the current device

Reimplementing evasi0n
Modding evasi0n installer

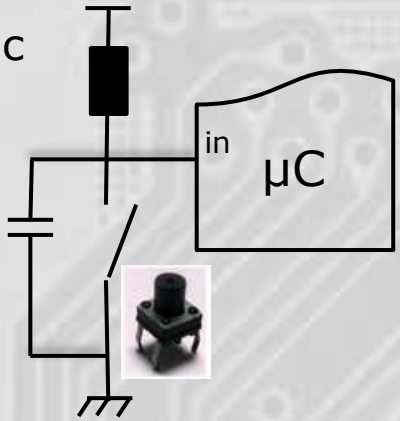
SOGETI

- Hijacking Music iPhone Application instead of “DemoApp.app”
  - Launched when connect the device is connected to a dock
  - Handle Remote accessory protocol
  - We can trigger the remount payload automatically
  - The payload can be triggered by the alarm

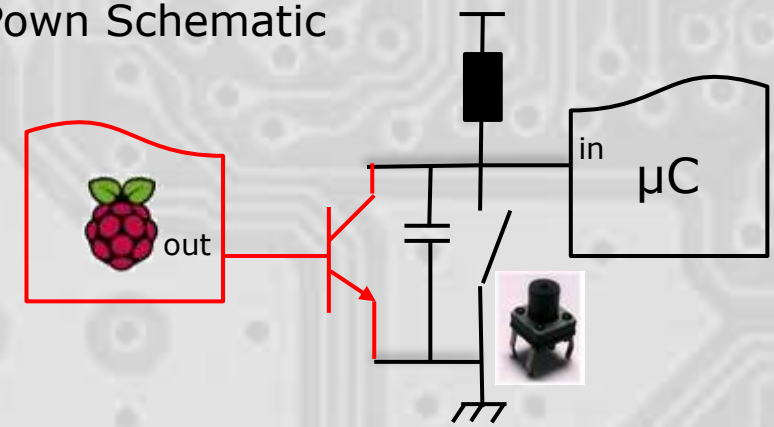
Simulating user action

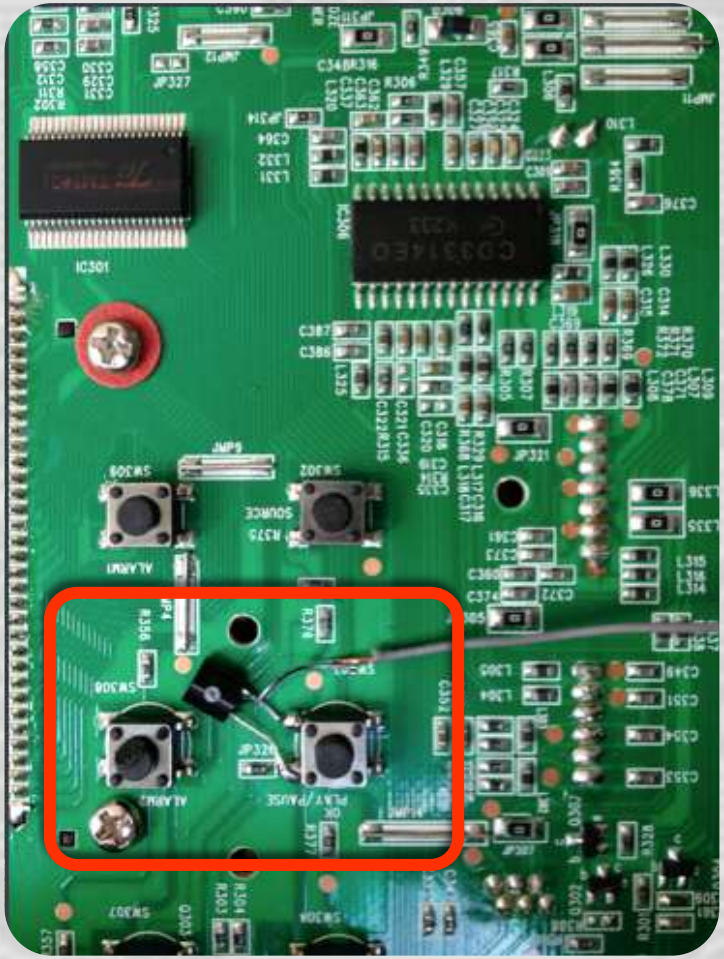
SOGETI

Original Schematic



iPown Schematic





Demo

Wake up Neo! Your phone got pwd…

Scenario

SOGETI

Room 1 : Victim

Room 2 : Attacker

Wake up Neo! Your phone got pwd…

Demo

Conclusion

Conclusion

- Apple made the choice of user experience instead of security.
  - It is possible to build up a malicious device in order to get both the data and the control of iDevices.

"When things get up close and personal,
the rule is always better safe than sorry"

Don't connect your device to an untrusted dock station

Credits

Jan0 @planetbeing @pod2g
@MuscleNerd @pimskeks @ih8sn0w @i0n1c
@p0sixninja @saurik @Comex

Thanks to all members of the jailbreak community for sharing their work and all of my friends who helped me to prepare this talk.

Don’t learn to hack but hack to learn !

Thank you for Listening
Questions ?

mathieu.renard[-at-]gotohack.org - http://www.gotohack.org