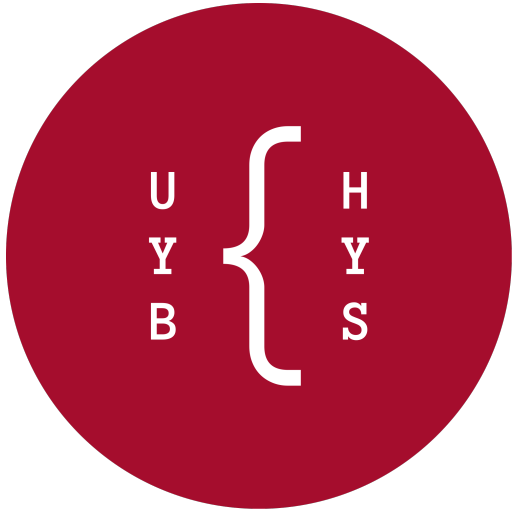
RANDORISEC

Mobile Hacking Workshop – iOS
UYBHYS 2020



Agenda

1. Basics
2. Testing environment
3. Static analysis – 2 labs
4. Data Security – 3 labs
5. Execution analysis – 2 labs
6. Transport Security – 1 lab

Legend

Lab: pratical exercice

OWASP M2 Insecure data storage

OWASP Top10 item covered by this lab

A macOS is needed for this lab

Only for macOS

Basics

Main steps of a security iOS application assessment

1. Review the codebase or reverse engineer the binary
2. Run the app on a jailbroken device
3. Inspect the app via instrumentation
4. Manipulate the runtime
5. MiTM all the network communications

OWASP Mobile Security Testing Guide

GitHub, Inc. [US] | https://github.com/OWASP/owasp-mstg

README.md

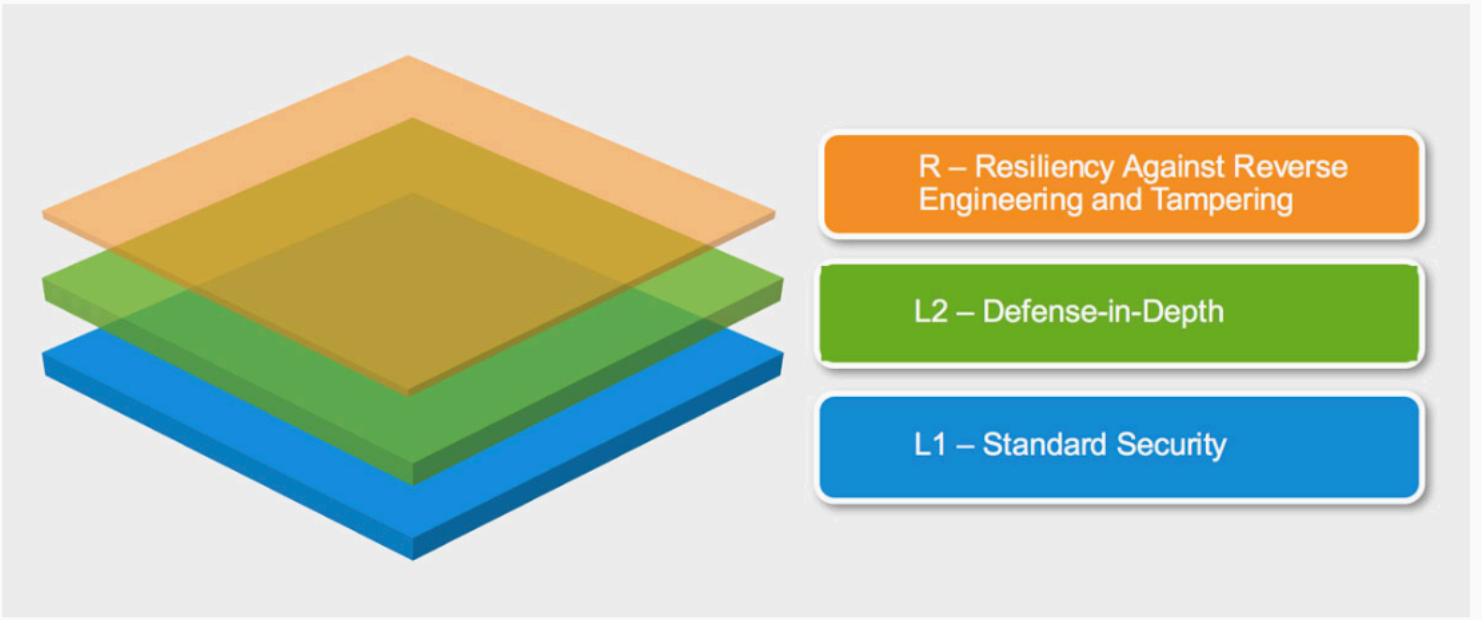
OWASP Mobile Security Testing Guide

Follow 542

owasp lab project build passing

This is the official GitHub Repository of the OWASP Mobile Security Testing Guide (MSTG). The MSTG is a comprehensive manual for mobile app security testing and reverse engineering. It describes technical processes for verifying the controls listed in the OWASP Mobile Application Verification Standard (MASVS). You can also read the MSTG on Gitbook or download it as an e-book.

OWASP Mobile AppSec Verification (standard)



Last version:
version 1.2 (march 2020)

OWASP Mobile AppSec Checklist

Mobile Application Security Requirements - iOS

ID	Detailed Verification Requirement	Level 1	Level 2	Status	Testing Procedure(s)
V1	**Architecture, design and threat modelling**				
1.1	All app components are identified and known to be needed.	✓	✓		-
1.2	Security controls are never enforced only on the client side, but on the respective remote endpoints.	✓	✓		-
1.3	A high-level architecture for the mobile app and all connected remote services has been defined and security has been addressed in that architecture.	✓	✓		-
1.4	Data considered sensitive in the context of the mobile app is clearly identified.	✓	✓		-
1.5	All app components are defined in terms of the business functions and/or security functions they provide.		✓	N/A	-
1.6	A threat model for the mobile app and the associated remote services has been produced that identifies potential threats and countermeasures.		✓	N/A	-
1.7	All security controls have a centralized implementation.		✓	N/A	-
1.8	There is an explicit policy for how cryptographic keys (if any) are managed, and the lifecycle of cryptographic keys is enforced. Ideally, follow a key management standard such as NIST SP 800-57.		✓	N/A	-
1.9	A mechanism for enforcing updates of the mobile app exists.		✓	N/A	-
1.10	Security is addressed within all parts of the software development lifecycle.		✓	N/A	-
V2	**Data Storage and Privacy**				
2.1	System credential storage facilities are used appropriately to store sensitive data, such as PII, user credentials or cryptographic keys.	✓	✓		Testing For Sensitive Data in Local Data Storage
2.2	No sensitive data should be stored outside of the app container or system credential storage facilities.				Testing For Sensitive Data in Local Data Storage
2.3	No sensitive data is written to application logs.	✓	✓		Testing For Sensitive Data in Logs
2.4	No sensitive data is shared with third parties unless it is a necessary part of the architecture.	✓	✓		Testing Whether Sensitive Data Is Sent To Third Parties
2.5	The keyboard cache is disabled on text inputs that process sensitive data.	✓	✓		Testing Whether the Keyboard Cache Is Disabled for Text Input Fields
2.6	No sensitive data is exposed via IPC mechanisms.	✓	✓		Testing Whether Sensitive Data Is Exposed via IPC Mechanisms
2.7	No sensitive data, such as passwords or pins, is exposed through the user interface.	✓	✓		Testing for Sensitive Data Disclosure Through the User Interface
2.8	No sensitive data is included in backups generated by the mobile operating system.		✓	N/A	Testing for Sensitive Data in Backups
2.9	The app removes sensitive data from views when backgrounded.		✓	N/A	Testing for Sensitive Information in Auto-Generated Screenshots
2.10	The app does not hold sensitive data in memory longer than necessary, and memory is cleared explicitly after use.		✓	N/A	Testing for Sensitive Data in Memory
2.11	The app enforces a minimum device-access-security policy, such as requiring the user to set a device passcode.		✓	N/A	Testing Local Authentication
2.12	The app educates the user about the types of personally identifiable information processed, as well as security best practices the user should follow in using the app.		✓	N/A	Testing user education
V3	**Cryptography**				
3.1	The app does not rely on symmetric cryptography with hardcoded keys as a sole method of encryption.	✓	✓		Verifying Key Management

OWASP - Top10 Mobile Risks

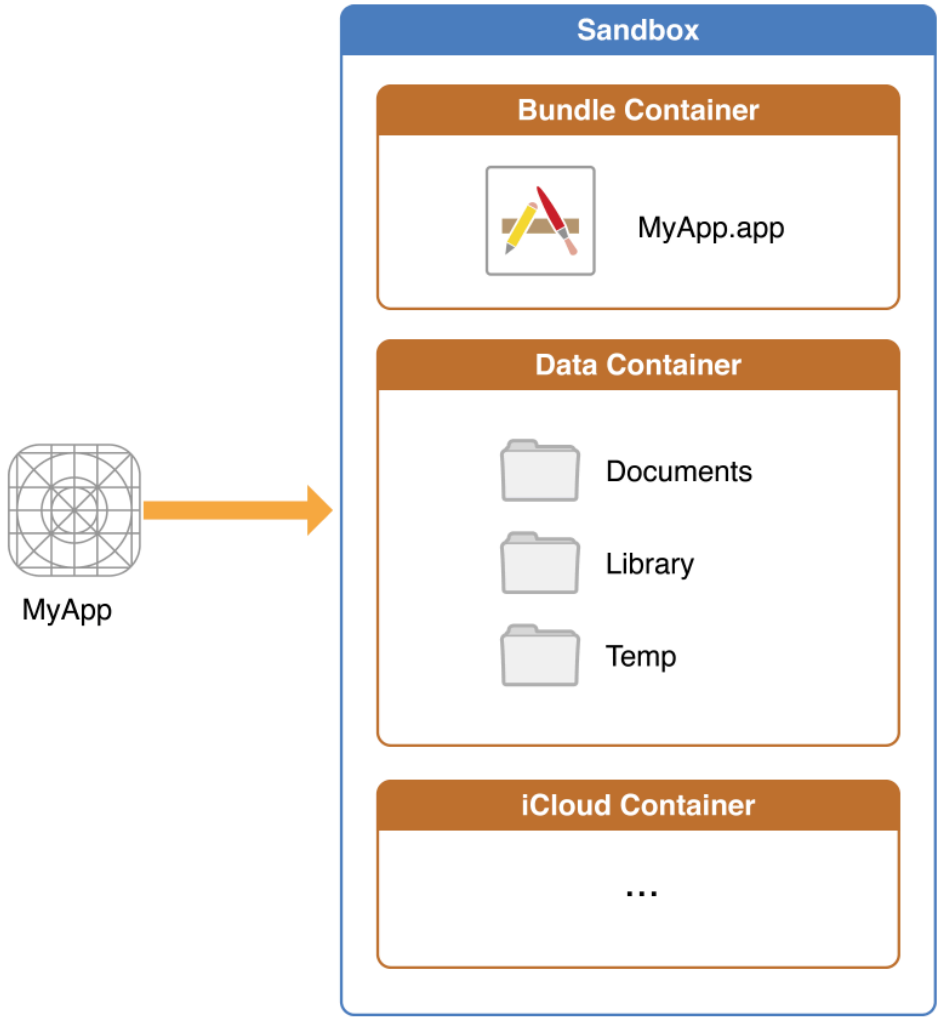
OWASP M1 Improper Platform Usage	OWASP M2 Insecure Data Storage	OWASP M3 Insecure Communication	OWASP M4 Insecure Authentication
OWASP M5 Insufficient Cryptography	OWASP M6 Insecure Authorization	OWASP M7 Client code quality	OWASP M8 Code Tampering
	OWASP M9 Reverse engineering	OWASP M10 Extraneous Functionality	

Filesystem: **System applications**

/Applications

```
iPhone:/Applications root# ls -al
total 0
drwxr-xr-x  76 root wheel 2432 May 31 06:20 ./
drwxr-xr-x  27 root wheel  864 May 31 06:18 ../
drwxrwxr-x  46 root admin 1472 May 21 20:47 AXUIViewService.app/
drwxrwxr-x   7 root admin  224 May 21 20:47 AccountAuthenticationDialog.app/
drwxrwxr-x   7 root admin  224 May 21 20:47 ActivityMessagesApp.app/
drwxrwxr-x   9 root admin  288 May 21 20:47 AdPlatformsDiagnostics.app/
drwxrwxr-x  52 root admin 1664 May 21 20:47 AppStore.app/
drwxrwxr-x  47 root admin 1504 May 21 20:47 AskPermissionUI.app/
drwxrwxr-x   7 root admin  224 May 21 20:47 BusinessExtensionsWrapper.app/
drwxrwxr-x  46 root admin 1472 May 21 20:47 CTCarrierSpaceAuth.app/
drwxrwxr-x  58 root admin 1856 May 21 20:47 Camera.app/
drwxrwxr-x  47 root admin 1504 May 21 20:47 CheckerBoard.app/
drwxrwxr-x  46 root admin 1472 May 21 20:47 CompassCalibrationViewService.app/
drwxrwxr-x   8 root admin  256 May 21 20:46 ContinuityCamera.app/
drwxrwxr-x  49 root admin 1568 May 21 20:46 CoreAuthUI.app/
drwxr-xr-x 124 root wheel 3968 May 31 01:37 Cydia.app/
drwxrwxr-x  46 root admin 1472 May 21 20:46 DDActionsService.app/
```

Filesystem: **User applications**



Filesystem: **User applications**

/private/var/containers/Bundle/Application

```
iPhone:/private/var/containers/Bundle/Application root# ls -al
total 0
drwxr-xr-x 29 _installd _installd 928 Jun 15 09:48 ./
drwxr-xr-x  6 _installd _installd 192 May 31 01:38 ../
drwxr-xr-x  5 _installd _installd 160 Jun 10 03:54 07B03D18-01A1-4F1E-A355-F000AC9B9F35/
drwxr-xr-x  5 _installd _installd 160 May 31 01:39 121F8420-4F80-4398-8C22-240CEFEF6F54/
drwxr-xr-x  5 _installd _installd 160 May 31 01:39 1E1ACAAC-CD00-47D2-BD0D-68A23A68A85A/
drwxr-xr-x  5 _installd _installd 160 May 31 02:19 277F19A6-D521-48ED-B75E-9D1E8FCA8C90/
drwxr-xr-x  5 _installd _installd 160 May 31 01:52 27FFB5C0-7872-4552-894D-D2374A0ACDD9/
drwxr-xr-x  5 _installd _installd 160 Jun 10 03:51 2E2449F1-AE2A-44A1-8A71-2FBF132852BD/
drwxr-xr-x  5 _installd _installd 160 May 31 01:38 43B016BC-5984-45A5-A4A5-B315E5046993/
drwxr-xr-x  5 _installd _installd 160 May 31 01:39 62784C25-81EB-4CCA-9BDE-C71AE162EA0A/
drwxr-xr-x  5 _installd _installd 160 May 31 01:39 65A879DB-CC9B-4AC6-8DB9-EFABC3DCDF90/
drwxr-xr-x  5 _installd _installd 160 May 31 01:39 7D73F79E-78CF-487B-8C9E-7A3B15CE13E0/
drwxr-xr-x  5 _installd _installd 160 May 31 01:38 81A56A73-B663-4566-8687-2EBE4C04ADD7/
```

Filesystem: User applications: **Bundle** Directory

/private/var/containers/Bundle/Application/UUID

```
iPhone:/private/var/containers/Bundle/Application/4E1CB17B-5A86-468D-AD57-F92C9F11A5B2/DamnVulnerableIOSApp.app root# ls -al
total 6456
drwxr-xr-x 38 _installd _installd    1216 Oct 26 18:49 ./
drwxr-xr-x  5 _installd _installd     160 Oct 26 18:49 ../
-rw-r--r--  1 _installd _installd   11553 Dec  2  2014 120x120.png
-rw-r--r--  1 _installd _installd   13907 Dec  2  2014 152x152.png
-rw-r--r--  1 _installd _installd    6525 Dec  2  2014 57x57.png
-rw-r--r--  1 _installd _installd  375699 Dec  2  2014 640_960_SplashScn.png
-rw-r--r--  1 _installd _installd  464522 Dec  2  2014 640x1136_SplashScn.png
-rw-r--r--  1 _installd _installd    7893 Dec  2  2014 72x72.png
-rw-r--r--  1 _installd _installd    8464 Dec  2  2014 76x76.png
-rw-r--r--  1 _installd _installd   11292 Dec  2  2014 AppIcon40x40\@2x.png
-rw-r--r--  1 _installd _installd   11553 Dec  2  2014 AppIcon60x60\@2x.png
drwxr-xr-x  3 _installd _installd      96 Oct 26 18:49 Base.lproj/
-rwxr-xr-x  1 _installd _installd 4483296 May 26 11:32 DamnVulnerableIOSApp*
-rw-r--r--  1 _installd _installd     423 May 26 11:32 DamnVulnerableIOSApp.entitlements
-rw-r--r--  1 _installd _installd    1410 Jan  1  1980 Info.plist
-rw-r--r--  1 _installd _installd  464522 Dec  2  2014 LaunchImage-700-568h\@2x.png
```

- Static files (signed):
- Binary
- Images
- Properties

Filesystem: User applications: **Data** Directory

/private/var/mobile/Containers/Data/Application/UUID

```
iPhone:/private/var/mobile/Containers/Data/Application/AA7D5264-12B6-4109-A397-C007299AB958 root# ls -al
total 4
drwxr-xr-x  7 mobile mobile  224 May 31 01:52 ./
drwxr-xr-x 81 mobile mobile 2592 Jun 15 09:48 ../
-rw-r--r--  1 root   mobile  211 May 31 01:52 .com.apple.mobile_container_manager.metadata.plist
drwxr-xr-x  3 mobile mobile   96 Jun 11 10:45 Documents/
drwxr-xr-x  5 mobile mobile  160 Jun 11 02:14 Library/
drwxr-xr-x  2 mobile mobile   64 May 31 01:52 SystemData/
drwxr-xr-x 25 mobile mobile  800 Jun 15 16:25 tmp/
iPhone:/private/var/mobile/Containers/Data/Application/AA7D5264-12B6-4109-A397-C007299AB958 root# du -ah
12K     ./Documents/credentials.sqlite
12K     ./Documents
4.0K    ./.com.apple.mobile_container_manager.metadata.plist
52K     ./Library/Caches/com.swaroop.iGoat/Cache.db
0       ./Library/Caches/com.swaroop.iGoat/Cache.db-wal
32K     ./Library/Caches/com.swaroop.iGoat/Cache.db-shm
84K     ./Library/Caches/com.swaroop.iGoat
40K     ./Library/Caches/Snapshots/com.swaroop.iGoat/600E684E-7B24-4386-947A-FDF646E30248@2x.atx
36K     ./Library/Caches/Snapshots/com.swaroop.iGoat/downscaled/D29866C1-7955-4271-B34A-5F63AF19BB65@2x.atx
36K     ./Library/Caches/Snapshots/com.swaroop.iGoat/downscaled
76K     ./Library/Caches/Snapshots/com.swaroop.iGoat
76K     ./Library/Caches/Snapshots
160K    ./Library/Caches
0       ./Library/Preferences
4.0K    ./Library/Cookies/com.swaroop.iGoat.binarycookies
```

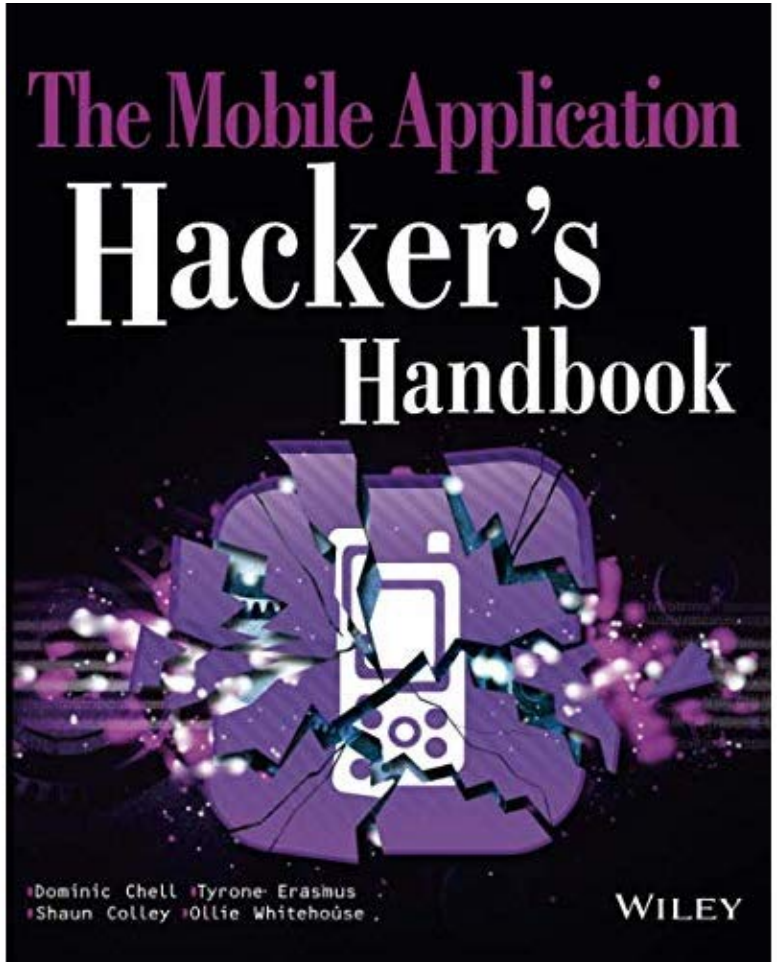
- User data
- Settings
- Cookies
- Cached files
- Temp files

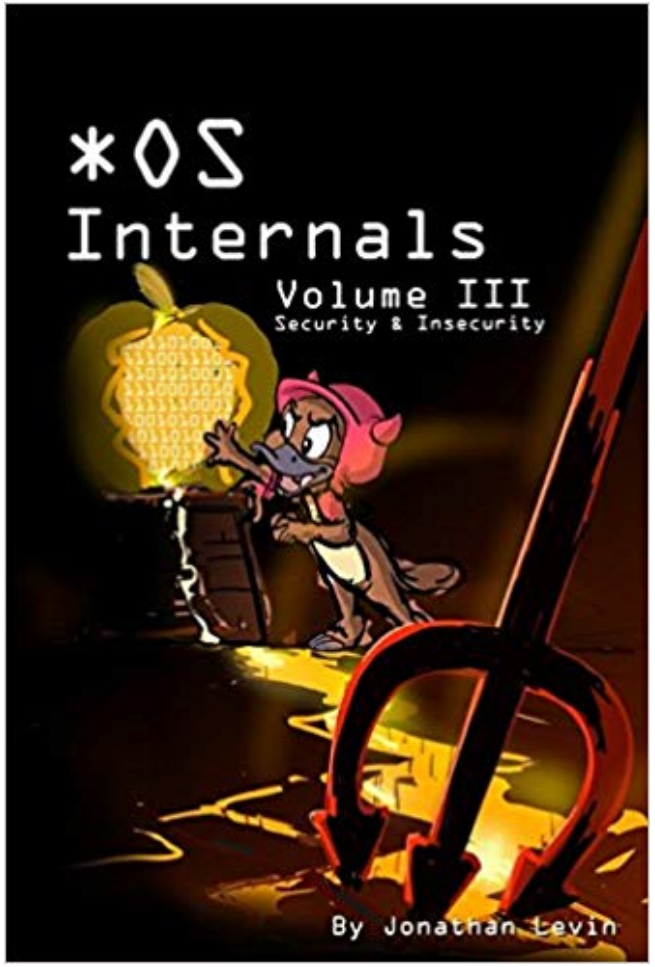
Info.plist

```
plistutil -i Info.plist
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
        <key>CFBundleName</key>
        <string>DamnVulnerableIOSApp</string>
        <key>DTSDKName</key>
        <string>iphoneos8.1</string>
        <key>DTXcode</key>
        <string>0610</string>
        <key>DTSDKBuild</key>
        <string>12B411</string>
        <key>CFBundleDevelopmentRegion</key>
        <string>en</string>
        <key>CFBundleVersion</key>
        <string>1.0</string>
        <key>BuildMachineOSBuild</key>
        <string>14B25</string>
        <key>DTPlatformName</key>
        <string>iphoneos</string>
        <key>CFBundleShortVersionString</key>
        <string>1.3</string>
```

To go further





Testing environnement

Setup – Intro

- For this workshop, we are going to use
  - A **VM**: Custom Kali **(login/pwd: kali/randorisec)**
    - With **tools**:
    - Cydia Impactor
    - Frida
    - Hopper
    - Objection
  - **Please use VMWare Workstation Player 16 or Fusion (DO NOT use Virtual Box)**

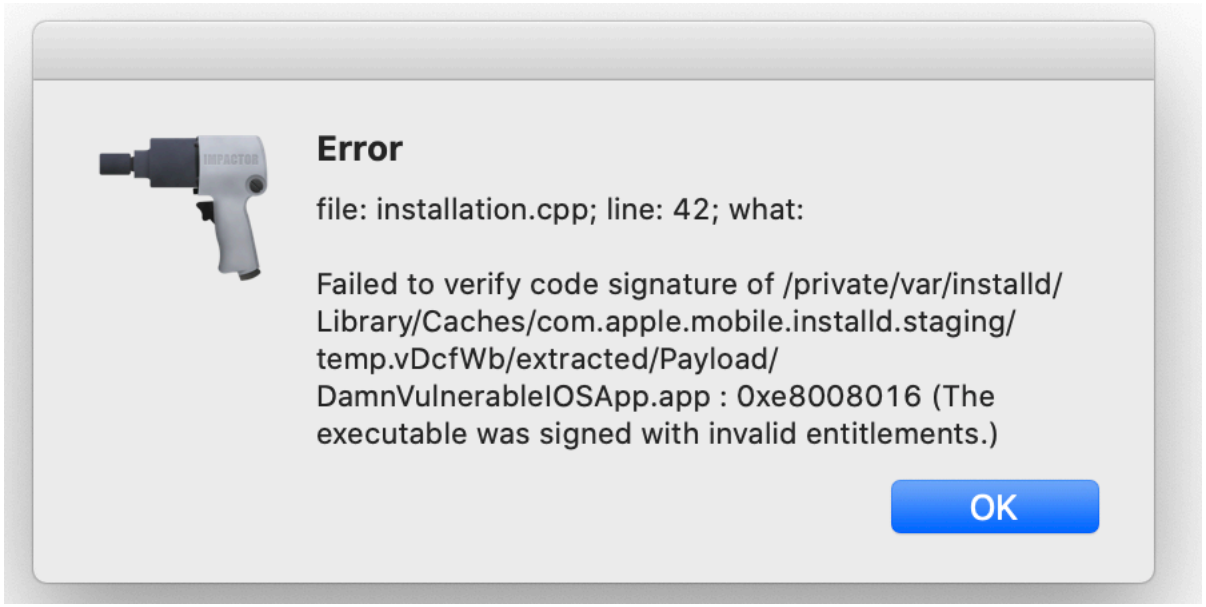
Vulnerable apps:

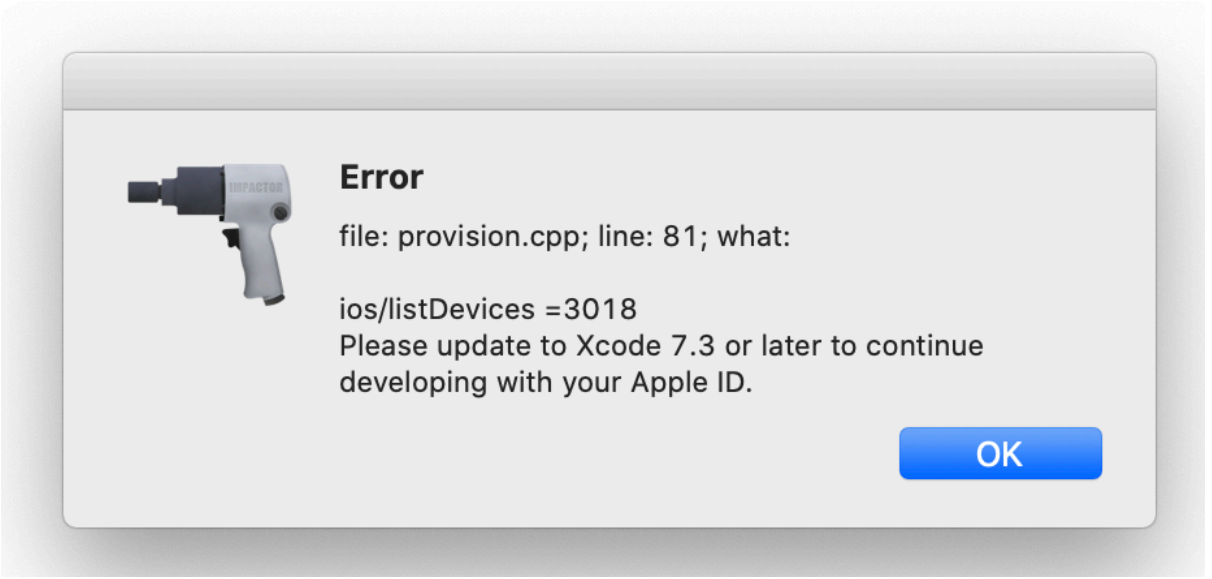
- Headbook
- DVIA (not DVIA-v2)
- iGoat (not iGoat-Swift)

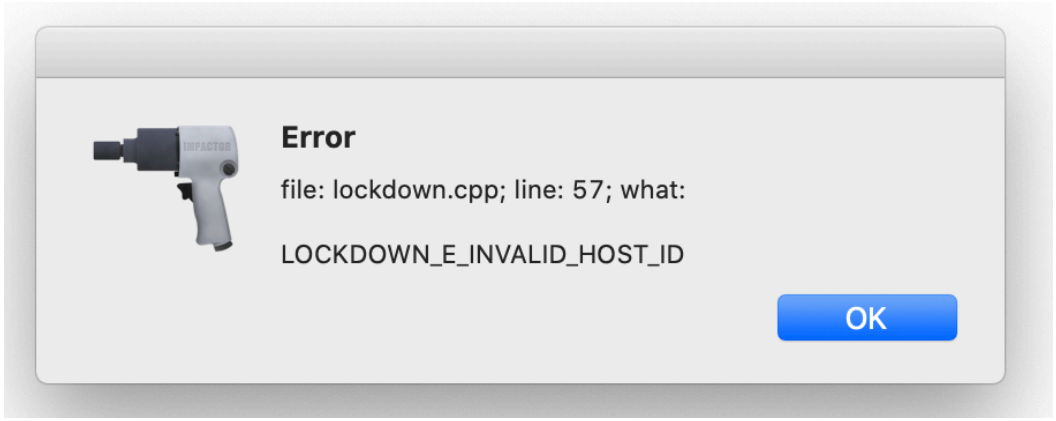
Setup – Vulnerable apps

- Vulnerable Apps
  - Install the following IPA on your jailbroken device using Cydia Impactor:
    - https://github.com/prateek147/DVIA/blob/master/DamnVulnerableiOSApp.ipa
    - https://github.com/OWASP/igoat/blob/develop/iGoat.ipa
  - No need to install the Headbook IPA

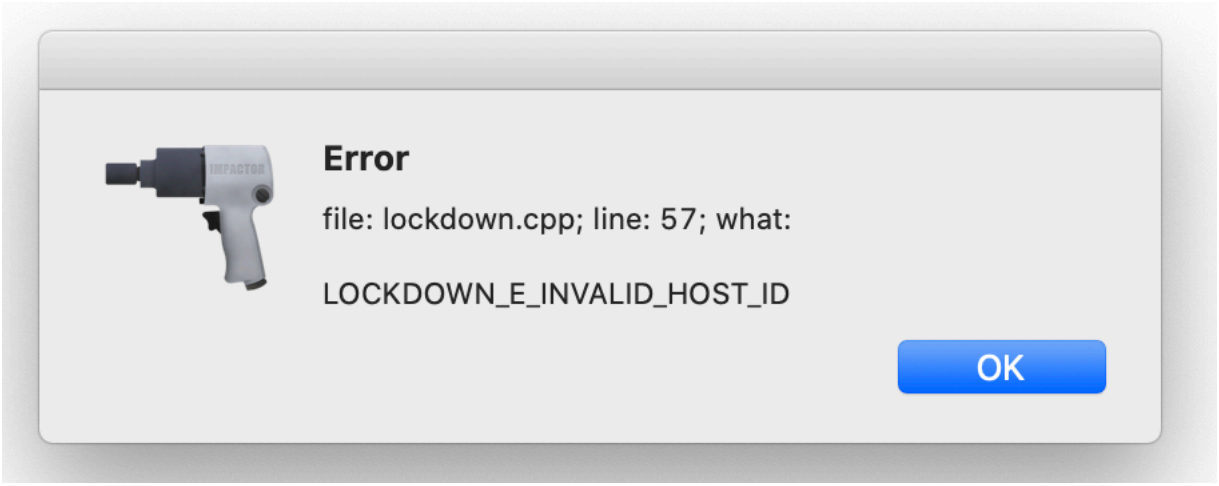
Signing problems





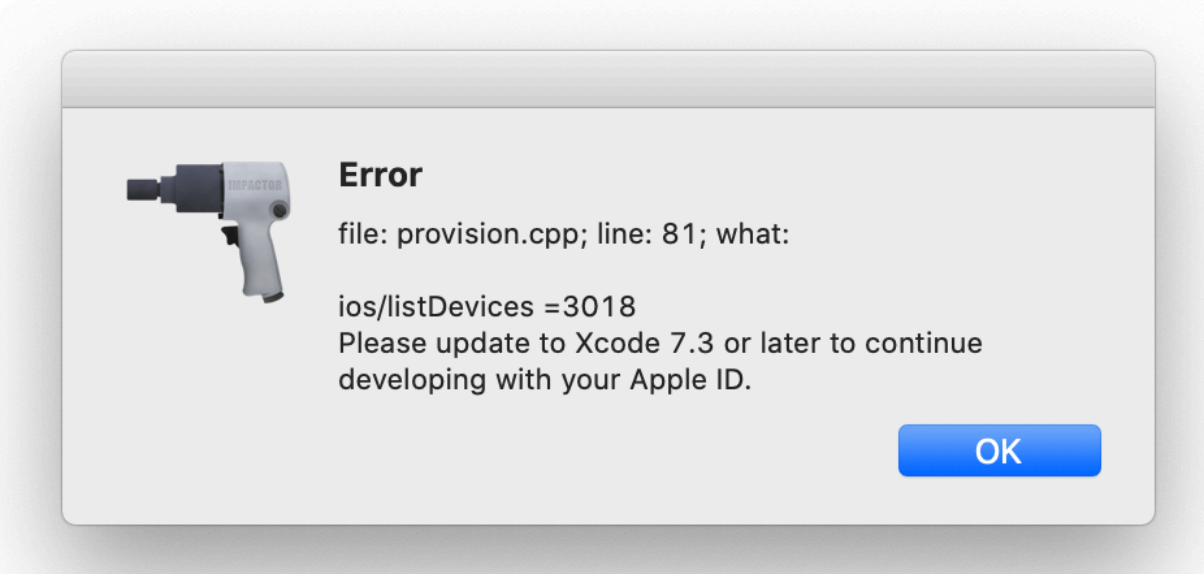


Signing problems: *file: lockdown.cpp; line: 57*



- Trust This Computer ?
- **Select « Trust »**

Signing problems: *file: provision.cpp; line: 81*



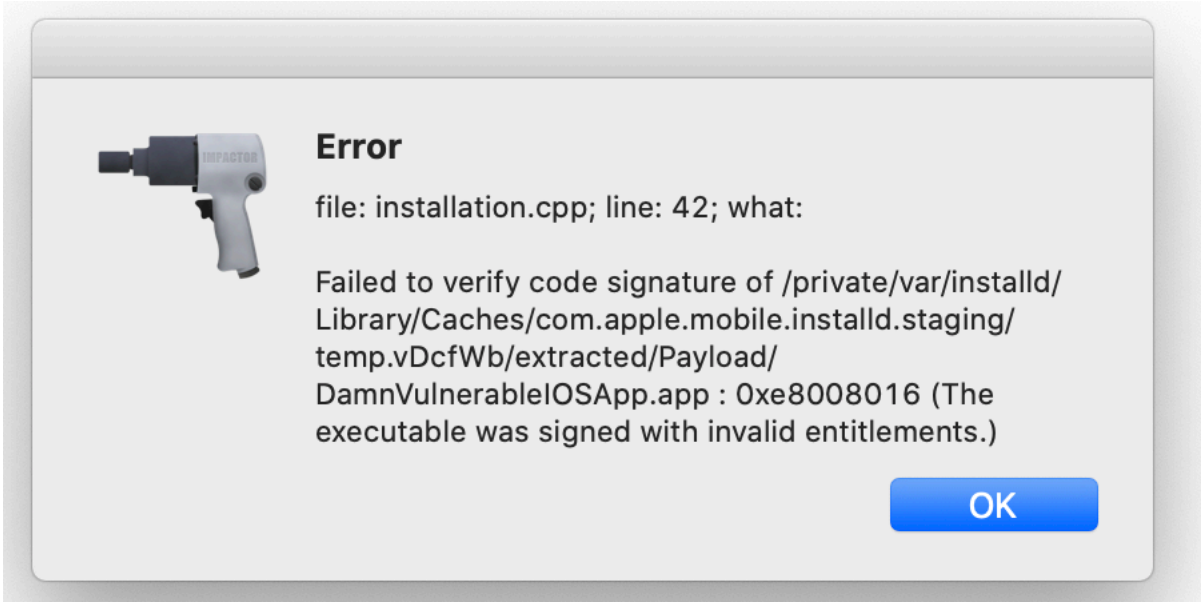
- Go to: https://developer.apple.com/programs/enroll/
- **Enroll on the Apple Developer Program**

Signing problems: *file: provision.cpp; line: 173*

- Go to: https://appleid.apple.com/account/manage
- Enable 2FA
- **Generate app-specific password**

Signing problems: *file: installation.cpp; line: 42*

Need a macOS



- Go to: https://developer.apple.com/
- Create a **provisioning profile**
- **Sign** your IPA with it

Signing problems: *file: installation.cpp; line: 42*

Need a macOS

Developer

Certificates, Identifiers & Profiles

‹ All Profiles

Register a New Provisioning Profile

Development

- **iOS App Development**
  Create a provisioning profile to install development apps on test devices.
- **tvOS App Development**
  Create a provisioning profile to install development apps on tvOS test devices.
- **macOS App Development**
  Create a provisioning profile to install development apps on test devices.

Distribution

- **Ad Hoc**
  Create a distribution provisioning profile to install your app on a limited number of registered devices.

Signing problems: *file: installation.cpp; line: 42*

Need a macOS

```
applesign -i <identity> -m <provision_profile> -o <signed.ipa> <original.ipa>
```

```
Unzipping /Users/davydouhine/Documents/iOS/ipa/dvia.ipa
Payload found
Main IPA executable is not encrypted
Embedding new mobileprovision
{"application-identifier":"T9T8LG2CVR.*","keychain-access-groups":["T9T8LG2CVR.*"],"get-task-allow":false,"com.apple.developer.team-identifier":"T9T8LG2CVR"}
Updated binary entitlements/Users/davydouhine/Documents/iOS/ipa/dvia.ipa.ae9b005e-61f9-4a59-84ce-67f61334c2c0/Payload/DamnVulnerableIOSApp.app/DamnVulnerableIOSApp.entitlements
Signing libraries and frameworks
Executable found at /Users/davydouhine/Documents/iOS/ipa/dvia.ipa.ae9b005e-61f9-4a59-84ce-67f61334c2c0/Payload/DamnVulnerableIOSApp.app/DamnVulnerableIOSApp
Resolving signing order using layered list
Signed /Users/davydouhine/Documents/iOS/ipa/dvia.ipa.ae9b005e-61f9-4a59-84ce-67f61334c2c0/Payload/DamnVulnerableIOSApp.app/DamnVulnerableIOSApp
Verifying /Users/davydouhine/Documents/iOS/ipa/dvia.ipa.ae9b005e-61f9-4a59-84ce-67f61334c2c0/Payload/DamnVulnerableIOSApp.app/DamnVulnerableIOSApp
Zipifying into /Users/davydouhine/Documents/iOS/ipa/dvia_signed.ipa ...
Cleaning up /Users/davydouhine/Documents/iOS/ipa/dvia.ipa.ae9b005e-61f9-4a59-84ce-67f61334c2c0
Target is now signed: dvia_signed.ipa
```

Materials

```
git clone https://github.com/randorisec/UYBHYS-Workshop
```

```
 49K May 27 23:10 Headbook-v1.0.ipa
281K Oct 22 17:32 Mobile_Hacking_iOS_cheatsheet_v0.1.pdf
5.5M May 27 23:10 dvia.ipa
8.8K May 27 23:10 frida.js
 11M May 27 23:10 iGoat.ipa
3.2K May 27 23:10 iOS_cheatsheet.txt
```

Static analysis

Static analysis

- Needs:
  - full source code !
  - tools (Checkmarx , Sonar, Clang Static Analyzer, …) !
- But
  - a simple “grep” can help
  - even **without the full source code a lot can be done** by **reverse engineering** the static files (binary, etc.)

OWASP M2 Insecure Data Storage

OWASP M3 Insecure Communication

OWASP M5 Insufficient Cryptography

OWASP M7 Client code quality

OWASP M10 Extraneous Functionality

Static analysis

To detect:

- Caching
- Cleartext credentials storage
- Back-end servers
- Bad pasteboard management
- SQL injection
- Code injection
- Verbose logging
- Vulnerable C function
- Custom URL schemes aka Deeplinks (e.g: cydia://)

OWASP M2 Insecure Data Storage

OWASP M3 Insecure Communication

OWASP M5 Insufficient Cryptography

OWASP M7 Client code quality

OWASP M10 Extraneous Functionality

Lab1: App metadata

OWASP M2 Insecure data storage

Goal: analyze Info.plist

Steps:

1. Unzip Headbook.ipa on your VM (you don't need to install it on your device)
2. Analyze Info.plist
3. Submit the flag to https://ctf.ivrodriguez.com

Lab2: Custom URL Scheme (Deep link)

OWASP M7
Client code
quality

Goal: find and use DVIA deep link

Steps:

1. Open DVIA
2. Go to “**Security Decisions via untrusted input**”
3. Click on “Start Challenge”

Lab2: Custom URL Scheme (Deep link)

OWASP M7 Client code quality

Goal: find and use DVIA deep link

Testing Custom URL Schemes (MSTG-PLATFORM-3)

Overview

Custom URL schemes allow apps to communicate via a custom protocol. An app must declare support for the schemes and handle incoming URLs that use those schemes.

Apple warns about the improper use of custom URL schemes in the Apple Developer Documentation:

> URL schemes offer a potential attack vector into your app, so make sure to validate all URL parameters and discard any malformed URLs. In addition, limit the available actions to those that do not risk the user's data. For example, do not allow other apps to directly delete content or access sensitive information about the user. When testing your URL-handling code, make sure your test cases include improperly formatted URLs.

Source: https://github.com/OWASP/owasp-mstg/blob/master/Document/0x06h-Testing-Platform-Interaction.md

Lab2: Custom URL Scheme (Deep link)

OWASP M7
Client code
quality

Goal: find and use DVIA deep link

Steps:

1. Analyze Info.plist

```
<key>CFBundleURLTypes</key>
<array>
        <dict>
                <key>CFBundleTypeRole</key>
                <string>None</string>
                <key>CFBundleURLSchemes</key>
                <array>
                        <string>dvia</string>
                </array>
        </dict>
</array>
```

Introducing Objection

```
davy@kali:~$ objection -g "DVIA" -N -h 192.168.1.25 explore
Using networked device @`192.168.1.25:27042`
Agent injected and responds ok!


     _   _         _   _
 ___| |_|_|___ ___| |_|_|___ ___
| . | . | | -_|  _|  _| | . |   |
|___|___| |___|___|_| |_|___|_|_|
      |___|(object)inject(ion) v1.9.2

     Runtime Mobile Exploration
        by: @leonjza from @sensepost

[tab] for command suggestions
com.highaltitudehacks.dvia on (iPhone: 13.6.1) [net] #
```

Introducing Objection

```
[tab] for command suggestions
com.highaltitudehacks.dvia on (iPhone: 13.6.1) [net] # env

Name              Path
----------------  ------------------------------------------------------------------------------------------
BundlePath        /private/var/containers/Bundle/Application/44EE06A7-3510-40B9-B17E-61741D2D7627/DamnVulnerableIOSApp.app
CachesDirectory   /var/mobile/Containers/Data/Application/C737E3BA-961A-4752-B322-05526770743A/Library/Caches
DocumentDirectory /var/mobile/Containers/Data/Application/C737E3BA-961A-4752-B322-05526770743A/Documents
LibraryDirectory  /var/mobile/Containers/Data/Application/C737E3BA-961A-4752-B322-05526770743A/Library
com.highaltitudehacks.dvia on (iPhone: 13.6.1) [net] # ios plist
                                                      pasteboard   Work with the iOS pasteboard
                                                      plist        Work with iOS Plists
                                                      sslpinning   Work with iOS SSL pinning
```

Introducing Objection

Steps:

1. Analyze Info.plist

To launch objection (on a device connected with USB):

```
objection -g "DVIA" explore
```

To launch objection (on a device connected with the network):

```
objection -g "DVIA" -N -h <IP> explore
```

To display Info.plist:

```
ios plist cat Info.plist
```

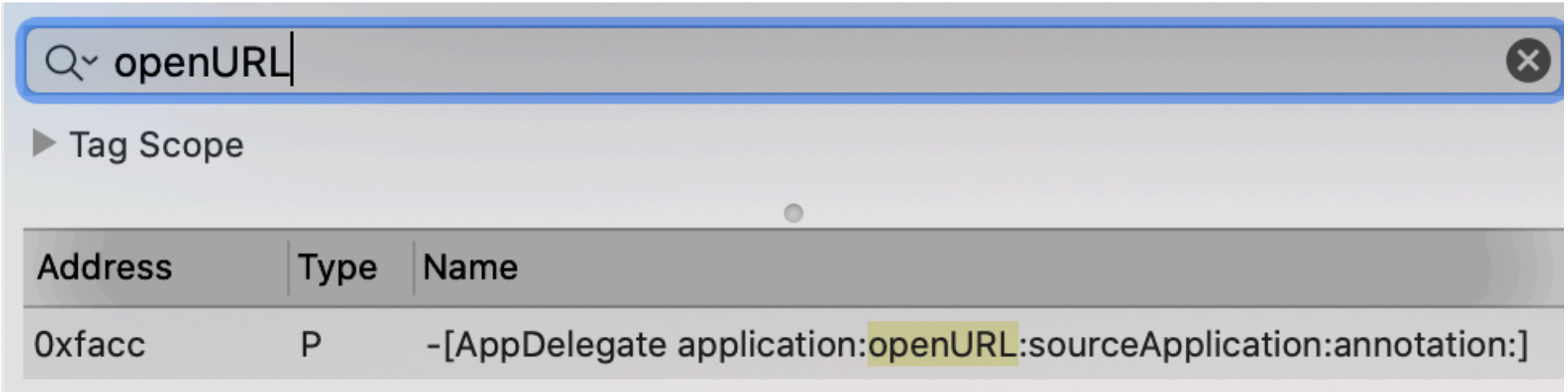
```
CFBundleURLTypes =     (
            {
        CFBundleTypeRole = None;
        CFBundleURLSchemes =                (
            dvia
        );
    }
);
```

Introducing Hopper

OWASP M7
Client code quality

Steps:

2. Analyze DVIA binary



Lab2: Custom URL Scheme (Deep link)

OWASP M7
Client code quality

Goal: find and use DVIA deep link

```
/* @class AppDelegate */
-(char)application:(void *)arg2 openURL:(void *)arg3 sourceApplication:(void *)arg4 annotation:(void *)arg5 {
    r7 = (sp - 0x14) + 0xc;
    sp = sp - 0x38;
    r6 = self;
    r11 = [arg3 retain];
    r7 = r7;
    r5 = [[r11 absoluteString] retain];
    if (r5 != 0x0) {
            [sp + 0x10 rangeOfString:r5, @"/call_number/"];
            if (var_20 == (0x80000000 ^ 0xffffffff)) {
                    r4 = 0x0;
            }
            else {
                    r10 = [[r6 getParameters:r11] retain];
                    r0 = [r10 objectForKey:@"phone"];
                    r7 = r7;
                    r0 = [r0 retain];
                    [r0 release];
                    if (r0 != 0x0) {
                            var_24 = [UIAlertView alloc];
                            r8 = [[r10 objectForKey:@"phone"] retain];
                            r4 = [[NSString stringWithFormat:@"Calling %@ without validation. Ring Ring !"] retain];
                            r6 = [var_24 initWithTitle:@"Success" message:r4 delegate:0x0 cancelButtonTitle:@"OK" otherButtonTitles:0x0];
                            [r6 show];
                            [r6 release];
                            [r4 release];
                            [r8 release];
                    }
                    [r10 release];
                    r4 = 0x1;
            }
    }
```

Lab2: Custom URL Scheme (Deep link)

OWASP M7
Client code quality

Goal: find and use DVIA deep link

```
/* @class AppDelegate */
-(char)application:(void *)arg2 openURL:(void *)arg3 sourceApplication:(void *)arg4 annotation:(void *)arg5 {
    r7 = (sp - 0x14) + 0xc;
    sp = sp - 0x38;
    r6 = self;
    r11 = [arg3 retain];
    r7 = r7;
    r5 = [[r11 absoluteString] retain];
    if (r5 != 0x0) {
            [sp + 0x10 rangeOfString:r5, @"/call_number/"];
            if (var_20 == (0x80000000 ^ 0xffffffff)) {
                    r4 = 0x0;
            }
            else {
                    r10 = [[r6 getParameters:r11] retain];
                    r0 = [r10 objectForKey:@"phone"];
                    r7 = r7;
                    r0 = [r0 retain];
                    [r0 release];
                    if (r0 != 0x0) {
                            var_24 = [UIAlertView alloc];
                            r8 = [[r10 objectForKey:@"phone"] retain];
                            r4 = [[NSString stringWithFormat:@"Calling %@ without validation. Ring Ring !"] retain];
                            r6 = [var_24 initWithTitle:@"Success" message:r4 delegate:0x0 cancelButtonTitle:@"OK" otherButtonTitles:0x0];
                            [r6 show];
                            [r6 release];
                            [r4 release];
                            [r8 release];
                    }
                    [r10 release];
                    r4 = 0x1;
            }
    }
```

Data security

Lab3: Cleartext Property List Files (Plist)

OWASP M2 Insecure data storage

Goal: find credentials stored in cleartext

Steps:

1. Open DVIA
2. Go to "**Insecure Data Storage**"
3. Click "Plist"
4. Enter credentials
5. Click "Save in Plist file"
6. The credentials are stored on your device
7. Get them with Objection:

```
ios plist cat userInfo.plist
```

Lab4: Keychain access

OWASP M2 Insecure data storage

Goal: find credentials stored in the keychain

Steps:

1. Open DVIA
2. Go to "**Insecure Data Storage**"
3. Click "Keychain"
4. Enter credentials
5. Click "Save in Keychain"
6. The credentials are stored on your device
7. Get them with Objection:

```
ios keychain dump
```

Syslog

```
Jan  6 17:35:45 iPad-de-davy kernel[0] <Notice>: xpcproxy[80834] Builtin profile: container (sandbox)
Jan  6 17:35:45 iPad-de-davy kernel[0] <Notice>: xpcproxy[80834] Container: /private/var/mobile/Containers/Data/Application/237ED97E-6F71-4650-B215-55C8E3831738
Jan  6 17:35:46 iPad-de-davy mytalkingtom[80834] <Notice>: MS:Notice: Injecting: com.outfit7.mytalkingtom [mytalkingtom] (1280.38)
Jan  6 17:35:46 iPad-de-davy mytalkingtom[80834] <Error>: MS:Error: unable to open() binary file
Jan  6 17:35:46 iPad-de-davy mytalkingtom[80834] <Notice>: MS:Notice: Loading: /Library/MobileSubstrate/DynamicLibraries/SSLKillSwitch2.dylib
Jan  6 17:35:46 iPad-de-davy mytalkingtom[80834] <Warning>: === SSL Kill Switch 2: Preference set to 1.
Jan  6 17:35:46 iPad-de-davy mytalkingtom[80834] <Warning>: === SSL Kill Switch 2: Subtrate hook enabled.
Jan  6 17:35:46 iPad-de-davy mytalkingtom[80834] <Error>: MS:Error: binary does not support this cpu type
Jan  6 17:35:46 iPad-de-davy mytalkingtom[80834] <Error>: MS:Error: failure to check trustme.dylib
Jan  6 17:35:46 iPad-de-davy gamecontrollerd[80836] <Notice>: MS:Notice: Injecting: (null) [gamecontrollerd] (1280.38)
Jan  6 17:35:47 iPad-de-davy gamecontrollerd[80836] <Error>: MS:Error: unable to open() binary file
Jan  6 17:35:47 iPad-de-davy gamecontrollerd[80836] <Warning>: === SSL Kill Switch 2: Preference set to 1.
Jan  6 17:35:47 iPad-de-davy gamecontrollerd[80836] <Warning>: === SSL Kill Switch 2: Subtrate hook enabled.
Jan  6 17:35:47 iPad-de-davy gamecontrollerd[80836] <Error>: MS:Error: binary does not support this cpu type
Jan  6 17:35:47 iPad-de-davy gamecontrollerd[80836] <Error>: MS:Error: failure to check trustme.dylib
Jan  6 17:35:47 iPad-de-davy mytalkingtom[80834] <Warning>: -> registered mono modules 0x10208f9f0
Jan  6 17:35:47 iPad-de-davy mytalkingtom[80834] <Warning>: You've implemented -[<UIApplicationDelegate> application:didReceiveRemoteNotification:fetchCompletion
ote-notification" to the list of your supported UIBackgroundModes in your Info.plist.
Jan  6 17:35:47 iPad-de-davy mytalkingtom[80834] <Notice>: <FIRAnalytics/INFO> Firebase Analytics v.3404000 started
Jan  6 17:35:47 iPad-de-davy mytalkingtom[80834] <Notice>: <FIRAnalytics/INFO> Successfully created Firebase Analytics App Delegate Proxy automatically. To disab
egateProxyEnabled to NO in the Info.plist
Jan  6 17:35:47 iPad-de-davy mytalkingtom[80834] <Warning>: AppsFlyer SDK version 4.6.3 started build (521)
Jan  6 17:35:47 iPad-de-davy mytalkingtom[80834] <Warning>: AppInit: r=1
Jan  6 17:35:47 iPad-de-davy mytalkingtom[80834] <Warning>: O7FunNetworkLib Version: 5.10.0
Jan  6 17:35:47 iPad-de-davy mytalkingtom[80834] <Warning>: *** -[NSKeyedUnarchiver initForReadingWithData:]: data is NULL
Jan  6 17:35:47 iPad-de-davy mytalkingtom[80834] <Warning>: -canOpenURL: failed for URL: "fb://" - error: "(null)"
Jan  6 17:35:47 iPad-de-davy mytalkingtom[80834] <Notice>: <FIRAnalytics/INFO> Firebase Analytics enabled
Jan  6 17:35:47 iPad-de-davy mytalkingtom[80834] <Notice>: <FIRAnalytics/INFO> Firebase Analytics enabled
```

Lab5: Syslog

OWASP M2 Insecure data storage

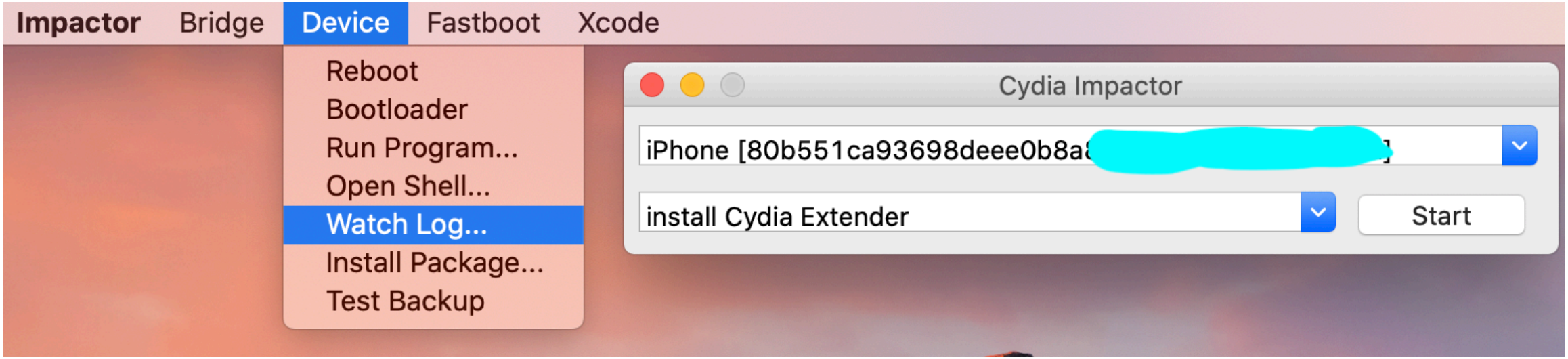
Goal: get sensitive info stored in the syslog

Steps:

1. Open **iGoat**
2. Go to “Key Management / **Random Key Generation**”
3. Find the encryption key
4. Get the syslog with Impactor GUI

Introducing Impactor

OWASP M2
Insecure data storage



Introducing Impactor

OWASP M2
Insecure data storage

davydouhine — Impactor idevicesyslog -u 80b551ca93698deee0b8a8b[redacted] — 111×24

```
Oct 27 10:40:16 iPhone routined(CoreLocation)[45] <Notice>: {"msg":"CLLocationManager", "event":"activity", "_c
md":"desiredAccuracy", "self":"0x10be6f580"}
Oct 27 10:40:16 iPhone routined(CoreLocation)[45] <Notice>: {"msg":"CLLocationManager", "event":"activity", "_c
md":"activityType", "self":"0x10be6f580"}
Oct 27 10:40:16 iPhone routined(CoreLocation)[45] <Notice>: {"msg":"CLLocationManager", "event":"activity", "_c
md":"stopUpdatingLocation", "self":"0x10be6f580"}
Oct 27 10:40:16 iPhone routined(CoreLocation)[45] <Notice>: {"msg":"state transition", "event":"state_transitio
n", "state":"LocationManager", "id":"0x10be6f580", "property":"updatingLocation", "old":1, "new":0}
Oct 27 10:40:16 iPhone routined(CoreLocation)[45] <Notice>: {"msg":"CLClientIsLocationServicesEnabled", "event"
:"activity"}
Oct 27 10:40:16 iPhone locationd[73] <Notice>: {"msg":"Incoming message", "event":"activity", "name":"kCLConnec
tionMessageLocation", "this":"0x12d0f2640", "registrationReceived":1}
Oct 27 10:40:16 iPhone locationd[73] <Notice>: Client /System/Library/LocationBundles/Routine.bundle (0x12d0f26
40) is unsubscribing to notification kCLConnectionMessageLocation
Oct 27 10:40:16 iPhone locationd[73] <Notice>: Client /System/Library/LocationBundles/Routine.bundle (0x12d0f26
40) is unsubscribing to notification kCLConnectionMessageLocationUnavailable
Oct 27 10:40:16 iPhone locationd[73] <Notice>: client '/System/Library/LocationBundles/Routine.bundle' unsubscr
ibing from location
Oct 27 10:40:16 iPhone locationd[73] <Notice>: @ClxClient, unsubscribe, /System/Library/LocationBundles/Routine
.bundle
Oct 27 10:40:16 iPhone locationd[73] <Notice>: #Warning Denying process assertion to <private>
Oct 27 10:40:18 iPhone locationd[73] <Notice>: os_transaction created: (<private>) <private>
Oct 27 10:40:18 iPhone locationd[73] <Notice>: os_transaction released: (<private>) <private>
```

Syslog

OWASP M2 Insecure data storage

To use Impactor using the command line you first need to get the device UDID

Get device UDID (serial number) on Linux:

```
lsusb -s :`lsusb | grep iPhone | cut -d ' ' -f 4 | sed 's/://'`
-v | grep iSerial | awk '{print $3}'
```

Get device UDID (serial number) on macOS:

```
idevice_id -l
```

or

```
ioreg -p IOUSB -l | grep "USB Serial"
```

(Don't copy the lsusb command from the slide, check **iOS_cheatsheet.txt**)

Syslog

OWASP M2
Insecure data storage

To use Impactor using the command line you first need to get the device UDID

Get device UDID (serial number) on Linux and Mac:
-> generate an error by launching Impactor without parameters !

```
./Impactor
```

```
┌──(kali㉿kali)-[~/iOS/Cydia-Impactor]
└─$ ./Impactor
I:0x7fa65c0018f8:80b551ca93698deee0b8a8b[redacted]
```

```
davys-macbook-pro$./Impactor
I:0x10182a2f8:80b551ca93698deee0b8a8b042[redacted]
```

Syslog

OWASP M2 Insecure data storage

Goal: get sensitive info stored in the syslog

Linux:

```
./Impactor idevicesyslog -u <UDID>
```

macOS:

```
./Applications/Impactor.app/Contents/MacOS/Impactor
idevicesyslog -u <UDID>
```

Execution analysis

Hooking

cycript

Cycript allows developers to explore and modify running applications on either iOS or Mac OS X using a hybrid of Objective-C++ and JavaScript syntax through an interactive console that features syntax highlighting and tab completion.
(It also runs standalone on Android and Linux and provides access to Java, but without injection.)

FRIDA

OVERVIEW DOCS NEWS CODE CONTACT

Inject JavaScript to explore native apps on Windows, macOS, Linux, iOS, Android, and QNX.

Introducing Frida

```
davy@kali:~$ frida -FH 192.168.1.25 -l frida.js

     ____
    / _  |   Frida 12.9.4 - A world-class dynamic instrumentation toolkit
   | (_| |
    > _  |   Commands:
   /_/ |_|       help      -> Displays the help system
   . . . .       object?   -> Display information about 'object'
   . . . .       exit/quit -> Exit
   . . . .
   . . . .   More info at https://www.frida.re/docs/home/
[Remote::DVIA]->
[Remote::DVIA]->
                ApiResolver
                Arm64Relocator
                Arm64Writer
                Array
                ArrayBuffer
                Backtracer
                Boolean
```

Frida: native API tracing

```
bibi:~ root# frida-trace -U -p 82924 -i "*URL*"
Instrumenting functions...
CaptiveCopyWiFiLandingPageURL: Auto-generated handler at "/private/var/root/__handlers__/CaptiveNetwork/CaptiveCopyWiFiLandingPageURL.js"
CFURLCreateBookmarkData: Auto-generated handler at "/private/var/root/__handlers__/Foundation/CFURLCreateBookmarkData.js"
_CFURLCreateDisplayPathComponentsArray: Auto-generated handler at "/private/var/root/__handlers__/Foundation/_CFURLCreateDisplayPathComponentsArray.js"
_CFBundleCreateWithExecutableURLIfMightBeBundle: Auto-generated handler at "/private/var/root/__handlers__/Foundation/_CFBundleCreateWithExecutableURL_3b8f9203.js"
_CFURLCreateByResolvingAliasFile: Auto-generated handler at "/private/var/root/__handlers__/Foundation/_CFURLCreateByResolvingAliasFile.js"
_CFURLCreateFromComponents: Auto-generated handler at "/private/var/root/__handlers__/Foundation/_CFURLCreateFromComponents.js"
CFBundleCopyExecutableURL: Auto-generated handler at "/private/var/root/__handlers__/Foundation/CFBundleCopyExecutableURL.js"
_CFURLIsItemPromiseAtURL: Auto-generated handler at "/private/var/root/__handlers__/Foundation/_CFURLIsItemPromiseAtURL.js"
CFCopyHomeDirectoryURL: Auto-generated handler at "/private/var/root/__handlers__/Foundation/CFCopyHomeDirectoryURL.js"
CFURLCreateFilePathURL: Auto-generated handler at "/private/var/root/__handlers__/Foundation/CFURLCreateFilePathURL.js"
```

(...)

```
CMByteStreamCreateForFileURL: Auto-generated handler at "/private/var/root/__handlers__/CoreMedia/CMByteStreamCreateForFileURL.js"
FigNote_FlushRunningLogAndCopyURLContainingLogs: Auto-generated handler at "/private/var/root/__handlers__/CoreMedia/FigNote_FlushRunningLogAndCopyUR_0e23347a.js"
Started tracing 1165 functions. Press Ctrl+C to stop.
           /* TID 0xc07 */
 19170 ms  CFURLCopyScheme()
 19171 ms  CFURLCopyAbsoluteURL()
 19171 ms  CFURLCopyFileSystemPath()
 19182 ms  CFURLCopyScheme()
 19184 ms  CFURLCopyAbsoluteURL()
 19184 ms  CFURLCopyFileSystemPath()
 19214 ms  CFBundleCopyResourceURL()
 19215 ms     | CFURLCreateWithFileSystemPath()
 19215 ms     |    | CFURLCreateWithFileSystemPathRelativeToBase()
```

Frida: CodeShare

Frida CodeShare

Log In

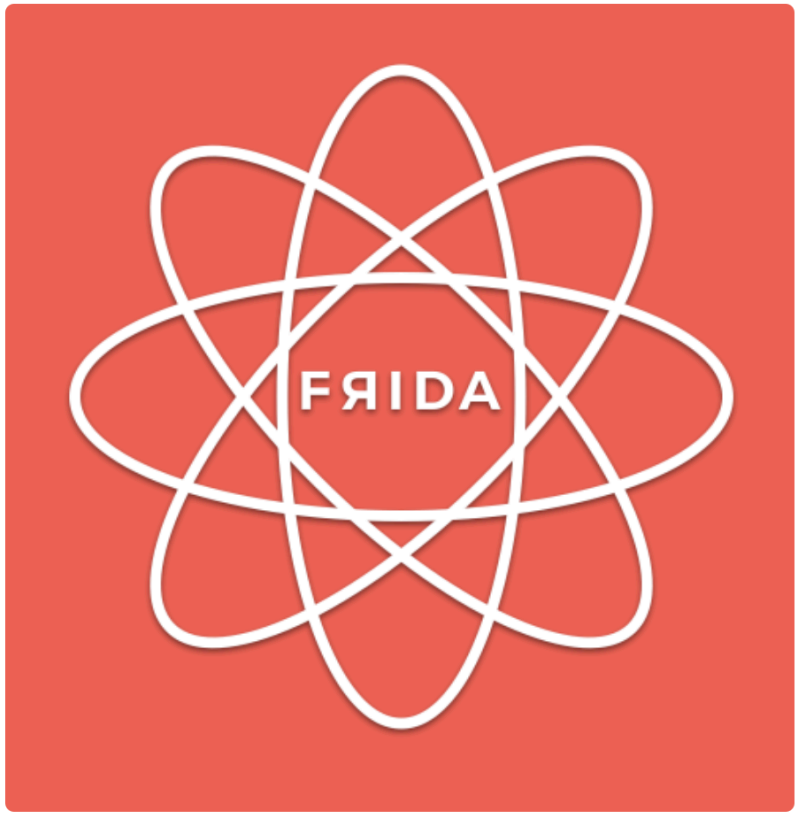
Unleash the power of Frida.

"If I have seen further, it is by standing on the shoulders of giants."
-Sir Issac Newton

The Frida CodeShare project is comprised of developers from around the world working together with one goal - push Frida to its limits in new and innovative ways.

Frida has amazing potential, but needed a better forum to share ideas, so we've put together CodeShare to help stoke innovation and push Frida to its highest potential!

BROWSE CODE



Frida: CodeShare

iOS DataProtection

7 | 4K

Uploaded by: @ay-kay

List iOS file data protection classes (NSFileProtectionKey) of an app

PROJECT PAGE

aesinfo

7 | 5K

Uploaded by: @dzonerzy

Show useful info about AES encryption/decryption at application runtime

PROJECT PAGE

frida-multiple-unpinning

5 | 4K

Uploaded by: @akabe1

Another Android ssl certificate pinning bypass script for various methods (https://gist.github.com/akabe1/5632cbc1cd49f0237cbd0a93bc8e4452)

PROJECT PAGE

ObjC method observer

4 | 8K

Uploaded by: @mrmacete

Observe all method calls to a specific class (e.g. observeClass('LicenseManager')) , or dynamically resolve methods to observe using ApiResolver (e.g. observeSomething('*[* *Password:*]')). The script tries to do its best to resolve and display input parameters and return value. Each call log comes with its stacktrace.

PROJECT PAGE

Frida: CodeShare: ObjC-method-observer

```
davy@kali:~$ frida -FH 192.168.1.25 --codeshare mrmacete/objc-method-observer
     ____
    / _  |   Frida 12.9.4 - A world-class dynamic instrumentation toolkit
   | (_| |
    > _  |   Commands:
   /_/ |_|       help      -> Displays the help system
   . . . .       object?   -> Display information about 'object'
   . . . .       exit/quit -> Exit
   . . . .
   . . . .   More info at https://www.frida.re/docs/home/
Hello! This is the first time you're running this particular snippet, or the snippet's source code has changed.

Project Name: ObjC method observer
Author: @mrmacete
Slug: mrmacete/objc-method-observer
Fingerprint: 1d6348576fd097903d4e4f897b332e028ec802418d66c3883f60adeca6d12539
URL: https://codeshare.frida.re/@mrmacete/objc-method-observer

Are you sure you'd like to trust this project? [y/N]
```

Frida: CodeShare: ObjC-method-observer

```
[Remote::iGoat]-> observeSomething('*[* *Pin:*]');
Observing  -[NEProfilePayloadBaseVPN setPin:]
Observing  -[BruteForceRuntimeVC validatePin:]
Observing  -[_UIActivityHelper _activitiesByApplyingBeforeTypePinningToActivities:activitiesToBeforeTypePin:]
Observing  -[CoreTelephonyClient unlockPUK:puk:newPin:completion:]
Observing  -[CoreTelephonyClient changePIN:oldPin:newPin:completion:]
Observing  -[CoreTelephonyClient unlockPUK:puk:newPin:error:]
Observing  -[CoreTelephonyClient changePIN:oldPin:newPin:error:]
Observing  -[DMFEraseDeviceRequest setPin:]
Observing  -[DMFLockDeviceRequest setPin:]
[Remote::iGoat]->
```

Frida: CodeShare: ObjC-method-observer

```
(0x1405232e0)  -[BruteForceRuntimeVC validatePin:]
validatePin: 1181
0x100166a30 iGoat!-[BruteForceRuntimeVC buttonClick:]
0x1877e0ad0 UIKit!-[UIApplication sendAction:to:from:forEvent:]
0x1877e0a4c UIKit!-[UIControl sendAction:to:forEvent:]
0x1877c8740 UIKit!-[UIControl _sendActionsForEvents:withEvent:]
0x1877e033c UIKit!-[UIControl touchesEnded:withEvent:]
0x1877dff6c UIKit!-[UIWindow _sendTouchesForEvent:]
0x1877d8b08 UIKit!-[UIWindow sendEvent:]
0x1877a8f4c UIKit!-[UIApplication sendEvent:]
0x1877a7528 UIKit!_UIApplicationHandleEventQueue
0x18261d124 CoreFoundation!__CFRUNLOOP_IS_CALLING_OUT_TO_A_SOURCE0_PERFORM_FUNCTION__
0x18261cbb8 CoreFoundation!__CFRunLoopDoSources0
0x18261a8b8 CoreFoundation!__CFRunLoopRun
0x182544d10 CoreFoundation!CFRunLoopRunSpecific
0x183e2c088 GraphicsServices!GSEventRunModal
0x187811f70 UIKit!UIApplicationMain
0x10015ed74 iGoat!main
RET: 0x1
```

Lab6: Frida login bypass

OWASP M8 code tampering

Goal: bypass a security check using Frida

Steps:

1. Open DVIA
2. Go to “**Runtime Manipulation**” / “Start Challenge”
3. Find the class and the method used to check the login
4. Instrument DVIA using Frida to bypass the login

Lab6: Frida login bypass

OWASP M8 code tampering

Steps (detailed):

1. Use **frida** and **ObjC-Method-Observer** to find the class/method used for the login:

```
frida -FU -c mrmacete/objc-method-observer
observeSomething('*[* *Login:*]');
```

Lab6: Frida login bypass

OWASP M8 code tampering

Steps (detailed):

1. Use **frida** and **ObjC-Method-Observer** to find the class/method used for the login:

```
frida -FU -c mrmacete/objc-method-observer
observeSomething('*[* *Login:*]');
```

2. Use **frida-trace:**

```
frida-trace -FU -m "-[RuntimeManipulationDetailsVC
isLoginValidated]"
```

to add “backtrace” and confirm the class/method used

Lab6: Frida login bypass

OWASP M8 code tampering

Steps (detailed):

1. Use **frida** and **ObjC-Method-Observer** to find the class/method used for the login:

```
frida -FU -c mrmacete/objc-method-observer
observeSomething('*[* *Login:*]');
```

2. Use **frida-trace:**

```
frida-trace -FU -m "-[RuntimeManipulationDetailsVC
isLoginValidated]"
```

to add “backtrace” and confirm the class/method used

3. Use the JS handler created by **frida-trace** to modify the method result

Add `retval.replace(1);` to the `onLeave` function

Lab7: Objection login bypass

OWASP M8 code tampering

Goal: bypass a security check using Objection

Steps:

1. Open DVIA
2. Go to "**Runtime Manipulation**" / "Start Challenge"
3. Instrument DVIA using Objection to bypass the login

Lab7: Objection login bypass

OWASP M8 code tampering

Steps (detailed):

Use **objection** and:

`ios hooking list class_methods` to list class/method used for the login

Lab7: Objection login bypass

OWASP M8 code tampering

Steps (detailed):

Use **objection** and:

- `ios hooking list class_methods` to list class/method used for the login
- `ios hooking watch method` to add “backtrace” and confirm again the class/method

Lab7: Objection login bypass

OWASP M8 code tampering

Steps (detailed):

Use **objection** and:

- **`ios hooking list class_methods`** to list class/method used for the login
- **`ios hooking watch method`** to add “backtrace” and confirm again the class/method
- **`ios hooking set return_value`** to modify the method result

Transport security

Lab8: network capture with rvictl

OWASP M3
Insecure Communication

Only for macOS - sorry

Goal: analyze network communications

Steps:

1. Connect a device using USB and click on “Trust”
2. Get device UDID (serial number)
3. Create redirection:

```
rvictl -s 81e9e294d31bf751b778c7abbb6bf83ee6dec8fd
```

4. Capture network traffic with tcpdump or wireshark

```
tcpdump -Xni rvi0
```

5. Launch DVIA, go to “**Transport Layer Security**” / “Send over HTTP”
6. Delete redirection:

```
rvictl -x 81e9e294d31bf751b778c7abbb6bf83ee6dec8fd
```

Lab8: network capture with rvi_capture

OWASP M3
Insecure Communication

Goal: analyze network communications

For Linux :)

Steps:

1. Connect a device using USB and click on “Trust”
2. Get device UDID (serial number)
(using the lsusb command in ***iOS_cheatsheet.txt***)
3. Launch capture:

```
./rvi_capture.py --udid <UDID> iPhone.pcap
```

4. Launch DVIA, go to “**Transport Layer Security**” / “Send over HTTP”
5. Stop capture and analyze network traffic with tcpdump or wireshark

```
tcpdump -Xnr iPhone.pcap
```

Questions