

## InfoSec Write-ups

# Setting Up an iOS Pentesting Lab on a Non-Jailbroken iDevice



Abdullah Khawaja · 4 min read · Jan 31, 2024



--  
2



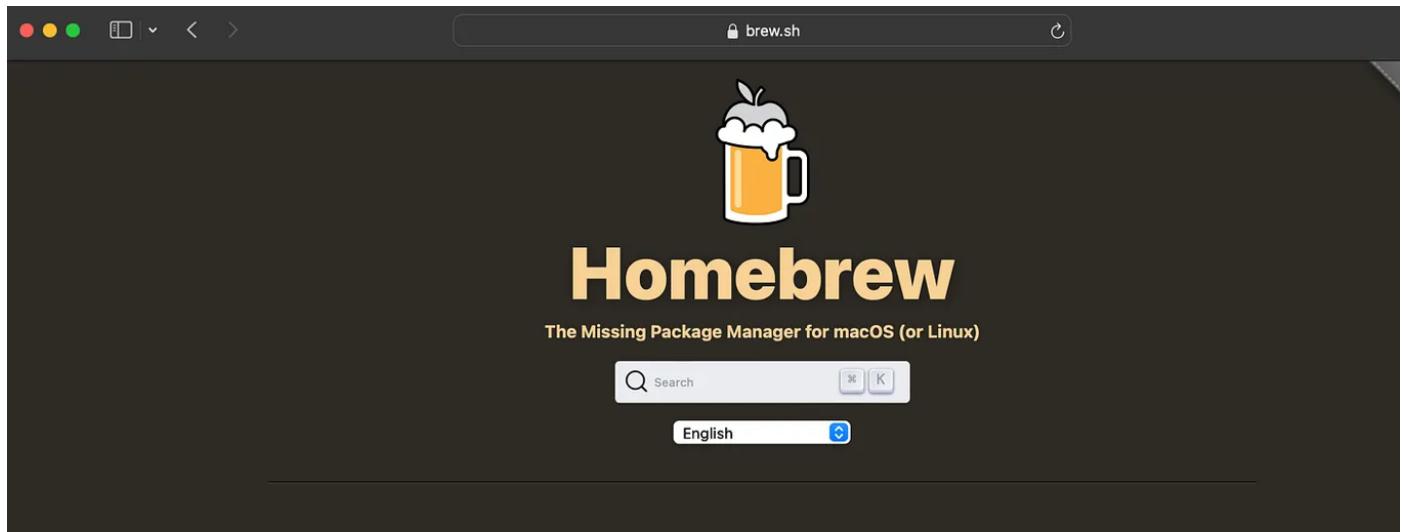
## Introduction:

Pentesting iOS apps can be tough when you can't jailbreak your iPhone or if there's no jailbreak available for your iOS version. This stops us from using powerful tools like Frida and Objection. I've been using a method for the past four years to patch iOS apps without jailbreaking my iPhone . These steps let us use FridaGadget easily, unlocking the full power of Frida and Objection for iOS testing. I'm sharing this method to help other pentesters facing the same issues.

## Prerequisites and Dependencies:

### 1. Install Homebrew

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/H
```



Medium



Search



Write

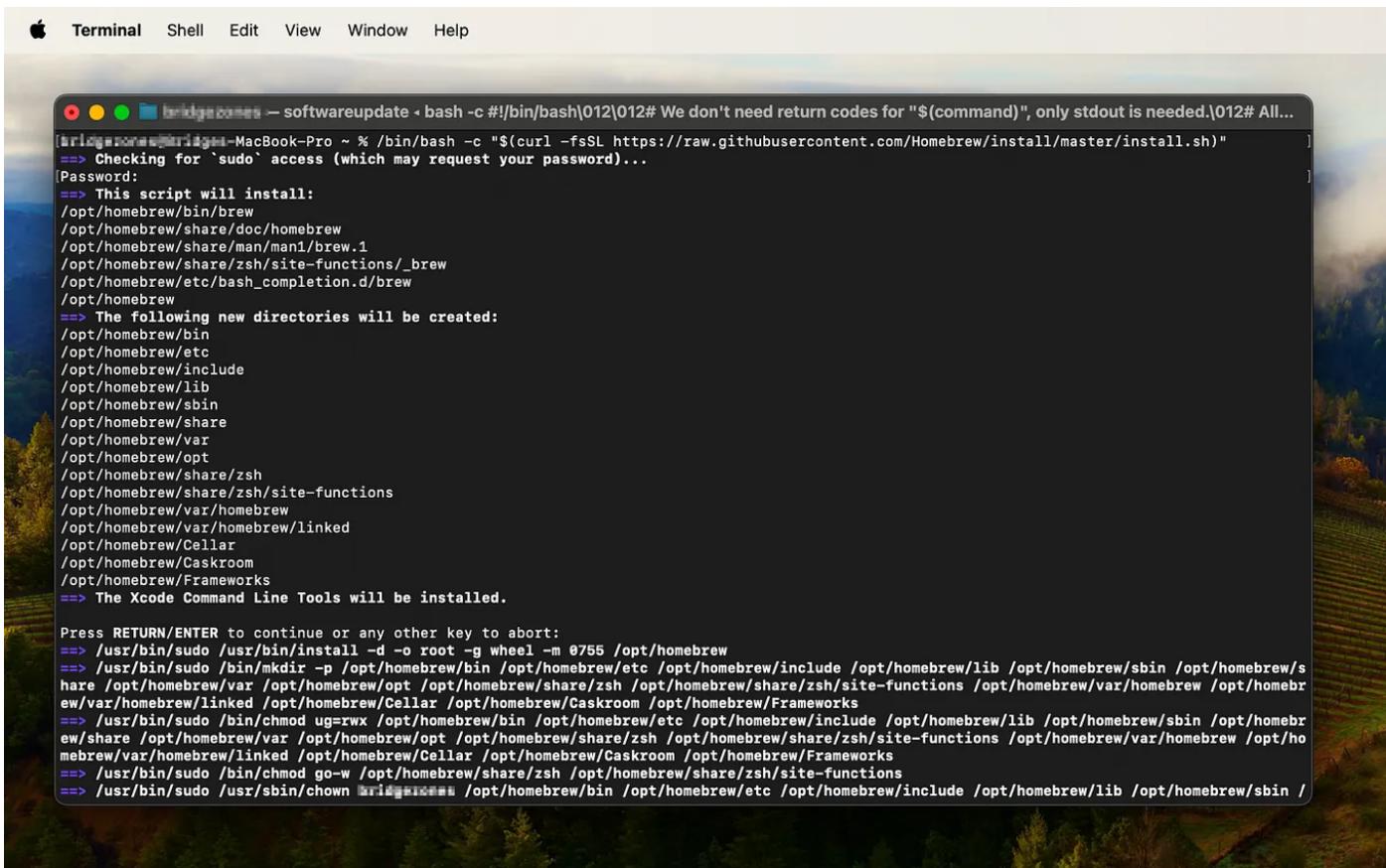
Sign  
up

Sign  
in



Paste that in a macOS Terminal or Linux shell prompt.

The script explains what it will do and then pauses before it does it. Read about other [Installation options](#).



A screenshot of a macOS Terminal window titled "bridgecomes — softwareupdate - bash -c #!/bin/bash|012|012# We don't need return codes for "\$(command)", only stdout is needed.|012# All...". The window shows the Homebrew installation process, starting with checking for sudo access, listing installed packages, creating new directories, installing Xcode Command Line Tools, and finally modifying system permissions. The background of the terminal window shows a scenic view of rolling hills and vineyards.

```
[bridgecomes@bridgecomes-MacBook-Pro ~ % /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install.sh)" ]  
==> Checking for `sudo` access (which may request your password)...  
[Password:  
==> This script will install:  
/opt/homebrew/bin/brew  
/opt/homebrew/share/doc/homebrew  
/opt/homebrew/share/man/man1/brew.1  
/opt/homebrew/share/zsh/site-functions/_brew  
/opt/homebrew/etc/bash_completion.d/brew  
/opt/homebrew  
==> The following new directories will be created:  
/opt/homebrew/bin  
/opt/homebrew/etc  
/opt/homebrew/include  
/opt/homebrew/lib  
/opt/homebrew/sbin  
/opt/homebrew/share  
/opt/homebrew/var  
/opt/homebrew/opt  
/opt/homebrew/share/zsh  
/opt/homebrew/share/zsh/site-functions  
/opt/homebrew/var/homebrew  
/opt/homebrew/var/homebrew/linked  
/opt/homebrew/Cellar  
/opt/homebrew/Caskroom  
/opt/homebrew/Frameworks  
==> The Xcode Command Line Tools will be installed.  
Press RETURN/ENTER to continue or any other key to abort:  
==> /usr/bin/sudo /usr/bin/install -d -o root -g wheel -m 0755 /opt/homebrew  
==> /usr/bin/sudo /bin/mkdir -p /opt/homebrew/bin /opt/homebrew/etc /opt/homebrew/include /opt/homebrew/lib /opt/homebrew/sbin /opt/homebrew/share /opt/homebrew/var /opt/homebrew/opt /opt/homebrew/share/zsh /opt/homebrew/share/zsh/site-functions /opt/homebrew/var/homebrew /opt/homebrew/Cellar /opt/homebrew/Caskroom /opt/homebrew/Frameworks  
==> /usr/bin/sudo /bin/chmod ug=rwx /opt/homebrew/bin /opt/homebrew/etc /opt/homebrew/include /opt/homebrew/lib /opt/homebrew/sbin /opt/homebrew/share /opt/homebrew/var /opt/homebrew/opt /opt/homebrew/share/zsh /opt/homebrew/share/zsh/site-functions /opt/homebrew/var/homebrew /opt/homebrew/Cellar /opt/homebrew/Caskroom /opt/homebrew/Frameworks  
==> /usr/bin/sudo /bin/chmod go-w /opt/homebrew/share/zsh /opt/homebrew/share/zsh/site-functions  
==> /usr/bin/sudo /usr/sbin/chown bridgecomes /opt/homebrew/bin /opt/homebrew/etc /opt/homebrew/include /opt/homebrew/lib /opt/homebrew/sbin /
```

## 2. Install NPM

```
brew install npm
```

```
[bridego@ridges-MacBook-Pro Pentest % brew install npm
==> Downloading https://ghcr.io/v2/homebrew/core/node/manifests/21.6.0
#####
==> Fetching dependencies for node: brotli, c-ares, icu4c, libnghttp2, libuv, ca-certificates and openssl@3
==> Downloading https://ghcr.io/v2/homebrew/core/brotli/manifests/1.1.0-1
#####
==> Fetching brotli
==> Downloading https://ghcr.io/v2/homebrew/core/brotli/blobs/sha256:2a95140d61198e3153ff27d8847b76dd34162f6e6e39f3e0f34d2b3a3e4f15dd
#####
==> Downloading https://ghcr.io/v2/homebrew/core/c-ares/manifests/1.25.0
#####
==> Fetching c-ares
==> Downloading https://ghcr.io/v2/homebrew/core/c-ares/blobs/sha256:00d1179348a7c14de25620f6924596b899228b731118e0f92d67168e9b952da9
#####
==> Downloading https://ghcr.io/v2/homebrew/core/icu4c/manifests/73.2
#####
==> Fetching icu4c
==> Downloading https://ghcr.io/v2/homebrew/core/icu4c/blobs/sha256:734c09d5285cf68f3c86c09522f6f24734bbe6d7338c4900d89092bcd9fb8fc0
#####
==> Downloading https://ghcr.io/v2/homebrew/core/libnghttp2/manifests/1.58.0
#####
==> Fetching libnghttp2
==> Downloading https://ghcr.io/v2/homebrew/core/libnghttp2/blobs/sha256:0aea68fb61161efd6c7f36fa64e48406a716e93cb2bc362ac443fd802914a6
#####
==> Downloading https://ghcr.io/v2/homebrew/core/libuv/manifests/1.47.0
#####
==> Fetching libuv
==> Downloading https://ghcr.io/v2/homebrew/core/libuv/blobs/sha256:8ec04556c7b8b6a08c0de32afa92d69d23b3d8e238eccf524a12e72532ce6
#####
==> Downloading https://ghcr.io/v2/homebrew/core/ca-certificates/manifests/2023-12-12
#####
==> Fetching ca-certificates
==> Downloading https://ghcr.io/v2/homebrew/core/ca-certificates/blobs/sha256:5c99ffd0861f01adc19cab495027024f7d890e42a9e7b689706b85c8e2b9c9b3
#####
==> Downloading https://ghcr.io/v2/homebrew/core/openssl@3/manifests/3.2.0_1
#####
100.0%
```

### 3. Install AppleSign

```
npm install -g applesign
```

```
[bridego@ridges-MacBook-Pro Pentest % npm install -g applesign
npm WARN skipping integrity check for git dependency ssh://git@github.com/TooTallNate/plist.js.git

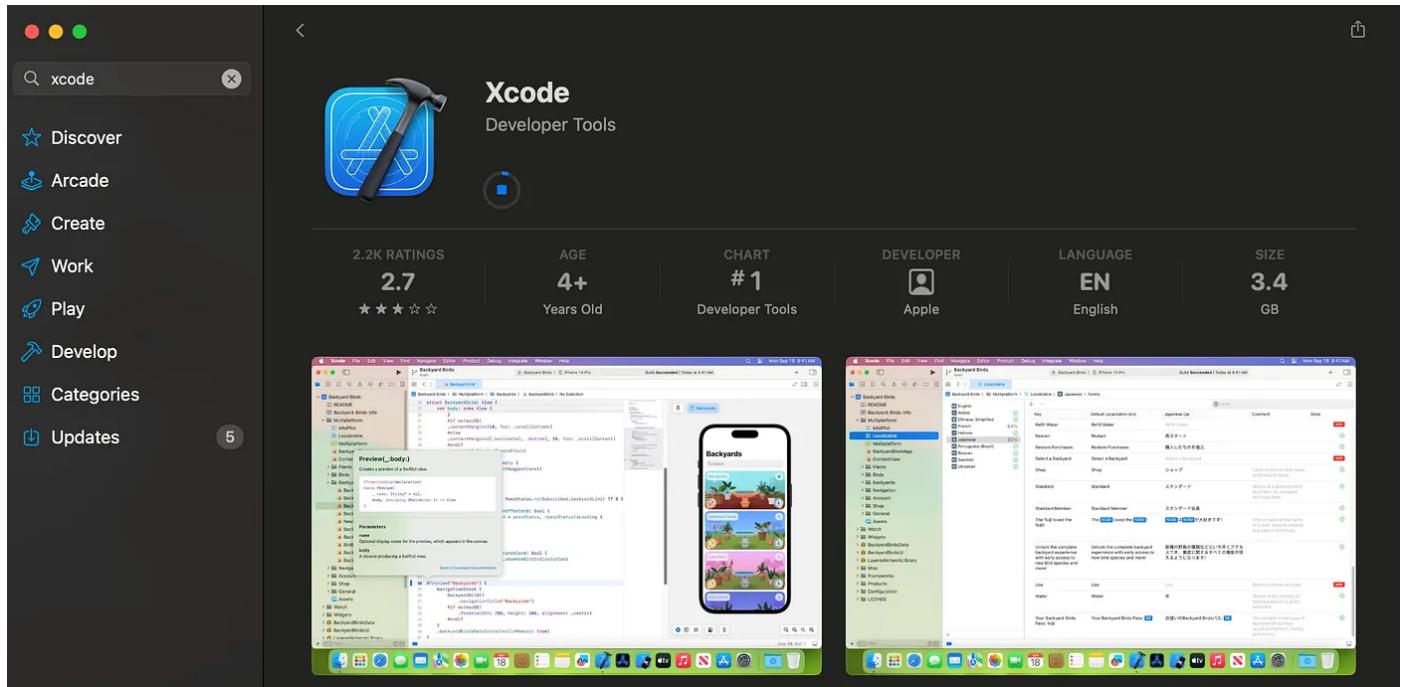
added 106 packages in 4s

23 packages are looking for funding
  run `npm fund` for details
npm notice New minor version of npm available! 10.2.4 -> 10.3.0
npm notice Changelog: https://github.com/npm/cli/releases/tag/v10.3.0
npm notice Run npm install -g npm@10.3.0 to update!
npm notice

[bridego@ridges-MacBook-Pro Pentest % ]
```

### 4. Install Xcode

First, install Xcode from the App Store and then install Xcode command line tools:



```
xcode-select --install
```



## 5. Build Insert\_dylib

## Get Abdullah Khawaja's stories in your inbox

Join Medium for free to get updates from this writer.

Enter your email

Subscribe

## Compile and install insert\_dylib from source:

```
git clone https://github.com/Tyilo/insert_dylib
cd insert_dylib
xcodebuild
cp build/Release/insert_dylib /usr/local/bin/insert_dylib
```

```
[brdige.com:~] MacBook-Pro % 
[brdige.com:~] MacBook-Pro % git clone https://github.com/Tyilo/insert_dylib
Cloning into 'insert_dylib'...
remote: Enumerating objects: 109, done.
remote: Total 109 (delta 0), reused 0 (delta 0), pack-reused 109
Receiving objects: 100% (109/109), 32.00 KiB | 414.00 KiB/s, done.
Resolving deltas: 100% (38/38), done.
[brdige.com:~] MacBook-Pro % 
[brdige.com:~] MacBook-Pro % cd insert_dylib
[brdige.com:~] MacBook-Pro insert_dylib %
[brdige.com:~] MacBook-Pro insert_dylib %
[brdige.com:~] MacBook-Pro insert_dylib % xcodebuild
Command line invocation:
/Applications/Xcode.app/Contents/Developer/usr/bin/xcodebuild

User defaults from command line:
IDEPackageSupportUseBuiltInSCM = YES

ComputeTargetDependencyGraph
warning: Building targets in manual order is deprecated - check "Parallelize build for command-line builds" in the project editor, or set
BLE_MANUAL_TARGET_ORDER_BUILD_WARNING in any of the targets in the current build to suppress this warning
note: Target dependency graph (1 target)
    Target 'insert_dylib' in project 'insert_dylib' (no dependencies)

GatherProvisioningInputs

CreateBuildDescription
Build description signature: 72a794113811e739136de289118b7e68
Build description path: /Users/[REDACTED]/Desktop/Pentest/insert_dylib/build/XCBuildData/72a794113811e739136de289118b7e68.xcbuilddata

CreateBuildDirectory /Users/[REDACTED]/Desktop/Pentest/insert_dylib/build
cd /Users/[REDACTED]/Desktop/Pentest/insert_dylib/insert_dylib.xcodeproj
builtin-create-build-directory /Users/[REDACTED]/Desktop/Pentest/insert_dylib/build
```

## 6. Install Objection

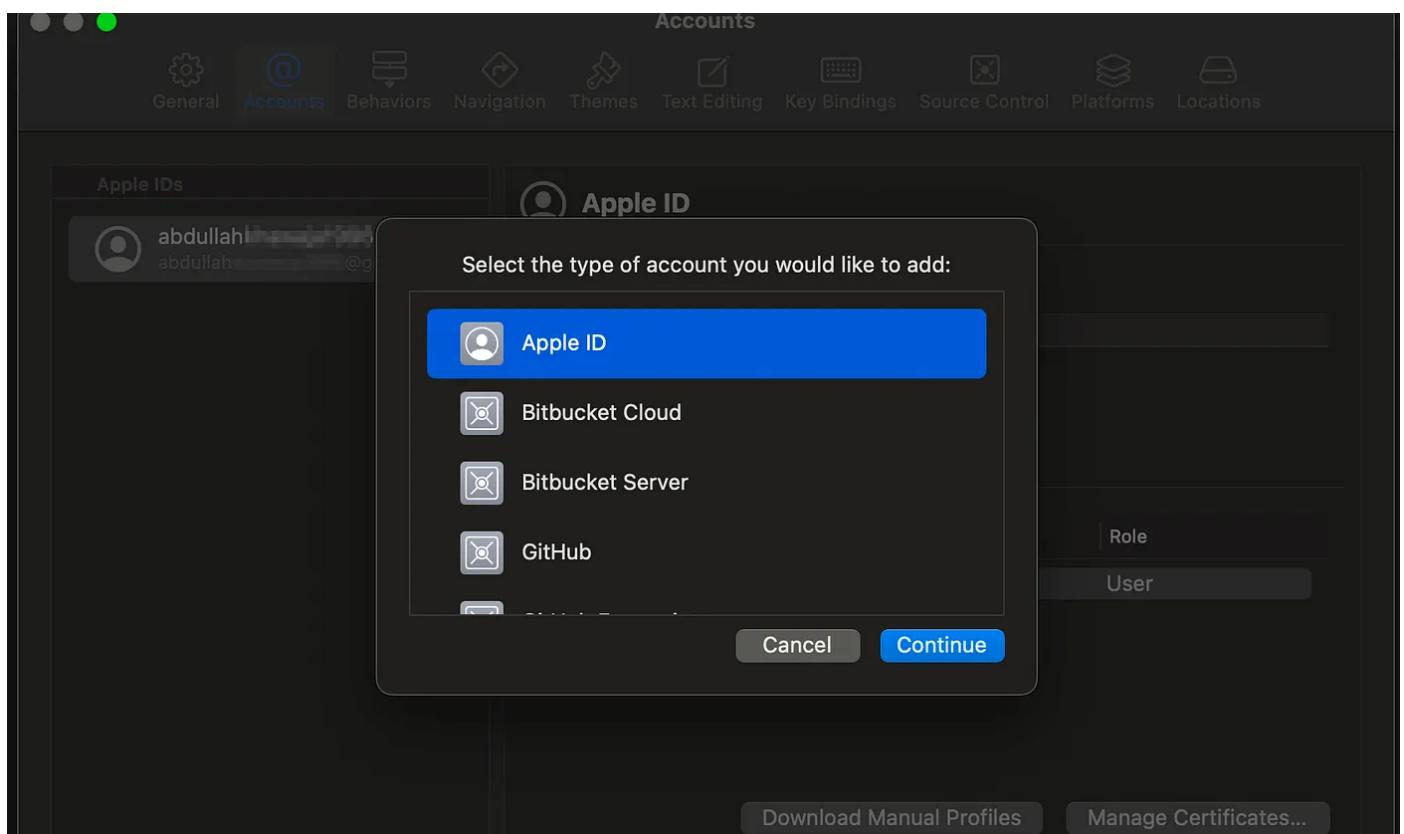
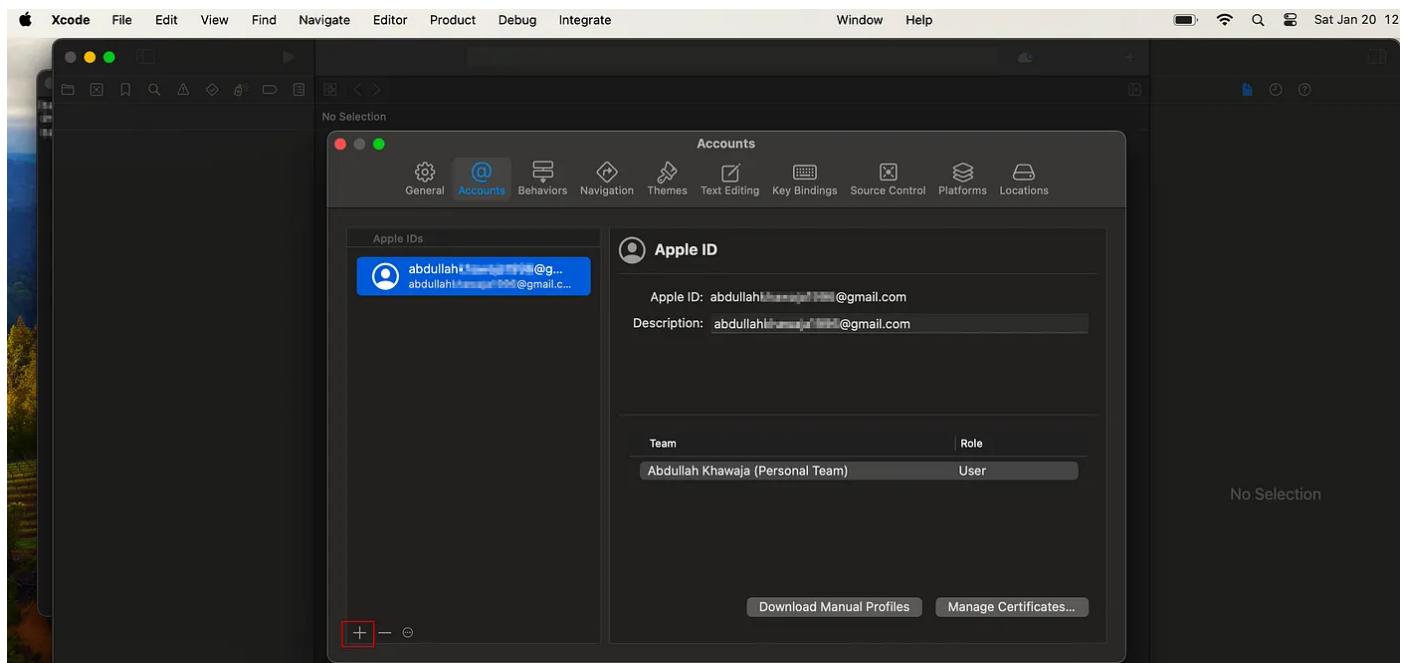
```
python3 -m pip install objection
```

```
MacBook-Pro Pentest % python3 -m pip install objection
Defaulting to user installation because normal site-packages is not writeable
Collecting objection
  Downloading objection-1.11.0.tar.gz (327 kB)
    |██████████| 327 kB 1.8 MB/s
Collecting frida>=14.0.0
  Downloading frida-16.1.11-cp37-abi3-macosx_11_0_arm64.whl (30.5 MB)
    |██████████| 30.5 MB 9.7 MB/s
Collecting frida-tools>=6.0.0
  Downloading frida-tools-12.3.0.tar.gz (200 kB)
    |██████████| 200 kB 8.3 MB/s
```

## Main Setup

### 1. Apple Developer Account

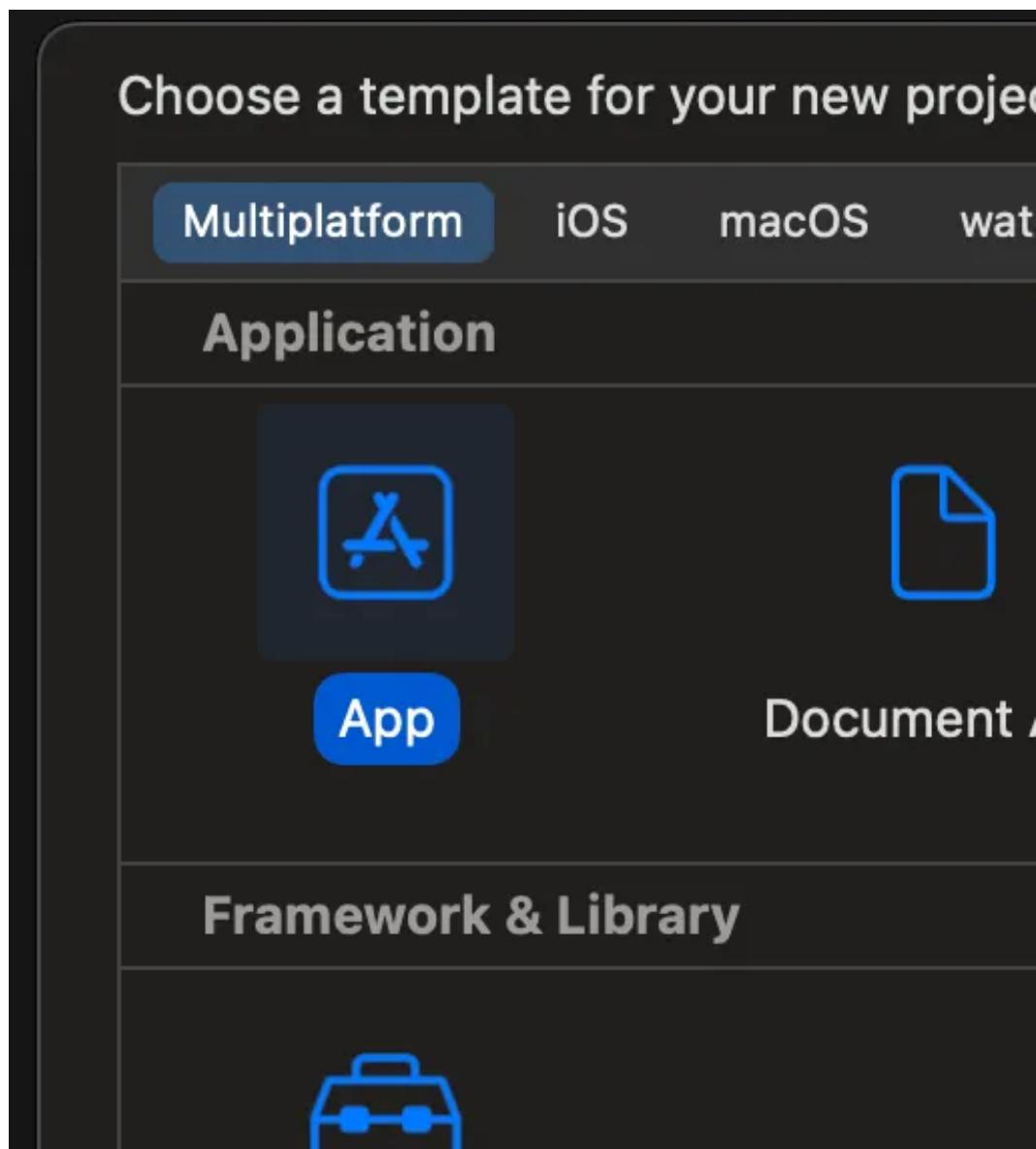
1. Open Xcode.
2. Navigate to Xcode > Preferences .
3. In the “Accounts” section, click the ‘+’ icon at the bottom left to add your Apple Developer account.

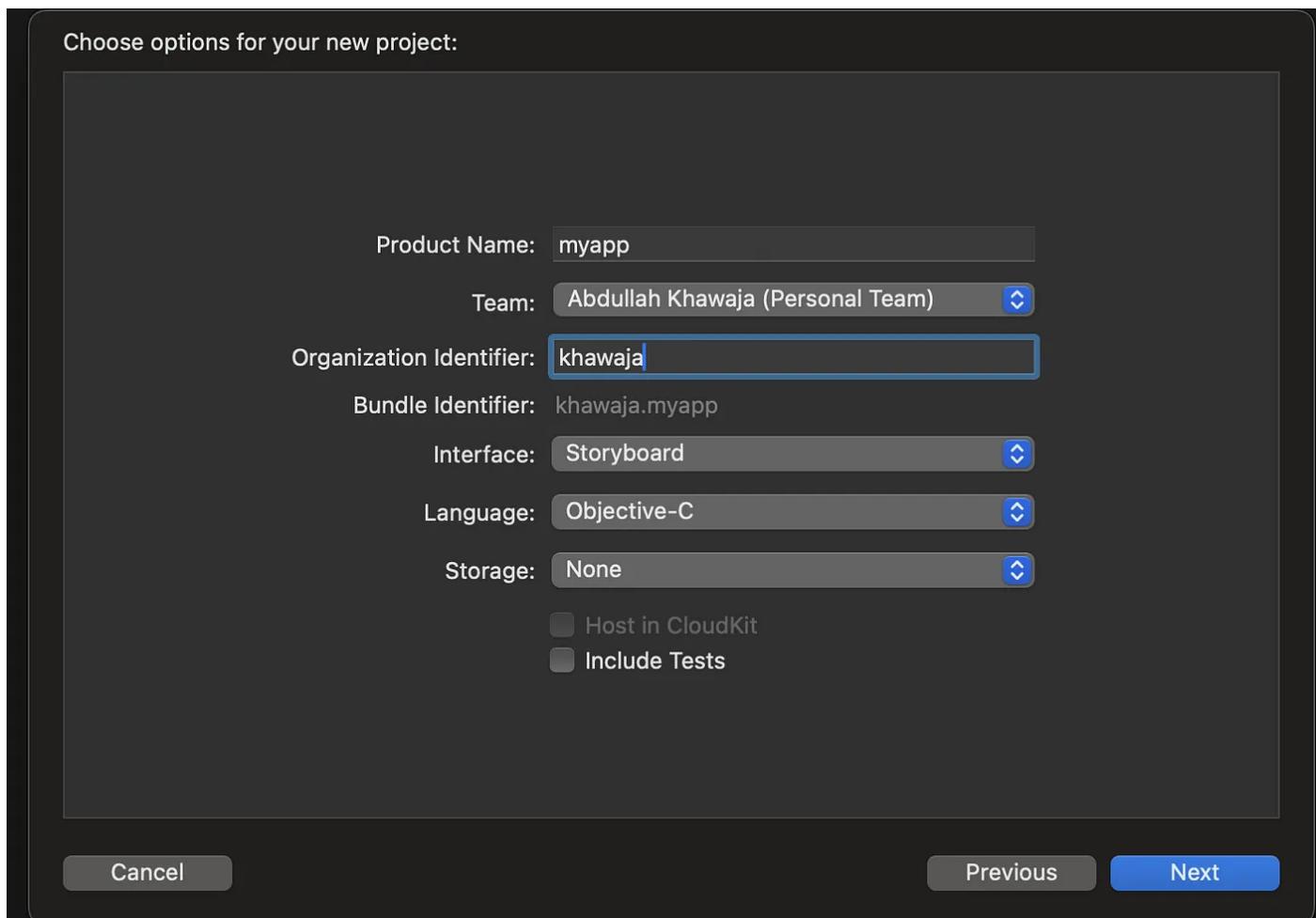


## 2. Create a Sample Application

1. Open Xcode and go to File > New > Project .
2. Select the “App” template.

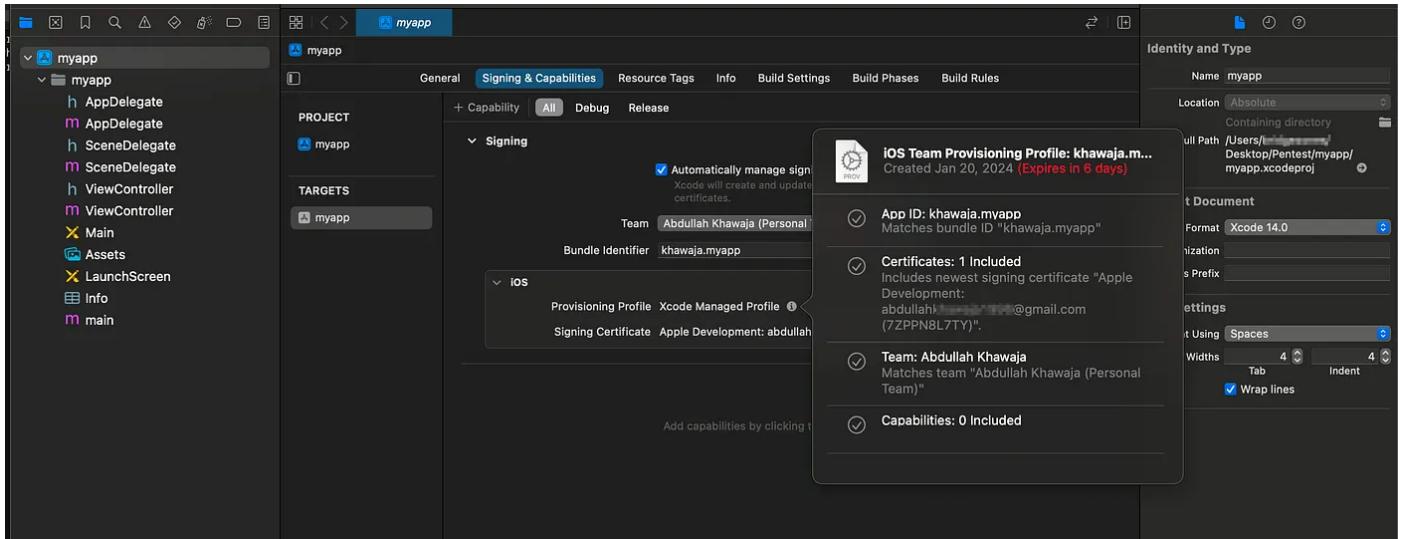
3. Once the project is created, navigate to `Signing & Capabilities`.
4. Verify that your developer account has been added, and you can see the signing certificate.





### 3. Check Provisioning Profile

1. Click on the information icon next to the developer account in Signing & Capabilities .
2. Review the properties of the provisioning profile.



## 4. Provisioning Profile Path

The path for the provisioning profile is `~/Library/MobileDevice/Provisioning Profiles`.

```
Pentest -- zsh -- 142x38
-MacBook-Pro Pentest % ls -larth ~/Library/MobileDevice/Provisioning\ Profiles
total 24
drwxr-xr-x  3 [REDACTED]  staff    96B 20 Jan 01:05 ..
-rw-r--r--  1 [REDACTED]  staff   12K 20 Jan 01:05 e0465ee9-[REDACTED].mobileprovision
drwxr-xr-x  3 [REDACTED]  staff    96B 20 Jan 01:05 .
-MacBook-Pro Pentest %
```

## 5. Check Signing Identity

Check your signing identity on this device using the command:

```
appleSign -L
```

```

MacBook-Pro ~ %
MacBook-Pro ~ %
MacBook-Pro ~ %
MacBook-Pro ~ % applesign -L
AD18F146XXXXXX Apple Development: abdullahkhawaja@gmail.com (7ZPPN8L7TY)
MacBook-Pro ~ %
MacBook-Pro ~ %
MacBook-Pro ~ %
MacBook-Pro ~ %

```

## 6. Patch the App

With all dependencies in place, patching an actual IPA is straightforward:

```
objection patchipa --source my-app.ipa --codesign-signature AD18F146xxxx -P e04
```

This command will extract the IPA, locate the app binary, patch it to load the FridaGadget.dylib, codesign the dylib and applications binary and repackage it for you.

```

MacBook-Pro Pентest %
MacBook-Pro Pентest %
MacBook-Pro Pентest %
MacBook-Pro Pентest % objection patchipa --source /var/folders/4/_mcy1vsin7850_x7gr0nm36dw0000gn/T/XXXXXX-frida.ipa.d2ceec0f-8b9a-4718-afa7-cd973b6790ed/Payload/myapp.app/PlugIns/ImportExtension.apex/importExtension'
Using latest Github gadget version: 16.1.11
Patcher will be using Gadget version: 16.1.11
Mobile provision bundle identifier is: khawaja.myapp
Working with app: /var/folders/4/_mcy1vsin7850_x7gr0nm36dw0000gn/T/XXXXXX-frida.ipa
Bundle identifier is: com.enchanted.snowball
Codesigning 27 .dylibs with signature AD18F146XXXXXX
Code signing: libswiftUIKit.dylib
Code signing: libswiftPhotos.dylib
Code signing: libswiftCore.dylib
Code signing: libswiftCoreImage.dylib
Code signing: libswiftObjectiveC.dylib
Code signing: libswiftCore.dylib
Code signing: libswiftCoreGraphics.dylib
Code signing: libswift_Concurrency.dylib
Code signing: libswiftUIKit.dylib
Code signing: libswiftMetal.dylib
Code signing: libswiftCoreData.dylib
Code signing: libswiftDispatch.dylib
Code signing: libswiftos.dylib
Code signing: libswiftCoreFoundation.dylib
Code signing: FridaGadget.dylib
Code signing: libswiftDarwin.dylib
Code signing: libswiftQuartzCore.dylib
Code signing: libswiftAssetsLibrary.dylib
Code signing: libswiftCoreAudio.dylib
Code signing: libswiftAVFoundation.dylib
Code signing: libswiftModelIO.dylib
Code signing: libswiftAccelerate.dylib
Code signing: libswiftFoundation.dylib
Code signing: libswiftCoreMedia.dylib
Code signing: libswiftCoreLocation.dylib
Code signing: libswiftGLKit.dylib
Code signing: libswiftsimd.dylib
Creating new archive with patched contents...
Codesigning patched IPA...
'/var/folders/4/_mcy1vsin7850_x7gr0nm36dw0000gn/T/XXXXXX-frida.ipa.d2ceec0f-8b9a-4718-afa7-cd973b6790ed/Payload/myapp.app/PlugIns/ImportExtension.apex/importExtension'

Copying final ipa from /var/folders/4/_mcy1vsin7850_x7gr0nm36dw0000gn/T/XXXXXX-frida-codesigned.ipa to current directory...
Cleaning up temp files...
MacBook-Pro Pентest %

```

Now, Objection will patch and sign the IPA file, generating a new IPA file named `xxx-frida-codesigned.ipa`.

## 7. Rename and Execute

1. Rename the newly generated IPA file and change its extension to .zip.
2. Unzip the file to access its contents.
3. Navigate to the Payload directory within the unzipped content.
4. Inside the Payload directory, you will see [bundle-ID].app, where [bundle-ID].app is the folder where your application lives.

```
cleaning up temp files...
MacBook-Pro Pentest %
MacBook-Pro Pentest %
MacBook-Pro Pentest % cp [REDACTED]-frida-codesigned.ipa app.zip
[REDACTED] MacBook-Pro Pentest % unzip app.zip
Archive: app.zip
  creating: Payload/
  creating: Payload/[REDACTED].lt.app/
  inflating: Payload/[REDACTED].lt.app/real1.jpeg
  creating: Payload/[REDACTED].lt.app/de.lproj/
```

```
[REDACTED]-MacBook-Pro Payload %
[REDACTED]-MacBook-Pro Payload % pwd
/Users/[REDACTED]/Desktop/Pentest/Payload
[REDACTED] MacBook-Pro Payload % ls
[REDACTED].lt.app
```

Now, we will deploy this application onto the device using a utility called ios-deploy:

1. Connect your iPhone to your MacBook.
2. Execute the following command in the terminal:

```
ios-deploy --bundle [bundle-ID].app --debug -W
```

Note: If you encounter the error “error: process launch failed,” it indicates that your iPhone hasn’t trusted the developer certificate. To resolve this:

1. Go to General > Device Management on your iPhone.
2. Tap on “Trust Apple Development” certificate.
3. Reinstall the app using ios-deploy, and it should now launch without any issues.
4. You should see the lldb debugger pop up and print out a ‘success’ message.

The screenshot shows a terminal window on the left and an 'Apple Development' certificate trust dialog on the right.

**Terminal Output (lldb):**

```
[ 90%] SandboxingApplication
[ 95%] GeneratingApplicationMap
[100%] Installed [REDACTED]
----- Debug phase -----
Starting debug of bf55b2385a615191a68e32c72b72af3024cbaea1 (D111AP, [REDACTED])
...
[ 0%] Looking up developer disk image
[ 95%] Developer disk image mounted successfully
[100%] Connecting to remote debug server
(lldb) command source -s 0 '/tmp/DE4DD935-D6CA-46F9-8450-26072975E831/fruitstrap-lldb-prep'
Executing commands in '/tmp/DE4DD935-D6CA-46F9-8450-26072975E831/fruitstrap-lldb-prep-cm
(lldb) platform select remote-ios --sysroot
Platform: remote-ios
Connected: no
(lldb) run
error: process launch failed: Security
(lldb)
```

**Certificate Trust Dialog:**

Apple Development: [REDACTED]

Apps from developer "Apple Development": [REDACTED] (CHVLR2857K) are not trusted on this iPhone and will not run until the developer is trusted.

Trust "Apple Development": [REDACTED]

APP FROM DEVELOPER "APPLE DEVELOPMENT" CLOUDS

Trust "Apple Development": (CHVLR2857K) Apps on This iPhone

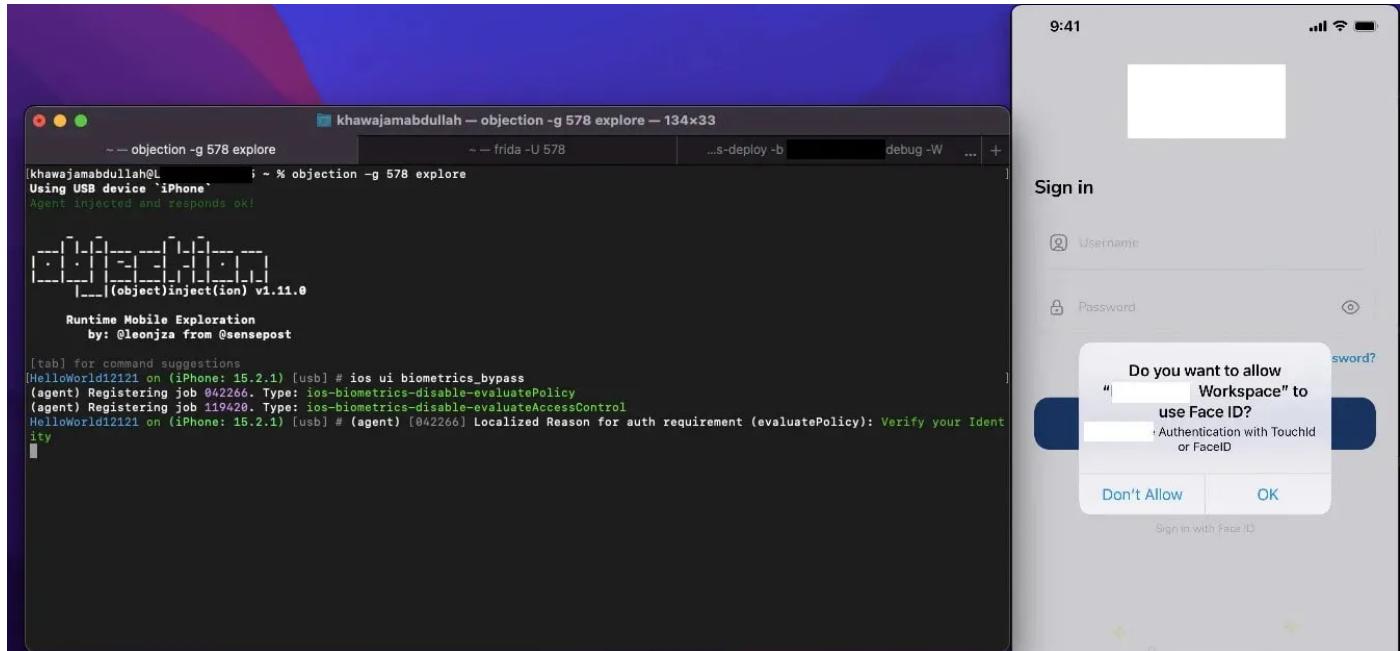
Trusting will allow any app from this developer to be used on your iPhone and may allow access to your data.

**Buttons:** Cancel (blue), Trust (red)

error: process launch failed

## App Successfully deployed & launched

Now, we can use tools like Frida or Objection to connect to this app.



## Objection in Action

## Conclusion:

Congratulations! You've set up a non-jailbroken iOS pentesting lab. Make sure to follow ethical guidelines and legal considerations during your testing. Happy testing!

Cybersecurity

Penetration Testing

Bug Bounty

Mobile Security

iOS Penetration Testing



### Published in InfoSec Write-ups

Follow

75K followers · Last published 1 day ago

A collection of write-ups from the best hackers in the world on topics ranging from bug bounties and CTFs to vulnhub machines, hardware challenges and real life encounters. Subscribe to our weekly newsletter for the coolest infosec updates: <https://weekly.infosecwriteups.com/>



### Written by Abdullah Khawaja

13 followers · 3 following

## Responses (2)



[See all responses](#)

---

[Help](#) [Status](#) [About](#) [Careers](#) [Press](#) [Blog](#) [Privacy](#) [Rules](#) [Terms](#) [Text to speech](#)