iOS Security: Deep Dive

Vadim Drobinin | @valzevul

About me

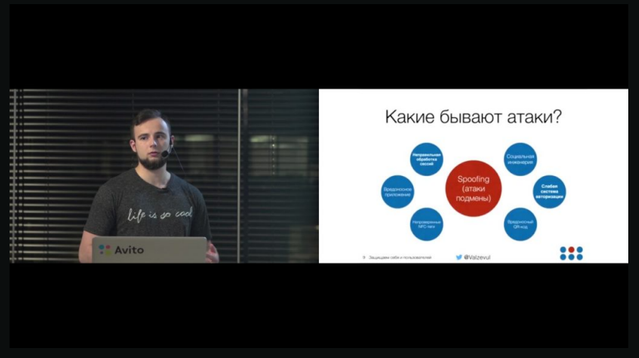

CocoaHeads Russia Декабрь 2016
Как задаётся токен
Кредитная карта
PAN, CVV, Exp. Date = X
Пользователь
Apple Pay
Apple Pay Servers
Банк
Token, Token-key, cvv-key = Y
Secure Element
Token Service Provider


"SEATBELT"


like a Pro
Vadim Drobinin | drobinin.com


NLP
...Grammar induction...
...PoC-tagging...
IF


Big θ Notation in your Average App


Какие бывают атаки?
Spoofing (атаки подмены)
Avito

Agenda (part I: Introduction)

- Why bother?
- Life without a jailbreak?
- Tools Overview
- Platform Overview
- Security 101:
  - OWASP
  - Data protection

Agenda (part II: Penetration testing)

- Local Authentication
- Network API
- Universal Links
- WebViews
- Unusual attack vectors
- Jailbreak detection
- Where to go from here

Disclaimer

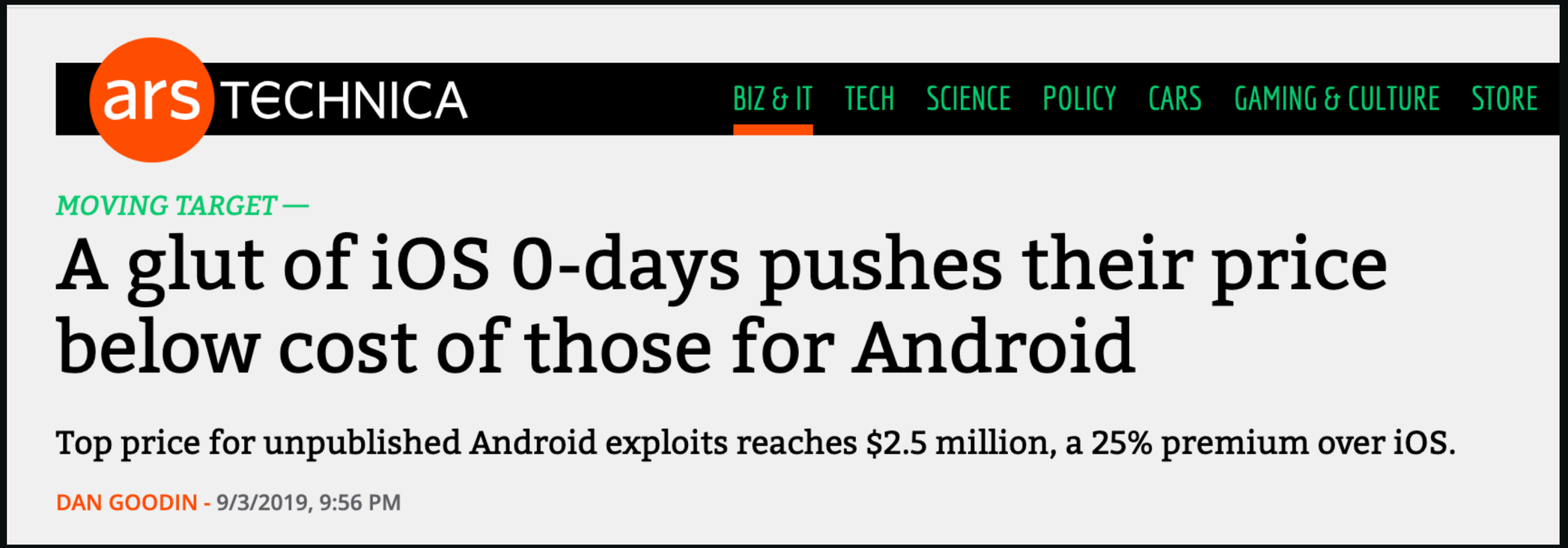

Why bother?

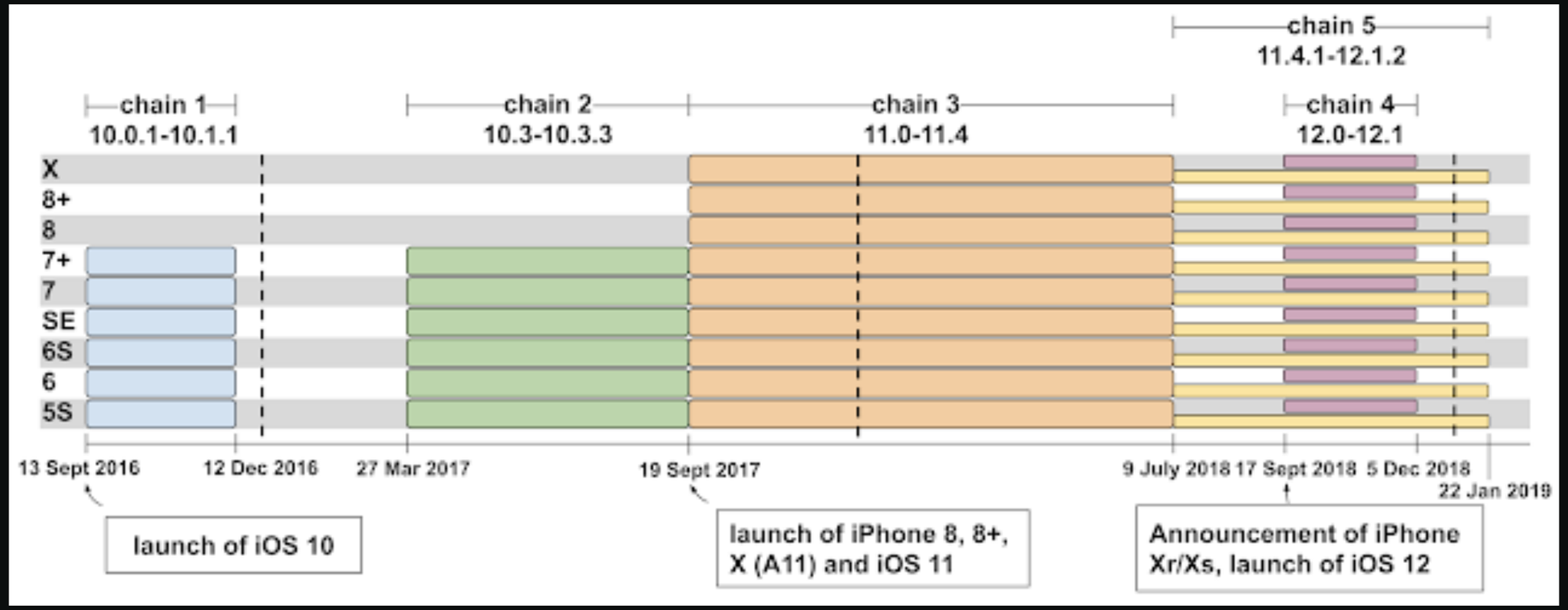
We've built advanced security into our products from the ground up **to make them secure by design** [...]

— iOS Security Overview

Does it work?


NO


ars TECHNICA
BIZ & IT TECH SCIENCE POLICY CARS GAMING & CULTURE STORE
MOVING TARGET —
A glut of iOS 0-days pushes their price below cost of those for Android
Top price for unpublished Android exploits reaches $2.5 million, a 25% premium over iOS.
DAN GOODIN - 9/3/2019, 9:56 PM



© https://googleprojectzero.blogspot.com/2019/08/a-very-deep-dive-into-ios-exploit.html

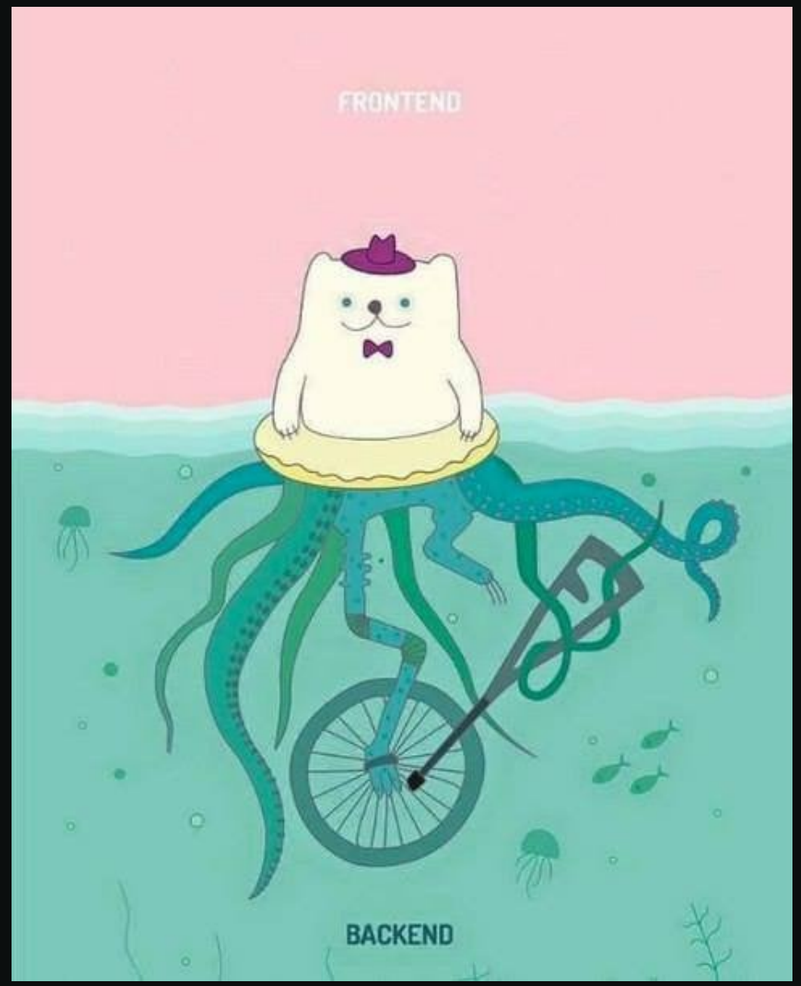
Mobile devices are the main source of users' private data *

Mobile devices are the main source of users' personal data *

* and we rarely protect it well enough

Protect from what?

Never trust frontends




Do I need
Jailbreak?

No Jailbreak

- ❌ Limited toolkit
- ❌ Inconvenient side-loading
- ✅ Real-world scenarios
- ✅ Sounds legal
- ✅ Always possible

Jailbreak

- ❌ Sometimes illegal
- ❌ Sometimes unstable
- ❌ Sometimes impossible
- ✅ Sounds fancy
- ✅ Versatile toolkit
- ✅ Easy side-loading

No Jailbreak

- Downloading application package*
- Setting up the environment
- Injecting custom `dylib` & modification of executable file
- Repacking and signing the package
- Installing the app on device in debug mode

No Jailbreak: Tools

Fridpa | github.com/tanprathan/Fridpa
An automated wrapper script for unpacking, patching, re-signing and deploying apps on a non-jailbroken device.

No Jailbreak: Tools

Fridpa | github.com/tanprathan/Fridpa
An automated wrapper script for unpacking, patching, re-signing and deploying apps on a non-jailbroken device.

Apple Configurator 2 | apps.apple.com
Among other features, it allows access to the device logs.

No Jailbreak: Tools

Fridpa | github.com/tanprathan/Fridpa
An automated wrapper script for unpacking, patching, re-signing and deploying apps on a non-jailbroken device.

Apple Configurator 2 | apps.apple.com
Among other features, it allows access to the device logs.

Objection | github.com/sensepost/objection
A runtime mobile exploration toolkit built to help you assess the security posture of your mobile applications, without needing a jailbreak.

Jailbreak: Tools

Frida | frida.re
Dynamic instrumentation toolkit for developers, reverse-engineers, and security researchers.

Jailbreak: Tools

Frida | frida.re
Dynamic instrumentation toolkit for developers, reverse-engineers, and security researchers.

Passionfruit | github.com/chaitin/passionfruit
Simple iOS app blackbox assessment tool. Powered by frida.re and vuejs.

Jailbreak: Tools

Frida | frida.re
Dynamic instrumentation toolkit for developers, reverse-engineers, and security researchers.

Passionfruit | github.com/chaitin/passionfruit
Simple iOS app blackbox assessment tool. Powered by frida.re and vuejs.

SSL-kill-switch | github.com/nabla-c0d3/ssl-kill-switch2
Blackbox tool to disable SSL certificate validation - including certificate pinning - within iOS and OS X Apps.

Jailbreak: Tools

Frida | frida.re
Dynamic instrumentation toolkit for developers, reverse-engineers, and security researchers.

Passionfruit | github.com/chaitin/passionfruit
Simple iOS app blackbox assessment tool. Powered by frida.re and vuejs.

SSL-kill-switch | github.com/nabla-c0d3/ssl-kill-switch2
Blackbox tool to disable SSL certificate validation - including certificate pinning - within iOS and OS X Apps.

and many many more...

Other tools

Burp Suite Community | portswigger.net/burp
Proxy your HTTPS traffic, edit and repeat requests, decode data, and more.

Other tools

Burp Suite Community | portswigger.net/burp
Proxy your HTTPS traffic, edit and repeat requests, decode data, and more.

HopperApp | www.hopperapp.com/
Hopper Disassembler, the reverse engineering tool that lets you disassemble, decompile and debug your applications.

Other tools

Burp Suite Community | portswigger.net/burp
Proxy your HTTPS traffic, edit and repeat requests, decode data, and more.

HopperApp | www.hopperapp.com/
Hopper Disassembler, the reverse engineering tool that lets you disassemble, decompile and debug your applications.

iMazing | imazing.com
File manager which also allows you to extract `ipa` on non-jailbroken devices.

Platform Overview

Platform Overview

- iOS is based on Darwin
- Secure boot
- Hardware-backed Keychain
- File system encryption
- Update rollouts
- iOS apps are isolated via Apple's iOS sandbox (“Seatbelt”)

“Seatbelt”

- OSX 10.5 “Leopard”, 2007
- Not mandatory
- Not many developers did this

“Seatbelt”

- OSX 10.7 “Lion”, 2011
  - **com.apple.security.app-sandbox** entitlement
  - Added automatically when signed via App Store
- iOS:
  - **/var/mobile/Containers** and **/var/Containers**

What's not safe?

- Usernames and passwords
- Location data
- Facial data
- Advertising data
- Address book entries
- Payment information
- Other personal information

OWASP*

* The Open Web Application Security Project, https://owasp.org/

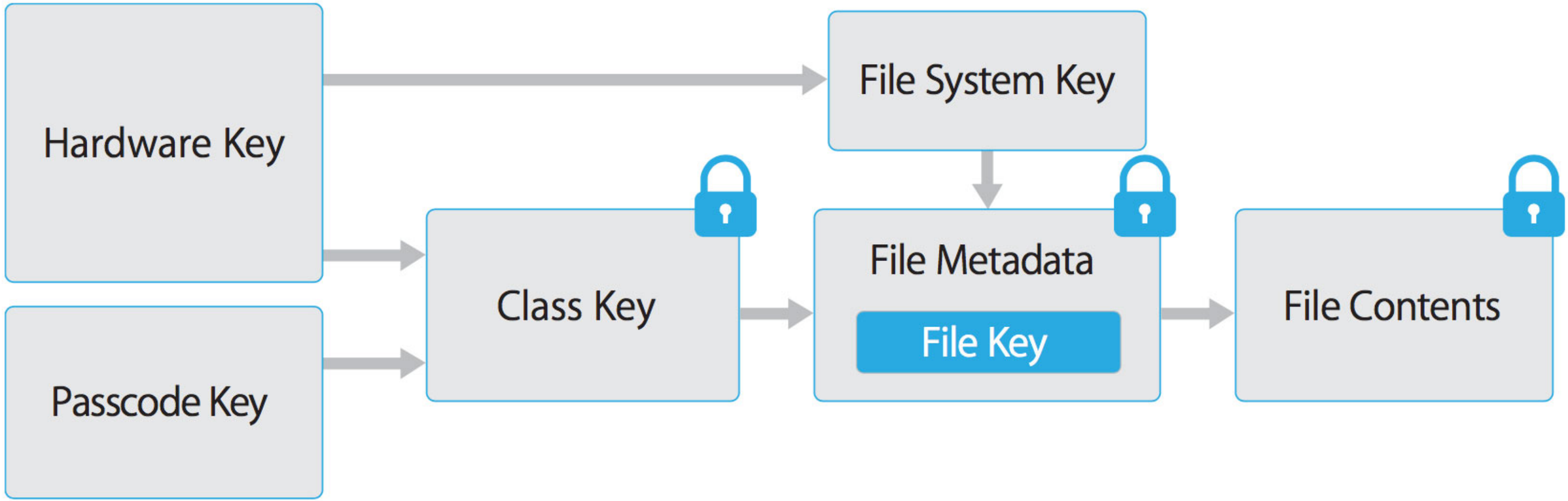
Essential parts

- Device
  - Local storage
  - Interaction with the mobile platform
- APIs
  - Communication with trusted endpoints
  - Authentication and Authorisation
- Prevention
  - Anti-Reversing

As little sensitive data as possible should be saved in permanent local storage.

Data Protection API

Data Storage on iOS



Protection Classes:

- Complete Protection (*NSFileProtectionComplete*)
- Protected Unless Open (*NSFileProtectionCompleteUnlessOpen*)
- **Protected Until First User Authentication** (*NSFileProtectionCompleteUntilFirstUserAuthentication*)
- No Protection (*NSFileProtectionNone*)

The Keychain

- Only one Keychain is available to all apps
- Access control among apps via *kSecAttrAccessGroup*

The Keychain

- Only one Keychain is available to all apps
- Access control among apps via *kSecAttrAccessGroup*
- Access for items:

```
kSecAttrAccessibleAlways
kSecAttrAccessibleAlwaysThisDeviceOnly
kSecAttrAccessibleAfterFirstUnlock
kSecAttrAccessibleAfterFirstUnlockThisDeviceOnly
kSecAttrAccessibleWhenUnlocked
kSecAttrAccessibleWhenUnlockedThisDeviceOnly
kSecAttrAccessibleWhenPasscodeSetThisDeviceOnly
```

Keychain Access Control flags

```
kSecAccessControlDevicePasscode
kSecAccessControlTouch IDAny
kSecAccessControlTouch IDCurrentSet
kSecAccessControlUserPresence
```

How to work with the Keychain

```
func devicePasscodeEnabled() -> Bool {
    return LAContext().canEvaluatePolicy(.deviceOwnerAuthentication,
                                         error: nil)
}

let userDefaults = UserDefaults.standard
if userDefaults.bool(forKey: "hasRunBefore") == false {
     // Remove Keychain items here
     userDefaults.set(true, forKey: "hasRunBefore")
     userDefaults.synchronize() // Forces the app to update UserDefaults
}

func logout() {
    // Logout the user here
    wipeKeychain()
}
```

What might go wrong?

- Make sure nothing sensitive (password, keys, tokens, other PII, etc) is stored in *NSUserDefaults* or via *NSData*, *writeToFile*, *NSFileManager*, CoreData, databases, etc without encryption.
- If the encryption is used, make sure the secret key is stored in the Keychain with secure settings,
  ideally *[…]WhenPasscodeSetThisDeviceOnly*.

Be careful with Firebase

- 47% of iOS apps that connect to a Firebase database are vulnerable[1]
- Get `PROJECT_ID` from *GoogleService-Info.plist*
- Check https://<firebaseProjectName>.firebaseio.com/.json
- Firebase Scanner https://github.com/shivsahni/FireBaseScanner

[1] Appthority Mobile Threat Team, Jan 2018

Be careful with Realm

```
// Generate a random encryption key
var key = Data(count: 64)
_ = key.withUnsafeMutableBytes { bytes in
    SecRandomCopyBytes(kSecRandomDefault, 64, bytes)
}

// Open the encrypted Realm file
let config = Realm.Configuration(encryptionKey: key)
do {
    let realm = try Realm(configuration: config)
    // Use as normal
} catch let error as NSError {
    // Wrong key -> “Invalid Database” error
}
```

Dynamic Analysis via iMazing

- Trigger the functionality that stores potentially sensitive data.
- Connect the iOS device and launch iMazing.
- Select the app and do "Extract App"
- Navigate to the output directory and locate *$APPNAME.imazing*. Rename it *$APPNAME.zip*.
- Unpack the zip file.
- To get Keychain items on a non-JB device, use objection

Other locations of sensitive data

```
textObject.autocorrectionType = .no
textObject.secureTextEntry = true
```

- Keyboard cache
- Logs
- Backups
- Auto-generated (overlay) screenshots
- Memory

True excellence at mobile application security requires a **deep understanding** of mobile operating systems, coding, network security, cryptography, and **a whole lot of other things**.

— OWASP


MORDAC, THE PREVENTER OF INFORMATION SERVICES.
SECURITY IS MORE IMPORTANT THAN USABILITY.
www.dilbert.com scottadams@aol.com
IN A PERFECT WORLD, NO ONE WOULD BE ABLE TO USE ANYTHING.
©2007 Scott Adams, Inc./Dist. by UFS, Inc
To complete the log-in procedure, stare directly at the sun.

Thank you

Part 2: Penetration Testing
at 15:10 CEST

drobinin.com | @valzevul