

SandJacking: Profiting from iOS Malware

Chilik Tamir

chilik@mi3security.com

Twitter: @_coreDump



Warning !!

This talk will disclose new vulnerability on all iOS devices, utilizing a new approach in malware creation running on a non-jail broken device.

This talk will not discuss targeting a jail-broken device

Who Am I

Security Researcher – iOS iNalyzer PT framework, Belch

Security Trainer:



Security Speaker:



Chief Architect of R&D at Mi3 Security

B.Sc. Biomedical Engineering

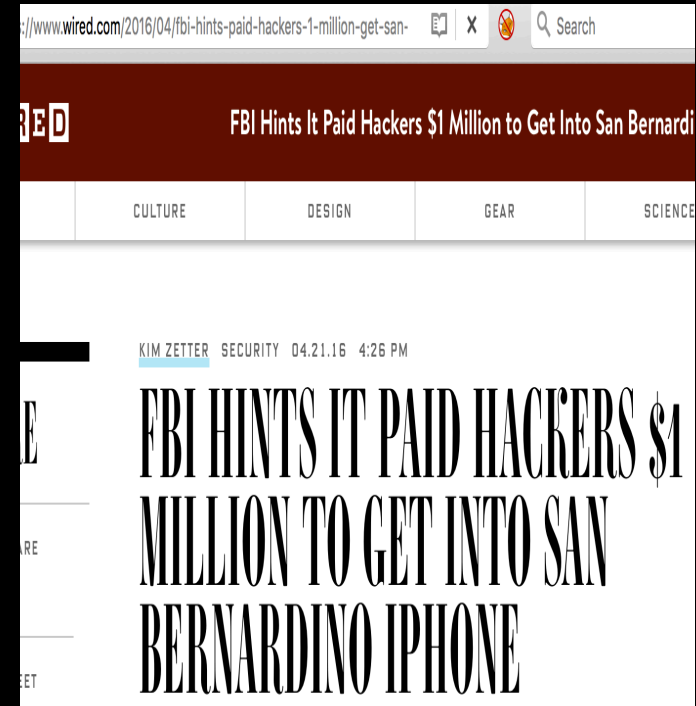
Twitter addict From Israel

Overview

- ⌘ Application Sandbox
- ⌘ Evil Client Creation & the benefits of Home brewed Malware
- ⌘ The SandJacking – When Malware takes over
- ⌘ Practical Demo: FBI Vs. ~~Apple~~ your UBER Password manager App
- ⌘ Mitigation and Detection

A Question to ponder:
FBI had allegedly paid 1.2M USD to
gain access to a device content.

how much would one pay to gain
access to your encrypted
application content?
(Vendor: we don't have keys)



About This Talk

This talk will cover the latest state of iOS malware creation and exploitation utilizing apples' home brewed certification program.



The Application Sandbox

Application Sandbox Rules

- ⌘ Every iOS Applications has its own eco-system
- ⌘ File system mapped to origin application by bundleid
- ⌘ Sandbox **Should** not be accessed from other processes

```
> Sandbox: Boardbooks(768) deny(1) file-read-data /private/var/mobile/Containers/Data/Application/39E8B5
```


- **Application Sandbox Contents:**

- ⌘ **App-Generated Documents**

- ⌘ **Application Files**

- ⌘ **Created Documents**

- ⌘ **SQLite Database**



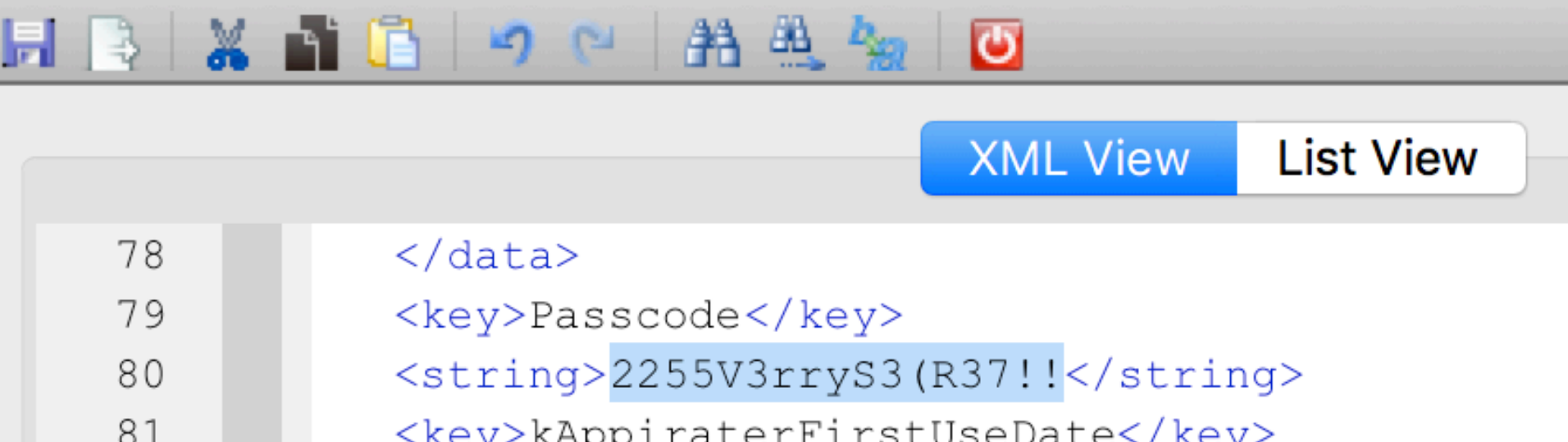
op.WhatsApp.shared/ChatStorage.sqlite

SECTIONID	ZHASH	ZPUSHNAME	ZSTANZAID	
			C634DFAB6EABBAE CDC	
			C521BE7FFAE3E6AFE6	Hi
			B4E2F6DE072BFC51BB	This is very secret

- **P.s. WhatsApp Doesn't encrypt on device**

Application Sandbox Contents:

⌘ App-Generated Library files and preferences:



The screenshot shows a file explorer window with a toolbar at the top containing icons for file operations (save, copy, paste, undo, redo, delete, refresh, power). Below the toolbar are two view options: "XML View" (selected) and "List View". The main content area displays XML code with line numbers 78 through 81 on the left. The XML content is as follows:

```
78     </data>
79     <key>Passcode</key>
80     <string>2255V3rryS3 (R37!!</string>
81     <key>kAppi raterFirstUseDate</key>
```


Application Sandbox Contents:

⌘ App-Generated Cookies:

AppDomain-com.facebook.Facebook/Library/Cookies/Cookies.binarycookies

Position: 0 / 438 (0%)

Hex Data	Value
00000000: 63 6F 6F 6B 00 00 00 02 00 00 02 9B 00 00 01 37 00 00 01 00	cook 7
00000014: 06 00 00 00 24 00 00 00 83 00 00 00 D1 00 00 00 37 01 00 00	\$ 7
00000028: CB 01 00 00 2D 02 00 00 00 00 00 00 5F 00 00 00 00 00 00 00	-
0000003C: 01 00 00 00 00 00 00 00 38 00 00 00 46 00 00 00 4D 00 00 00	8 F M
00000050: 4F 00 00 00 00 00 00 00 00 00 00 00 00 00 00 B1 14 D3 BE 41	O A
00000064: 00 00 00 32 E1 F1 BC 41 2E 66 61 63 65 62 6F 6F 6B 2E 63 6F	2 A.facebook.co
00000078: 6D 00 63 5F 75 73 65 72 00 2F 00 31 30 30 30 31 32 31 31 39	m c_user / 100012119
0000008C: 35 30 36 38 35 36 00 4E 00 00 00 00 00 00 00 00 00 00 00 00	506856 N
000000A0: 00 00 00 38 00 00 00 00 46 00 00 00 4A 00 00 00 4C 00 00 00	8 F J L
000000B4: 00 00 00 00 00 00 00 00 00 00 00 B1 14 D3 BE 41 00 00 00 32 E1	A 2
000000C8: F1 BC 41 2E 66 61 63 65 62 6F 6F 6B 2E 63 6F 6D 00 63 73 6D	A.facebook.com csm
000000DC: 00 2F 00 32 00 66 00 00 00 00 00 00 00 00 00 00 00 00 00 00	/ 2 f
000000F0: 00 38 00 00 00 46 00 00 00 4B 00 00 00 4D 00 00 00 00 00 00	8 F K M
00000104: 00 00 00 00 00 00 00 80 18 24 5A C0 41 00 00 00 32 E1 F1 BC	\$Z A 2
00000118: 41 2E 66 61 63 65 62 6F 6F 6B 2E 63 6F 6D 00 64 61 74 72 00	A.facebook.com datr
0000012C: 2F 00 5F 36 68 42 56 77 6F 62 70 4B 75 77 31 4B 36 71 33 30	/ __6hBVwobpKuw1K6q30
00000140: 36 56 7A 6F 47 78 00 94 00 00 00 00 00 00 00 00 00 00 00 00	6VzoGx
00000154: 00 00 00 38 00 00 00 46 00 00 00 49 00 00 00 4B 00 00 00 00	8 F I K
00000168: 00 00 00 00 00 00 00 00 00 00 00 B1 14 D3 BE 41 00 00 00 32 E1	A 2
0000017C: F1 BC 41 2E 66 61 63 65 62 6F 6F 6B 2E 63 6F 6D 00 66 72 00	A.facebook.com fr
00000190: 2F 00 30 66 71 4E 49 44 77 30 6E 35 55 79 39 57 32 41 35 2E	/ 0fqNIDw0n5Uy9W2A5.



Application Sandbox Contents:

- ⌘ App-Generated Documents
- ⌘ App-Generated Libraries and preferences
- ⌘ App-Generated Cookies
- ⌘ App-Generated Temporary files

Highly Sensitive information
Must not be accessed by Malware !!!

iOS Malware



NOM NOM NOM

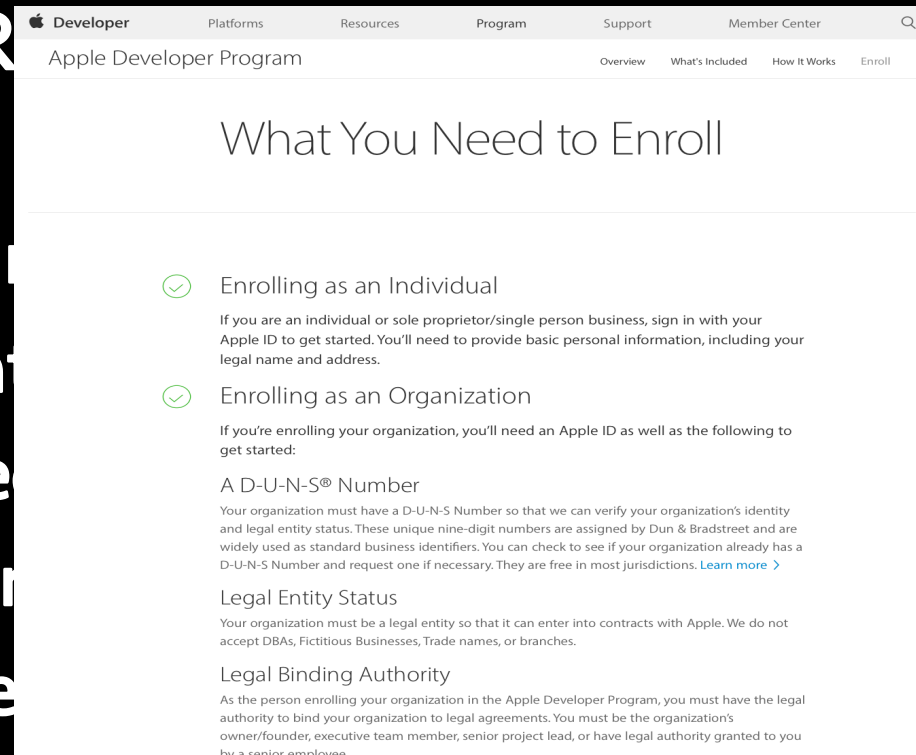
**In order for iOS Malware to run on a device
It must follow Apple's rules.....**

iOS Malware Playground Rules

- ⌘ All code must be signed
- ⌘ All apps are subjected to a review process
- ⌘ All certificates require identification
- ⌘ All installation are validated on device
- ⌘ Any misbehaving developer will be accountable
- ⌘ Every installation on the device requires a signed package

iOS Malware Playground R

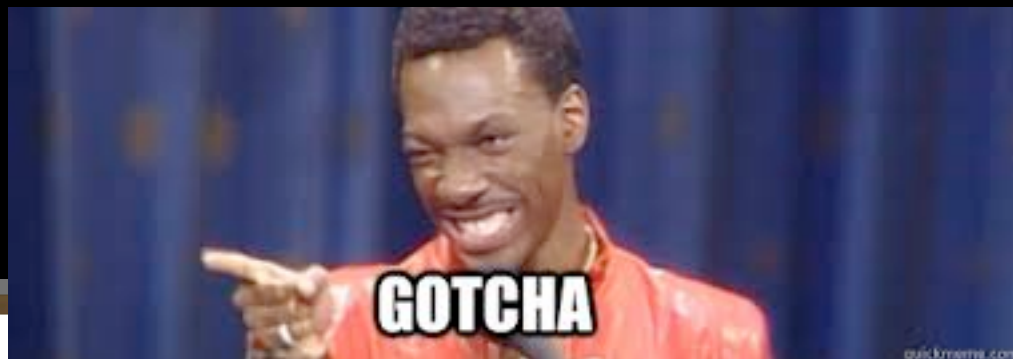
- ⌘ All code must be signed
- ⌘ All apps are subjected to a
- ⌘ All code is required
- ⌘ All installations are validated
- ⌘ Any misbehaving developer
- ⌘ Every installation on the device package



The screenshot shows the Apple Developer Program enrollment page. The navigation bar includes 'Developer', 'Platforms', 'Resources', 'Program', 'Support', and 'Member Center'. The main heading is 'Apple Developer Program' with sub-links for 'Overview', 'What's Included', 'How It Works', and 'Enroll'. The main content area is titled 'What You Need to Enroll' and lists two options: 'Enrolling as an Individual' and 'Enrolling as an Organization'. The 'Enrolling as an Organization' section lists requirements: a D-U-N-S® Number, Legal Entity Status, and Legal Binding Authority.

<https://developer.apple.com/programs/enroll/>

If you *misbehave*:



Stefan Esser

@i0n1c

 Follow

Apple's decision to ban us and all the frauds coming up nicely exposes how defenseless their AppReview is against actual bad people.

RETWEETS

18

LIKES

27

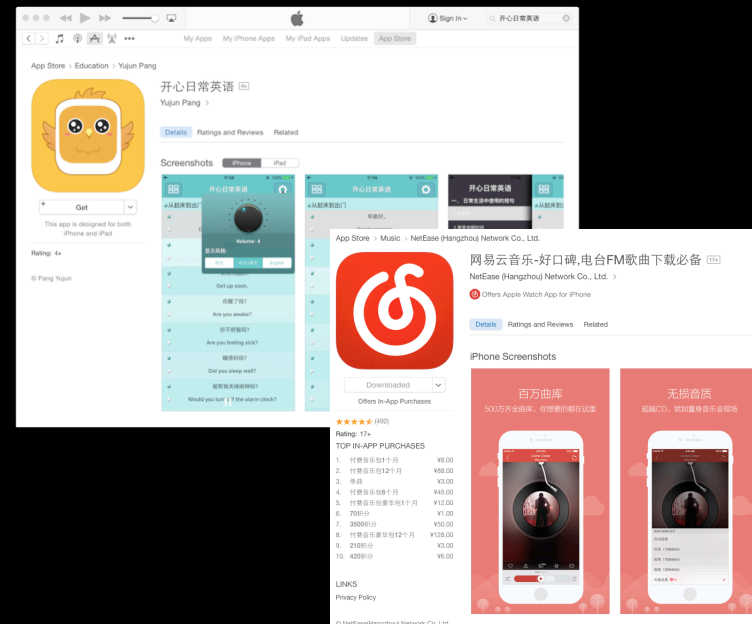


10:20 PM - 17 May 2016



Malware Distribution Tracks – App Store

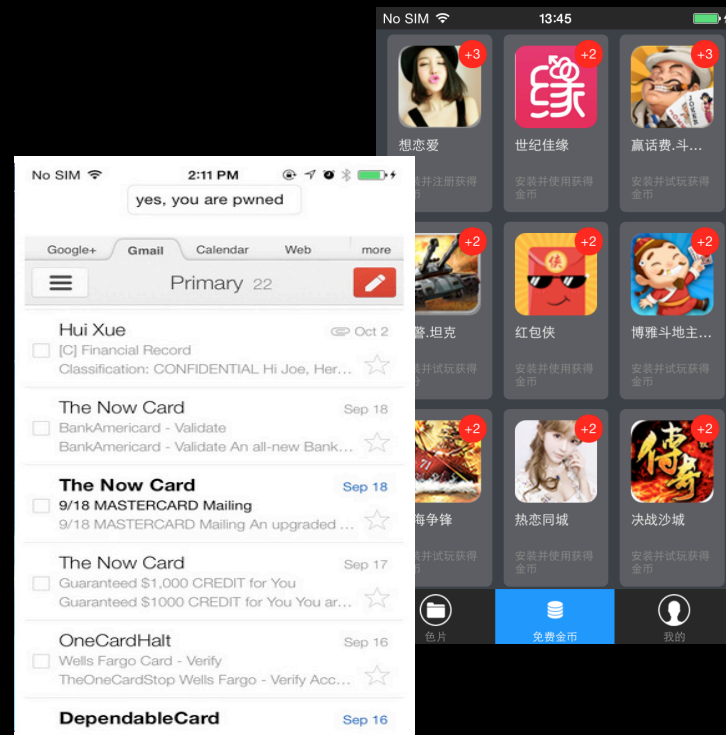
- ⌘ **ZergHelper (Claud Xiao, paloalto networks)**
- ⌘ **xCodeGohst (Claud Xiao, paloalto networks)**



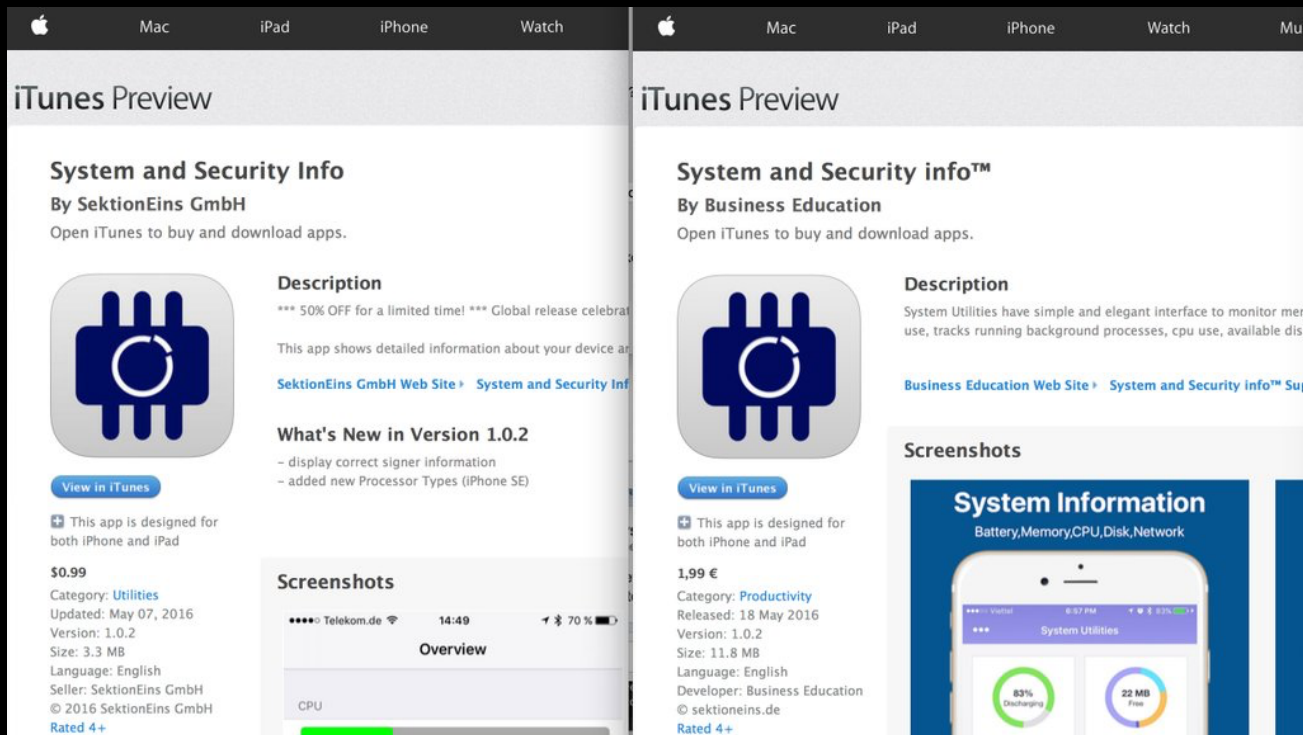
Malware Distribution Tracks – Distributor

⌘ **Yispector, WireLurker (Claud Xiao, paloalto networks)**

⌘ **masque-attack (Hui Xue, Tao Wei, Yulong Zhang, FireEye)**



Malware Distribution Tracks – ScamWare



The image displays two side-by-side screenshots of the iTunes Store interface, showing the details for two different versions of an app named 'System and Security Info'.

Left Screenshot: 'System and Security Info' by SektionEins GmbH

- Price:** \$0.99
- Category:** Utilities
- Updated:** May 07, 2016
- Version:** 1.0.2
- Size:** 3.3 MB
- Language:** English
- Seller:** SektionEins GmbH
- © 2016 SektionEins GmbH**
- Rated 4+**

Description: *** 50% OFF for a limited time! *** Global release celebration... This app shows detailed information about your device and... [SektionEins GmbH Web Site](#) [System and Security Info](#)

What's New in Version 1.0.2

- display correct signer information
- added new Processor Types (iPhone SE)

Screenshots: Overview, CPU

Right Screenshot: 'System and Security info™' by Business Education

- Price:** 1,99 €
- Category:** Productivity
- Released:** 18 May 2016
- Version:** 1.0.2
- Size:** 11.8 MB
- Language:** English
- Developer:** Business Education
- © sektioneins.de**
- Rated 4+**

Description: System Utilities have simple and elegant interface to monitor memory use, tracks running background processes, cpu use, available disk... [Business Education Web Site](#) [System and Security info™ Support](#)

Screenshots: System Information (Battery, Memory, CPU, Disk, Network), System Utilities (83% Discharging, 22 MB Free)

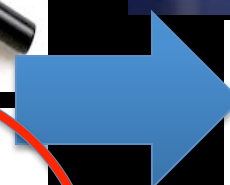
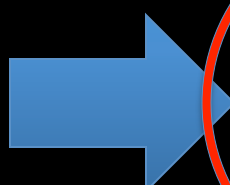
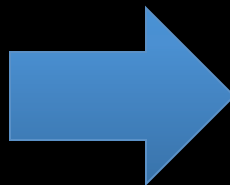
Historical Malware Capabilities

- ⌘ Abuse private API to install and remove apps programmatically and more...
- ⌘ Abuse access to Address Book
- ⌘ Abuse access to Calendar
- ⌘ Abuse access to Photo EXIF metadata
- ⌘ Abuse access to Microphone recording
- ⌘ Abuse pin-point GPS Locationing

Historical Malware capabilities

- ⌘ De-anonymization of user
- ⌘ Hijacking of legitimate CFURL calls
- ⌘ Phishing credentials
- ⌘ Polymorphism by remote updates (e.g. Lua based application with remote update)

Distribution & Attribution:

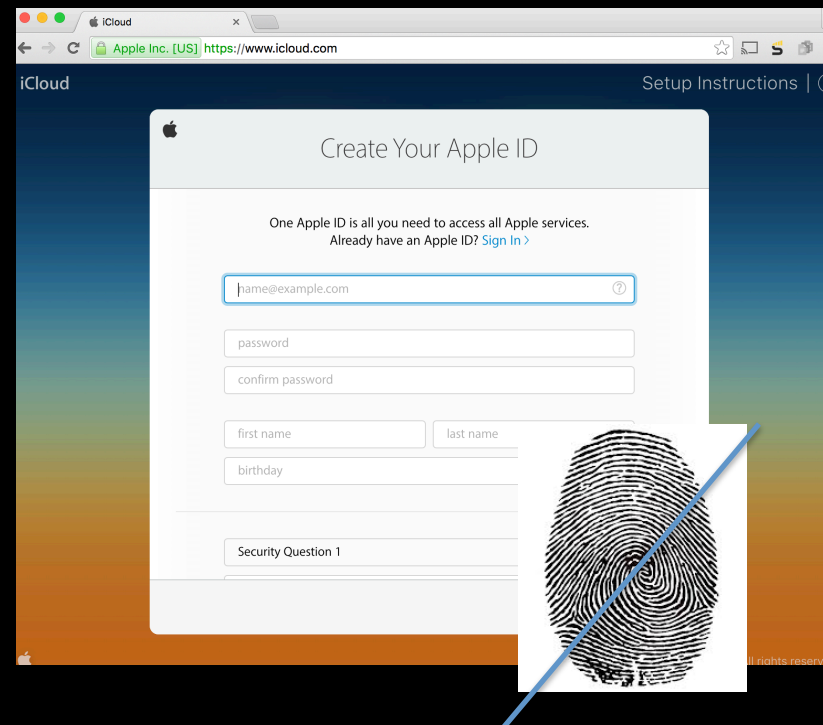


Home Brewed Evil Clients Malware

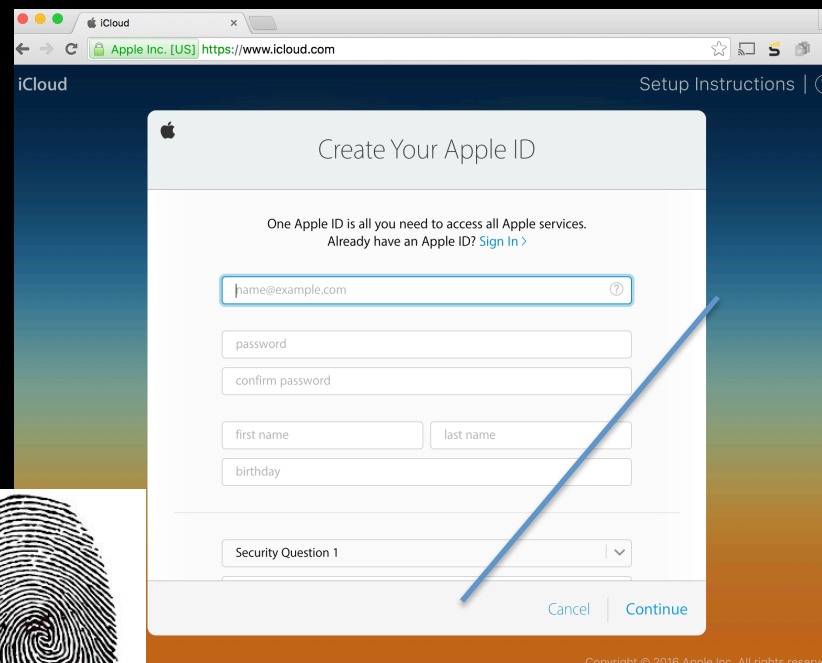


“Sign in With Apple ID”

- ⌘ Anonymous Developer
- ⌘ No target for attribution
- ⌘ Can always regenerate
- ⌘ Resigning with new Certs



Creating Anonymous / Fake Apple ID Demo



iCloud Setup Instructions | ?

Create Your Apple ID

One Apple ID is all you need to access all Apple services.
Already have an Apple ID? [Sign In >](#)

ⓘ

Security Question 1

[Cancel](#) [Continue](#)



Capabilities Available to Developers

	Sign in with Apple ID	Apple Developer Program members
App Groups	•	•
Background Modes	•	•
Data Protection	•	•
HealthKit	•	•
HomeKit	•	•
Inter-App Audio	•	•
Keychain Sharing	•	•
Maps	•	•
Wireless Accessory Configuration	•	•

Apple Pay
Associated Domains
Game Center
iCloud / CloudKit
In-App Purchase
Passbook / Wallet
Personal VPN
Push Notifications

Malware Capabilities

⌘ Pinpoint GPS Locationing – Abusage



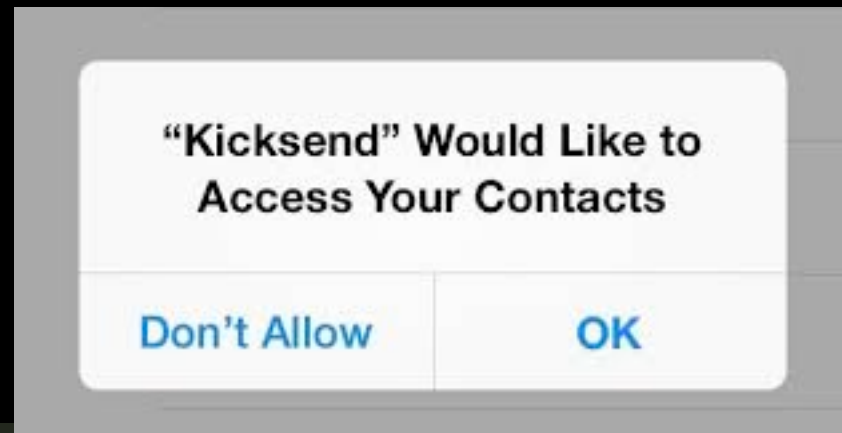
```
- (void) locationManager:(CLLocationManager *)manager didUpdateToLocation:(CLLocation *)
```

```
CLLocation *location;  
location = [manager location];  
CLLocationCoordinate2D coordinate = [location coordinate];  
_currentLocation = [[CLLocation alloc] initWith:  
_currentLocation = newLocation;  
_longitude = [NSString stringWithFormat:@"%f", coordinate.longitude];  
_latitude = [NSString stringWithFormat:@"%f", coordinate.latitude];
```



Malware Capabilities

⌘ Address-Book Stealing



```
case CNAuthorizationStatus.Authorized :
```

```
NSArray *keysToFetch = @[CNContactGivenNameKey, CNContactFamilyNameKey, CNContactPhoneNumbersKey];  
NSString *containerId = [self.CN_contacts defaultCenterIdentifier];  
NSPredicate *predicate = [CNContact predicateForContactsInContainerWithIdentifier:containerId];  
self.allContacts = [self.CN_contacts unifiedContactsMatchingPredicate:predicate keysToFetch:keysToFetch  
error:nil];
```



Malware Capabilities

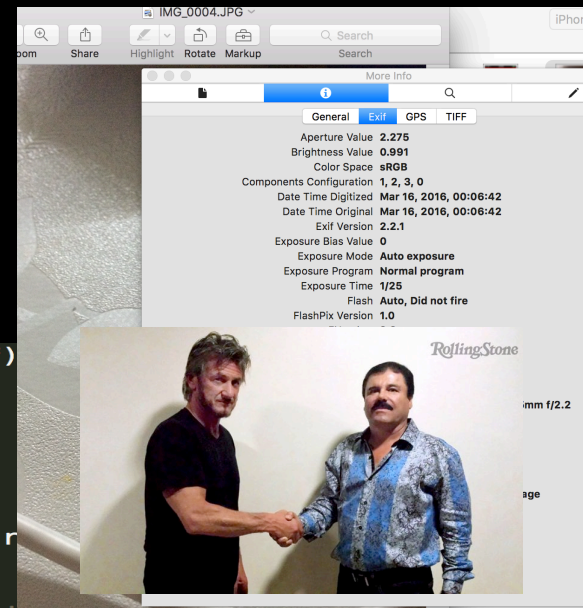
⌘ EXIF data extraction (GPS...)

```
CGImageSourceRef source = CGImageSourceCreateWithURL((__bridge CFURLRef)
    if (source == NULL) {
        NSLog(@"Source is NULL");
    }

    //get all the metadata in the image
    NSDictionary *metadata = (__bridge NSDictionary *)CGImageSource
        , NULL);

    //make the metadata dictionary mutable so we can add properties to it
    NSMutableDictionary *metadataAsMutable = [metadata mutableCopy];

    NSMutableDictionary *EXIFDictionary = [[metadataAsMutable objectForKey:(NSString *)
        kCGImagePropertyExifDictionary]mutableCopy];
    NSMutableDictionary *GPSDictionary = [[metadataAsMutable objectForKey:(NSString *)
        kCGImagePropertyGPSDictionary]mutableCopy];
    NSMutableDictionary *RAWDictionary = [[metadataAsMutable objectForKey:(NSString *)
        kCGImagePropertyRawDictionary]mutableCopy];
    NSMutableDictionary *GIFDictionary = [[metadataAsMutable objectForKey:(NSString *)
        kCGImagePropertyGIFDictionary]mutableCopy];
```



Malware Capabilities

⌘ Calendar Access

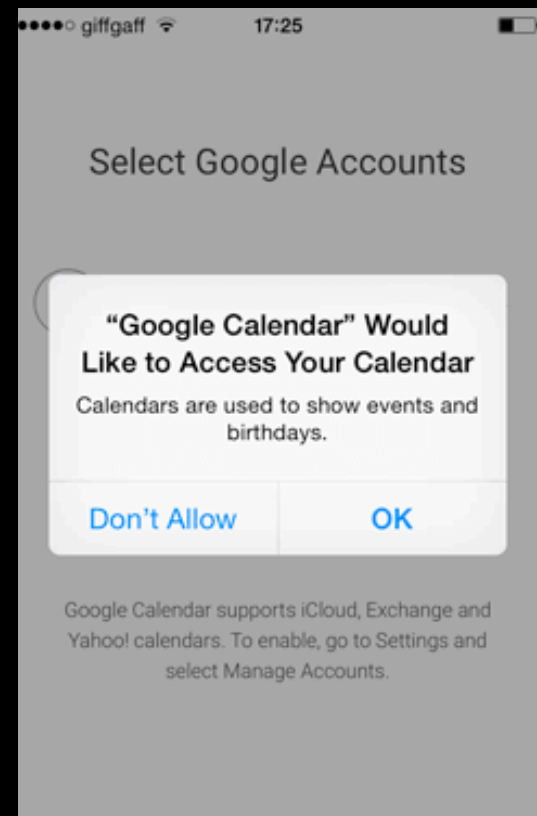
```
// Get the appropriate calendar
NSCalendar *calendar = [NSCalendar currentCalendar];

// Create the start date components
NSDateComponents *oneDayAgoComponents = [[NSDateComponents alloc] init];
oneDayAgoComponents.day = -1;
NSDate *oneDayAgo = [calendar dateByAddingComponents:oneDayAgoComponents
                                                    toDate:[NSDate date]
                                                    options:0];

// Create the end date components
NSDateComponents *oneYearFromNowComponents = [[NSDateComponents alloc] init];
oneYearFromNowComponents.year = 1;
NSDate *oneYearFromNow = [calendar dateByAddingComponents:oneYearFromNowComponents
                                                         toDate:[NSDate date]
                                                         options:0];

// Create the predicate from the event store's instance method
NSPredicate *predicate = [store predicateForEventsWithStartDate:oneDayAgo
                                                         endDate:oneYearFromNow
                                                         calendars:nil];

// Fetch all events that match the predicate
NSArray *events = [store eventsMatchingPredicate:predicate];
```



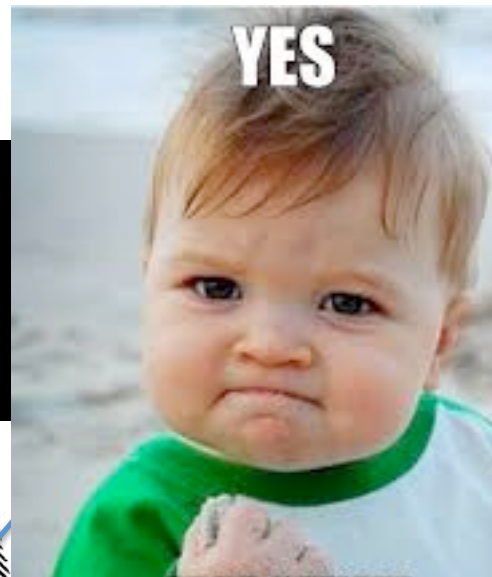
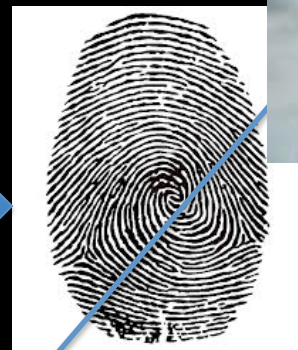
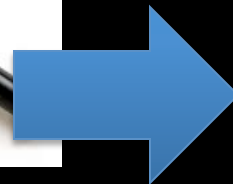
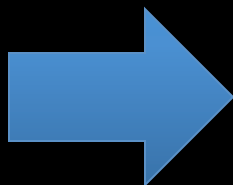
Malware Capabilities

⌘ Health Kit Access

- If the user consents, you may share his or her HealthKit data with a third party for medical research.
- You must clearly disclose to the user how you and your app will use their HealthKit data.

You must also provide a privacy policy for any app that uses the HealthKit framework. You can find guidance on creating a privacy policy at the following sites:

- Personal Health Record model (for non-HIPAA apps): <http://www.healthit.gov/policy-researchers-implementers/personal-health-record-phr-model-privacy-notice>
- HIPAA model (for HIPAA covered apps): <http://www.hhs.gov/ocr/privacy/hipaa/modelnotices.html>



Malware Creation

⌘ Evil Client Demo

```
evilskyper — crypted -r 127.0.0.1 — 115x48
java -X...6.39.jar | idevicesyslog ... ..acker — -bash ...skype — -bash ... irb rvm...vm/bin ... crypted...27.0.0.1
## General Public License, version 3 (or later), available here:
## http://www.gnu.org/licenses/gpl-3.0.html.
##
## Objects created using Theos and/or Logos are not considered derivative works
## (from a licensing standpoint, or, for that matter, any other standpoint) and
## are, as such, not required to be licensed under the GNU GPL.
##
## The included project templates are license-free. The use of a template does
## not confer a license to your project.
##
## DISCLAIMER:
## This tool is an education tool for demonstration PoC of iOS Malware, Only!
##
##!# Connect the Device to the USB port, and press any key to continue or ^C to abort...
##!# Running Make to Create the evil .dylib
##!# Running Make to Create the evil .dylib: Done
##!# Preparing App Containers..
##!# Preparing App Containers: Done
##!# Preparing Application provisioning Profile..
-----
The login information you enter will be stored in your Mac OS Keychain
You can also pass the password using the 'FASTLANE_PASSWORD' env variable
More information about it on GitHub: https://github.com/fastlane/credentials_manager
-----
Username: merrypoppines@gmail.com
##!# Preparing Application provisioning Profile: Done
##!# Creating Evil Client with provisioning Profile..
##!# Creating Evil Client with provisioning Profile: Done
##!# Installing Evil Client provisioning Profile to USB device..
##!# Installing Evil Client provisioning Profile to USB device: Done
##!# Installing Evil Client to USB device..!# Installing Evil Client to USB device: Done
##!# PoC app is ready...
iLabMBP:evilskyper _coredump$ crypted -r 127.0.0.1:31337
cy# UIApp
{"-SKPApplication: 0x16c19de0"}
cy# UIApp.keyWindow.firstResponder
{"-UITextField: 0x16e1dc00; frame = (40 184 5: 240 24); text = ''; clintToBounds = YES; opaque = NO; gestureRecogni-
zers = -NSArray: 0x16f8c450; layer = -CA
cy# UIApp.keyWindow.firstResponder.text=
{"Skype is now controlled by Attacker"}
cy#
```

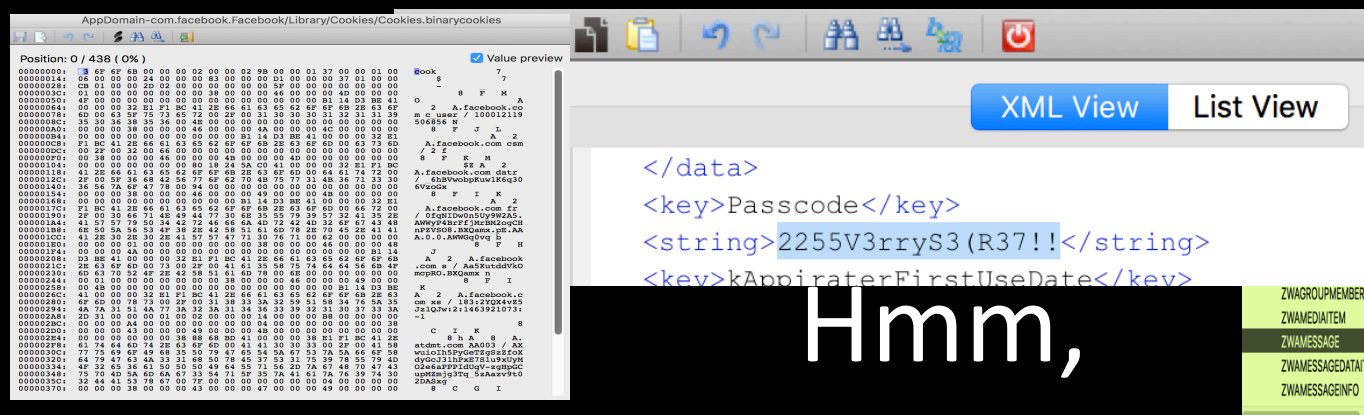
DEMO



~~iPhone Repair shops~~ iPwn shops

DEVICE + PASSCODE FTW !!!



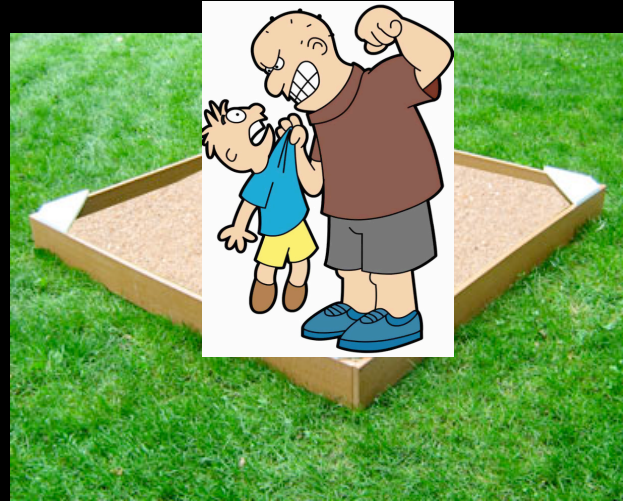


Hmm,

Can an Evil Client Access the Sandbox “Juicy” Content ?

- ZWAGROUPMEMBERSCH
- ZWAMEDIAITEM
- ZWAMESSAGE
- ZWAMESSAGE DATAITEM
- ZWAMESSAGEINFO





Sandjacking: Evil client hijacking of host
Application Sandbox content

SandJacking, Sample Use case:

An FBI vs. ~~Apple~~ Encrypted Application:

- ⌘ **Law Enforcement / Forensic Extraction**
- ⌘ **Application Content Encryption**
- ⌘ **Brute-force implementation**
- ⌘ **Unknown Application passphrase / passcode**
- ⌘ **DEMO: Secure application with WIPE Enabled**



SandJacking on iOS < 8.3

- ⌘ **Install Evil Client Overwriting the original Application**
- ⌘ **Brute-force**
- ⌘ **Profit \$\$\$**

Alas....

```
installed application's application-identifier string (GG7XEEU22E.com.dataviz.PasswordsPlus); rejecting upgrade.}
May 23 00:35:10 iPhone mobile_installation_proxy[214] <Error>: 0x16e12f000 handle_install: Installation failed: Error Domain=LaunchServicesError Code=0
"(null)" UserInfo={Error=MismatchedApplicationIdentifierEntitlement, ErrorDescription=Upgrade's application-identifier entitlement string (PZ5YCPA8WR.
com.dataviz.PasswordsPlus-patchedc2b61f8ab286c27e) does not match installed application's application-identifier string (GG7XEEU22E.com.dataviz.Passwor
dsPlus); rejecting upgrade.}
```

- ⌘ Apple had patched this vulnerability on any iOS > 8.3
- ⌘ Installation process will deny upgrade of any application

Hmm, it seems that apple had patch
the front door...

But apperantly they left a backdoor
wide open ..!

Introducing SandJacking on any iOS

- ⌘ Apple had patched the installation process
- ⌘ Apple left the Restore process unpatched
- ⌘ So, we install our Evil client prior to a restore
- ⌘ Then restore process will grant our Evil client with sandbox access (As there is No Validation)
- ⌘ Profit \$\$\$\$

Introducing SandJacking on any iOS > 8.3

- ⌘ Backup Device
- ⌘ Delete Original Application
- ⌘ Install Evil Client with Tainted functionality
- ⌘ Restore Device from Backup
- ⌘ Brute-force
- ⌘ Profit \$\$\$

Demo time

SandJacking: Timeline

- ⌘ **2015-12-27: discovery**
- ⌘ **2016-01-27: notification to Apple, ask for patch release**
- ⌘ **2016-02-05: Apple responds, initiating a responsible disclosure**
- ⌘ **2016-05-23: fix in progress**

SandJacking: *SandJacker* - The Tool

Due to responsible disclosure, and Apple's request, the *SandJacker* tool Will be released once Apple has a patch available, Stay tuned:

<https://www.mi3security.com>

Questions & Answers

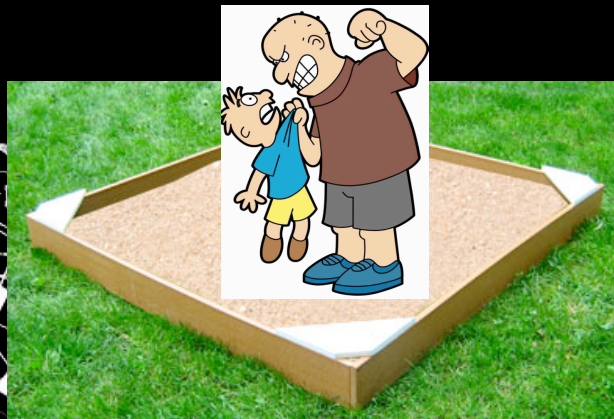
Other Resources

- ⌘ Chilik Tamir, Su-A-Cyber: Home-brewing iOS malware like a BO\$\$ BHAsia 2016

<https://www.blackhat.com/docs/asia-16/materials/asia-16-Tamir-Su-A-Cyber-Homebrewing-Malware-For-iOS-Like-A-BOSS.pdf>

- ⌘ Claud Xiao, Palo-Alto Networks,

<http://researchcenter.paloaltonetworks.com/author/claud-xiao/>



SandJacking: Profiting from iOS Malware

Chilik Tamir

chilik@mi3security.com

Twitter: @_coreDump