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 pounder:
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 Application has its own eco-system
- File system mapped to origin application by bundleId
- Sandbox *Should* not be accessed from other processes
• Application Sandbox Contents:

- App.WhatsApp.shared/ChatStorage.sqlite

• App-Generated Documents
  - Application Files
  - Created Documents
  - SQLite Database

• P.s. WhatsApp Doesn't encrypt on device

<table>
<thead>
<tr>
<th>ASSESSIONID</th>
<th>ZPHASH</th>
<th>ZPUSHNAME</th>
<th>ZSTANZAID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>C634DFAB6EABBAECD</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>C521BE7FFAE3E6AFE6</td>
<td></td>
<td>Hi</td>
</tr>
<tr>
<td></td>
<td>B4E2F6DE072BFC51BB</td>
<td></td>
<td>This is very secret</td>
</tr>
</tbody>
</table>
Application Sandbox Contents:

✶ App-Generated Library files and preferences:

```xml
<key>Passcode</key>
<string>2255V3rryS3(R37!!</string>
<key>kAppiraterFirstUseDate</key>
```

Application Sandbox Contents:

**App-Generated Cookies:**

```plaintext
Position: 0 / 438 ( 0% )

00000000: 3F 6F 6F 6B 00 00 00 02 00 00 02 9B 00 00 01 37 00 00 01 00
00000014: 06 00 00 00 24 00 00 00 83 00 00 00 00 D1 00 00 00 37 01 00 00
00000028: CB 01 00 00 2D 02 00 00 00 00 00 00 00 5F 00 00 00 00 00 00 00
0000003C: 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000050: 4F 00 00 00 32 E1 F1 BC 41 2E 66 61 63 65 62 6F 00 00 00 46 00 00 00
00000064: 00 00 00 32 E1 F1 BC 41 2E 66 61 63 65 62 6F 00 00 00 46 00 00 00
00000078: 6D 00 00 32 E1 F1 BC 41 2E 66 61 63 65 62 6F 00 00 00 46 00 00 00
0000008C: 35 30 36 38 35 36 00 00 00 4B 00 00 00 00 00 00 00 00 00 00
000000A0: 00 00 38 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000B4: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000C8: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000DC: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000F4: 41 2E 66 61 63 65 62 6F 00 00 00 46 00 00 00 00 00 00 00 00
00000108: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000011C: 41 2E 66 61 63 65 62 6F 00 00 00 46 00 00 00 00 00 00 00 00
00000130: 41 2E 66 61 63 65 62 6F 00 00 00 46 00 00 00 00 00 00 00 00
00000144: 41 2E 66 61 63 65 62 6F 00 00 00 46 00 00 00 00 00 00 00 00
00000158: 41 2E 66 61 63 65 62 6F 00 00 00 46 00 00 00 00 00 00 00 00
0000017C: 41 2E 66 61 63 65 62 6F 00 00 00 46 00 00 00 00 00 00 00 00
00000190: 41 2E 66 61 63 65 62 6F 00 00 00 46 00 00 00 00 00 00 00 00
```

Value:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>cook</td>
<td>A</td>
</tr>
<tr>
<td>value</td>
<td>7</td>
</tr>
</tbody>
</table>

---

2 A.facebook.com
m_c_user / 100912119
506856 N
8 F J L
A 2
A.facebook.com datr
/ 6hbVwobpKuwlK6g30
6VzoGx
A 2
A.facebook.com fr
/ 0fgNIDw0n5Uy9W2A5.
```
Application Sandbox Contents:

- App-Generated Documents
- App-Generated Library files and preferences
- App-Generated Cookies
- App-Generated Temporary files

*Highly Sensitive information Must not be accessed by Malware !!!*
iOS Malware
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 Requirements:

- All code must be signed.
- All apps will be subjected to a review.
- All certificates are required identity validation.
- All information is validated.
- Any misbehaving developers will be blacklisted.
- Every installation on the device is verified.

If you *misbehave*:

Apple’s decision to ban us and all the frauds coming up nicely exposes how defenseless their AppReview is against actual bad people.
Malware Distribution Tracks – App Store

- ZergHelper (Claud Xiao, paloalto networks)
- xCodeGohst (Claud Xiao, paloalto networks)
Malware Distribution Tracks – Distributor

- Yispecter, WireLurker (Claud Xiao, paloalto networks)
- masque-attack (Hui Xue, Tao Wei, Yulong Zhang, FireEye)
Malware Distribution Tracks – ScamWare
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:

Malware

Malware

GOTCHA

GO TO JAIL
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
## Capabilities Available to Developers

<table>
<thead>
<tr>
<th></th>
<th>Sign in with Apple ID</th>
<th>Apple Developer Program members</th>
</tr>
</thead>
<tbody>
<tr>
<td>App Groups</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>Background Modes</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>Data Protection</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>HealthKit</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>HomeKit</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>Inter-App Audio</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>Keychain Sharing</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>Maps</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>Wireless Accessory Configuration</td>
<td>●</td>
<td>●</td>
</tr>
</tbody>
</table>

Malware Capabilities

- Pinpoint GPS Locationing – Abusage

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

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

* Address-Book Stealing

```swift
case CNAuthorizationStatus.Authorized :

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

“Kicksend” Would Like to Access Your Contacts

Don’t Allow

OK
Malware Capabilities

*X* EXIF data extraction (GPS...)

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

    // get all the metadata in the image
    NSDictionary *metadata = (__bridge NSDictionary *)CGImageSourceCopyProperties(source, NULL);
    // make the metadata dictionary mutable so we can add properties to it
    NSDictionary *mutableDictionary = [metadata mutableCopy];

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

Calendar Access

```swift
// 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
date: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:


iOS 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
su-A-cyder v0.9.2.1
An Home-Brewed iOS Malware PoC Generator
Created by Chilik Tamir (@coreDump) for BlackHatASIA 2016

It is heavily based on the great work done by the following (and many more):
  # Cydia & Theos tweaking system (@saurik & Dustin Howett)
  # libimobiledevice utilities (https://www.libimobiledevice.org)
  # Bishop-Fox, theos-jailed (https://github.com/BishopFox/theos-jailed)
  # Asger Hautop Drewsen, insert_dylib (https://github.com/Tyilo/insert_dylib)
  # Spaceship, an Apple development automation platform (https://fastlane.tools/)

LICENSE:
su-A-cyder, Theos (and by extension, Logos) are available under the provisions of the GNU
General Public License, version 3 (or later), available here:

Projects 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!

https://github.com/Mi3Security/su-a-cyder
Malware Creation

* Evil Client Demo

---

```bash
#!/usr/bin/env bash

# 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

# 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....

LastError:Evilskyper_0223001 cycript -127/0.0.1 115+48

$criteria
```

---

**DISCLAIMER:**

This tool is an education tool for demonstration PoC of iOS Malware, Only!

---

**DEMO**

---

07:10
iPhone Repair shops  iPwn shops

DEVICE + PASSCODE FTW !!!
Hmm,
Can an Evil Client Access the Sandbox “Juicy” Content?
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
SandJack on iOS < 8.3

- Install Evil Client Overwriting the original Application
- Brute-force
- Profit $$$
Alas...

- Apple had patched this vulnerability on any iOS > 8.3
- Installation process will deny upgrade of any application
Hmm, it seems that Apple had patched the front door...

But apparently they left a backdoor wide open ..!
Introducing SandJack 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-Cyder: Home-brewing iOS malware like a Bo$$ BHAsia 2016

✱ Claud Xiao, Palo-Alto Networks,
http://researchcenter.paloaltonetworks.com/author/claud-xiao/
SandJacking: Profiting from iOS Malware

Chilik Tamir
chilik@mi3security.com
Twitter: @_coreDump