

# Pentesting iOS apps without jailbreak



Wojciech Reguła [Follow](#)

Jul 4, 2018 · 4 min read



This is a write-up that summarizes a practical part of the presentation that I gave on AppSec EU 2018 in London.



In theory, the general concept of pentesting iOS app without jailbreak includes 5 points:

1. Downloading application package
2. Setting up the environment
3. Injecting custom dylib & modification of executable file
4. Repacking and signing the package

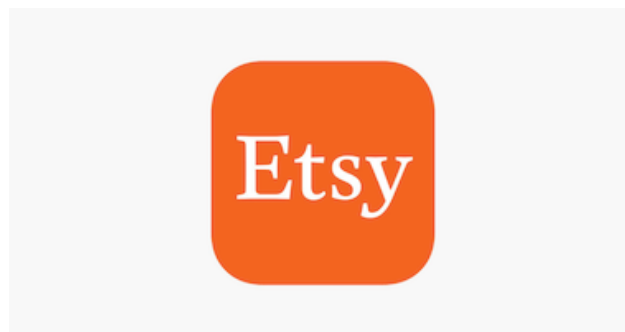
## 5. Installing the app on device in debug mode

### Downloading the application package

Every application package is encrypted using your iOS device's encryption key. This mechanism is called Apple FairPlay (<https://en.wikipedia.org/wiki/FairPlay>). So, if you don't have application package you won't be able to use this method — you still need a jailbroken iDevice. This is not a solution for bug bounty program attendees. Fortunately in the “pentests world”, the client usually delivers the application package 😊.

\*You can of course download IPA files from websites like <https://www.iphonecake.com/> but it's not recommended — apps may be infected with malware and usually are outdated.

In our example we will use Etsy application that is available on bug bounty program.



### Setting up the environment

At first you need to be registered as a developer. In this example we will be using free developer account, so the \$99 paid dev account is not needed. Then you have to take care about 2 things: signing certificate and embedded.mobileprovision file.





The easiest way to handle it, is just to create an empty XCode project.

Choose options for your new project:

Product Name:

Team:

Organization Name:

Organization Identifier:

Bundle Identifier:

Language:

☐ Use Core Data

☐ Include Unit Tests

☐ Include UI Tests

Press next and go to project's settings.

Tested Application > Halina's iPhone Finished running Tested Application on Halina's iPhone

Tested Application

Application

Tested Application

AppDelegate.swift

ViewController.swift

Main.storyboard

Assets.xcassets

LaunchScreen.storyboard

Info.plist

Products

General

Capabilities

Resource Tags

Info

Build Settings

Identity

Display Name:

Bundle Identifier:

Version:

Build:

Signing

☒ Automatically manage signing

Xcode will create and update profiles, app IDs, and certificates.

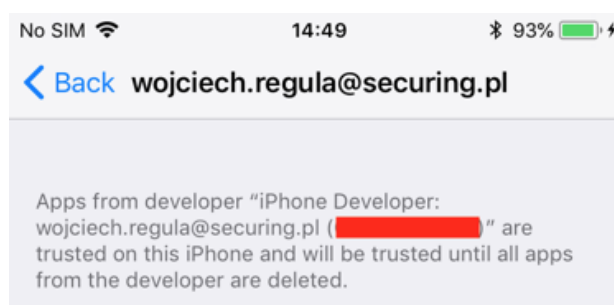
Team:

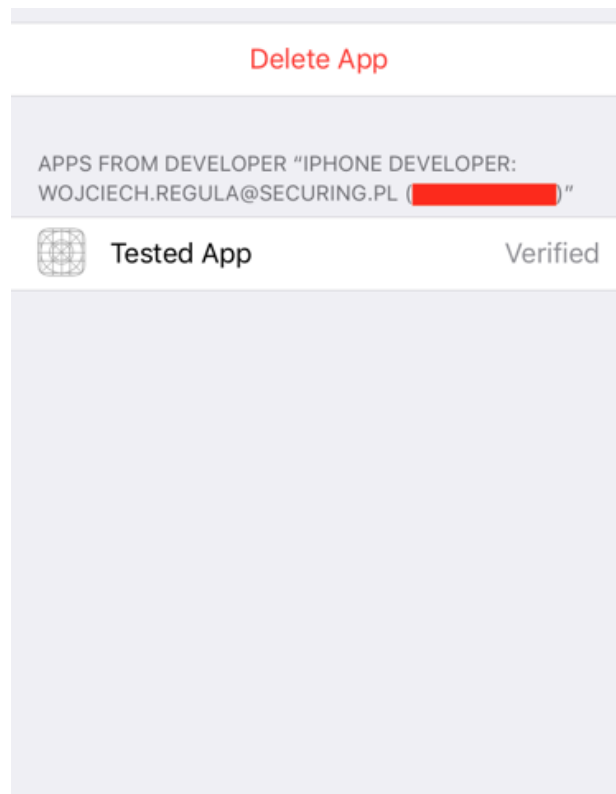
Provisioning Profile Xcode Managed Profile ⓘ

Plug in your iOS device, select it in XCode (in the top bar) and then select the “Automatically manage signing” checkbox. XCode will do all the job for you. Then, press build and run button and wait for the application to be installed.



If you are using free developer account, you also have to trust the application in your iDevice's Settings.





## Automating point 3 and 4 (injecting dylib, repacking and signing ipa)

Mentioned 3 and 4 point was described in details in my presentation. If you want to dive into those — check OWASP's recording.

The most simple way how to do it is just opening terminal and type:

```
security find-identity -p codesigning -v
```

This will display all valid codesigning signatures for you. Then you need to have copy the ID and run objection with following parameters:

```
objection patchipa — source ./etsy.ipa — codesign-signature  
SIGNATUREHERE
```

The process should look like on the screen below:

```
Bombon:etsy wregula$ security find-identity -p codesigning -v
1) [redacted] "Mac Developer: wojciech.regula@securing.pl ([redacted])"
2) [redacted] "iPhone Developer: [redacted] ([redacted])" (CSSMERR_TP_CERT_REVOKED)
3) [redacted] "iPhone Developer: [redacted] ([redacted])"
4) 22 [redacted] "iPhone Developer: wojciech.regula@securing.pl ([redacted])"
4 valid identities found
Bombon:etsy wregula$ objection patchipa —source ./etsy.ipa —codesign-signature 22 [redacted]
Using latest Github gadget version: 11.0.3
[redacted]
```

```

No provision file specified, searching for one...
Found provision file /Users/wregula/Library/Developer/Xcode/DerivedData/Tested_Application-ffzffoepcmlydefgquamicfteoub/Build/Product
Found a valid provisioning profile
Working with app: Etsy.app
Bundle identifier is: com.etsy.etsyforios
Codesigning 14 .dylib's with signature 22 [REDACTED]
Code signing: libswiftCoreImage.dylib
Code signing: libswiftObjectiveC.dylib
Code signing: libswiftCore.dylib
Code signing: libswiftCoreGraphics.dylib
Code signing: libswiftUIKit.dylib
Code signing: libswiftMetal.dylib
Code signing: libswiftDispatch.dylib
Code signing: libswiftos.dylib
Code signing: libswiftCoreFoundation.dylib
Code signing: FridaGadget.dylib
Code signing: libswiftDarwin.dylib
Code signing: libswiftContacts.dylib
Code signing: libswiftQuartzCore.dylib
Code signing: libswiftFoundation.dylib
Creating new archive with patched contents...
Codesigning patched IPA...
Cannot find entitlements in binary. Using defaults

Copying final ipa from /var/folders/vk/xcbq28894k589sct2g1_fhqm0000gn/T/etsy-frida-codesigned.ipa to current directory...
Cleaning up temp files...
Bombon:etsy wregula$ █

```

## Installing the app on device in debug mode

The most convenient way to install the patched ipa is staying in the same terminal and typing:

```
unzip etsy-frida-codesigned.ipa
```

And then use ios-deploy tool:

```
ios-deploy — bundle ./Payload/Etsy.app -W -d
```

Below a recording of described process:



### Pentesting iOS apps without jailbreak - ios-deploy

from [Wojciech Reguła](#)

01:06

## What to do next?

Since you now have prepared the environment, you need to connect your favourite tool to the Frida's dylib.

On the presentation I used Passionfruit — kudos for (ChiChou and oleavr).



## Pentesting iOS apps without jailbreak - running passion-fruit

from [Wojciech Regula](#)

00:55



That's all guys! Now you can perform the pentest on unjailbroken iDevice!



. . .

This article is about technical challenges during security testing of iOS apps. If you are interested in more general mobile application security requirements and best practices, **check out our guide.**

. . .

You should also meet the **iOS Security Suite** — a Swift library which simplify the implementation of anti-tampering mechanism in your iOS application.

[iOS](#) [Penetration Testing](#) [Jailbreak](#) [App Security](#) [Cybersecurity](#)

### Discover Medium

Welcome to a place where words matter. On Medium, smart voices and original ideas take center stage - with no ads in sight. Watch

### Make Medium yours

Follow all the topics you care about, and we'll deliver the best stories for you to your homepage and inbox. Explore

### Become a member

Get unlimited access to the best stories on Medium — and support writers while you're at it. Just \$5/month. Upgrade

