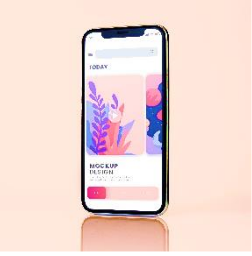
Mobile Hacking

ASSESSING MOBILE APPLICATIONS



CHEAT SHEET

V0.1

MAIN STEPS

- Review the codebase
- Run the app
- Dynamic instrumentation
- Analyze network communications

OWASP MOBILE SECURITY PROJECTS

Mobile Security Testing Guide
- https://github.com/OWASP/owasp-mstg

Mobile Application Security Verification Standard
- https://github.com/OWASP/owasp-masvs

Mobile Security Checklist
- https://github.com/OWASP/owasp-mstg/tree/master/Checklists

TOOLS

- Frida
- Objection
- Impactor
- BurpSuite
- Wireshark

Filesystem

UUID (Universally Unique Identifier): random 36 alphanumeric characters string unique to the app
Data-UUID: random 36 alphanumeric characters string unique to the app

/User/Library/FrontBoard/applicationState.db
- App list database

/private/var/containers/Bundle/Application/UUID/App.app
- Binary directory: include all the static resources of the app

/private/var/containers/Bundle/Application/UUID/App.app/App
- Path of the binary (executable)

/private/var/containers/Bundle/Application/UUID/App.app/Info.plist
- App metadata: configuration of the app (icon to display, supported document types, etc.)

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

Bundle ID

The bundle ID represents the app's unique identifier (e.g. for YouTube)

com.google.ios.youtube

How to find the data and binary directories

Grep is the not-so-quick 'n dirty way to find where are the data and binary directories of your app

```
iPhone:~ root# grep -r <App_name> /private/var/*
```

How to find the data and binary directories and the Bundle ID

By launching Frida with the ios-app-info script

```
# frida -U <App_name> -c dki/ios-app-info
```

And then

```
[iPhone::App]-> appInfo()
```

Or manually by opening the app list database

```
iPhone:~ root# sqlite3 /User/Library/FrontBoard/applicationState.db
```

And displaying the key_tab table to get the binary directories

```
sqlite> select * from key_tab;
```

Or displaying the application_identifier_tab table to get the bundle IDs

```
sqlite> select * from application_identifier_tab;
```

App decryption

1. Add level3tjg.github.io src to Cydia and install bfdecrypt tool
2. Go to bfdecrypt pref pane in Settings and set the app to decrypt
3. Launch the app: decrypted IPA is stored in the Documents folder of the app

Dynamic analysis with Frida

List all processes

```
# frida-ps –U
```

Analyse the calls to a method by launching Frida with the objc-method-observer script

```
# frida -U <App_name> –c mrmacete/objc-method-observer
```

And then using the command 'observeSomething'

```
[iPhone::App]-> observeSomething('*[* *doSecurityChecks*]');
```

Hook the calls to a method

```
# frida-trace -U <App_name> -m "-[NSFileManager fileExistsAtPath*]"
```

Then open the JavaScript handler file to edit the **onEnter** or **onLeave** functions to manipulate the behavior of the app

Dynamic analysis with Cycript

Get a reference of the app instance

```
app = [UIApplication sharedApplication]
```

Display the data directory of the app

```
app.userHomeDirectory
```

Display the Bundle ID of the app

```
NSBundle.mainBundle.bundleIdentifier
```

Display the content of a directory

```
[[NSFileManager defaultManager] contentsOfDirectoryAtPath:@"/var/mobile/Containers/Data/"]
```

Check the existence of a file

```
[[NSFileManager defaultManager] fileExistsAtPath: @"/etc/passwd"]
```

Display the content of a file

```
[[NSFileManager defaultManager] contentsAtPath: @"/etc/passwd"]
```

Get the NSLog (syslog)

Impactor (http://www.cydiaimpactor.com) let you display the NSLog (syslog) on command line

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

RANDORISEC

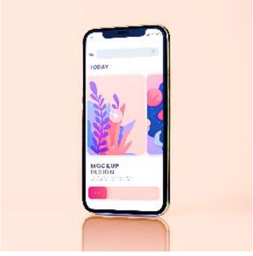
CC BY-SA 4.0 • contact@randorisec.fr • https://www.randorisec.fr

Version 0.1 • Updated: 2020-01

Template: https://rstudio.com/resources/cheatsheets/how-to-contribute-a-cheatsheet/ Background psd created by rawpixel.com – https://www.freepik.com

Mobile Hacking

ASSESSING MOBILE APPLICATIONS



CHEAT SHEET

V0.1

MAIN STEPS

- Review the codebase
- Run the app
- Dynamic instrumentation
- Analyze network communications

OWASP MOBILE SECURITY PROJECTS

Mobile Security Testing Guide
- https://github.com/OWASP/owasp-mstg

Mobile Application Security Verification Standard
- https://github.com/OWASP/owasp-masvs

Mobile Security Checklist
- https://github.com/OWASP/owasp-mstg/tree/master/Checklists

TOOLS

- Frida
- Objection
- Impactor
- BurpSuite
- Wireshark

SSL Interception with BurpSuite

1. Launch Burp and modify proxy settings in order to listen on “All interfaces”
2. Browse to the IP/port of your Burp proxy using Safari
3. Tap on the “CA Certificate” at the top right of the screen
4. Tap on “Allow” on the pop-up asking to download a configuration profile
5. Go to “Settings->Profile Downloaded” and select the “PortSwigger CA” profile
6. Tap on “Install” then “Install” again and then “Install” one last time
7. Edit the wireless network settings on your device to set a proxy (“Settings->Wi-Fi” then tap on the blue “i”, slide to the bottom of the screen and tap on “Configure Proxy”)
8. Tap on ”Manual”, set the IP/port of your Burp proxy, tap on “Save”
9. Go to “Settings->General->About->Certificate Trust Settings” & toggle on the PortSwiggerCA

Bypass SSL Pinning using SSL Kill Switch 2

Download and install SSL Kill Switch 2 tweak

```
# wget https://github.com/nabla-c0d3/ssl-kill-switch2/releases/download/0.14/com.nablac0d3.sslkillswitch2_0.14.deb
# dpkg -i com.nablac0d3.sslkillswitch2_0.13.deb
# killall -HUP SpringBoard
```

Go to “Settings->SSL Kill Switch 2” to ”Disable Certificate Validation”

UDID (Unique Device Identifier)

UDID is a string that is used to identify a device. Needed for some operations like signature, app installation, network monitoring

Get UDID with MacOS

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

Get UDID with Linux

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

Network capture (works also on non jailbroken devices)

MacOS (install Xcode and additional tools and connect the device with USB)

```
# rvictl -s <UDID>
# tcpdump | tshark | wireshark –i rvi0
```

Linux (get https://github.com/gh2o/rvi_capture and connect the device with USB)

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

Sideloading an app

Sideloading an app including an instrumentation library like Frida or Cycript let you interact with the app even if it’s installed on a non jailbroken device.

Here’s the process to do it with IPAPatch:

1. Clone the IPAPatch project

```
# git clone https://github.com/Naituw/IPAPatch
```

2. Move the IPA of the app you want to sideload to the Assets directory

```
# mv <IPAfile> IPAPatch/Assets/
```

3. Download the FridaGadget library (in **Assets/Dylibs/FridaGadget.dylib**)

```
# curl -O https://build.frida.re/frida/ios/lib/FridaGadget.dylib
```

4. Select the identity to sign the app

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

5. Sign FridaGadget library

```
# codesign -f -s <IDENTITY> FridaGadget.dylib
```

6. Then open IPAPatch Xcode project, Build and Run.

Here’s the process to do it with Objection (detailed steps on https://github.com/sensepost/objection/wiki/Patching-iOS-Applications)

```
# security find-identity -p codesigning –v
# objection patchipa --source <IPAfile> --codesign-signature <IDENTITY>
# unzip <patchedIPAfile>
# ios-deploy --bundle Payload/my-app.app -W –d
# objection explore
```

Data Protection Class

Four levels are provided by iOS to encrypt automatically files on the device:

1. **NSProtectionComplete:** file is only accessible when device is unlocked (files are encrypted with a key derived from the user PIN code & an AES key generated by the device)
2. **NSProtectionCompleteUntilFirstUserAuthentication:** (defaut class) same except as before, but the decryption key is not deleted when the device is locked
3. **ProtectedUnlessOpen:** file is accessible until open
4. **NoProtection:** file is accessible even if device is locked

Get Data Protection Class

By launching Frida with the ios-dataprotection script

```
# frida -U <App_name> -c ay-kay/ios-dataprotection
```

RANDORISEC

• contact@randorisec.fr • https://www.randorisec.fr

Version 0.1 • Updated: 2020-01

The OWASP brand is the property of the OWASP Foundation. OWASP does not endorse any product, services or tools.

Template: https://rstudio.com/resources/cheatsheets/how-to-contribute-a-cheatsheet/ Background psd created by rawpixel.com – https://www.freepik.com