Keeping Secrets for iOS Developers

Randy Beiter

Plugh Studios, Ltd.

Our Goal Is More Secure iOS Apps for the App Store

The Right Balance For Your Application

Weigh Usability / Security

Don’t Overwhelm the User

Use the right level of security for the job

Assume Users Expect Privacy

Data protection makes basic security easy – use it

Pick an Appropriate Scheme for Your Domain

Balance cost of security with cost of a breach

Tie secrets to your user; establish trust externally

Challenging to Hide Bundled Sensitive Data

Application binary is not 100% safe from prying eyes

Consider storing application-level secrets remotely

Store user secrets in iOS keychain

The Is No Perfect Security

If you assume the worse, you can better try to plan for it

Mitigate Impact When Security Fails

Expose as little as possible

Don’t make it easy

Apps On An Unmodified Device Must Be Signed

Apps Are Signed By

You, the developer

Apple

Code Signing Is Not:

Encryption Of Your Data

Hiding Of Your App’s Resources

Apps delivered by AppStore are just specially named ZIP files

Modification Prevention

(at least not for Non-Bundled Data)

Code Signing Is:

Mechanism to Prevent Running Unauthorized Applications on Stock Devices

Encryption Of Your Binary
(but don't count on this to keep your secrets)

Assurance Your Bundle Is Not Modified

Light Piracy Protection

Protect User Data and Application Data

Mechanisms The Could Expose Data:

Browsing with iExplorer or similar
iTunes backup
Your app bundle and binary
Network traffic inspection

Application Binary is Encrypted From App Store

Don’t rely on this – cracked apps are laid bare

All Other Bundled Resources Are Not

Images, databases, data files are in the clear

A Little Obfuscation Goes a Long Way

Simple encoding / obfuscation

Deter Casual Browsing

Fight the compiler making strings too browsable

Bundled Files Are Transparent

Application binary is encrypted
None of your other resources are
PNG files are optimized for iOS

Consider Intellectual Property Issues

If You Use IAP – Validate Receipts

Don’t Limit Security Strategy to Purely Visual Measures

NSUserDefaults entries are stored in the clear

NSDocumentDirectory Is Backed Up by iTunes

… and stored on a possible shared computer, in the clear by default

NSCachesDirectory and NSTemporaryDirectory Are Not Backed Up

App Data Directories Are Browsable

App Data Directories Are Mutable

Obscure Sensitive Resources

- Byte Swapping/Bit Shifting
- Full Blown Encryption
- Bundle Resources Into Single File

Hash Files to Deter Tampering

Data Protection Overview

Multiple Accessibility Options

NSFileProtectionComplete

NSFileProtectionCompleteUnlessOpen (iOS 5)

NSFileProtectionCompleteUntilFirstUserAuthentication (iOS 5)

NSFileProtectionNone (default)

Impact on Apps In Background

UIApplicationProtectedDataWillBecomeUnavailable

Legacy Items Default to “None”

Keychain Overview

Multiple Accessibility Options

kSecAttrAccessibleWhenUnlocked (same as Complete)

kSecAttrAccessibleAfterFirstUnlock (default)

kSecAttrAccessibleAlways (same as None)

Can Mark As Non-Migratable

use …ThisDeviceOnly versions

Legacy Items Default to “Always”

Protect OTA Communications

Use TLS or SSL When Appropriate

Most Encryption Requires
Export Compliance (even SSL)
IANAL but: bit.ly/cocoaconf-ssl

Assume Requests From Client
May Not be Worthy of Trust

Consider Public Key Encrypting
Request/Response Contents

Beware of Predictable URLs

Validate URLs Passed to App Via Launch

Consider Malicious Requests

To Local Application

To Server

Don’t Assume A Single User Device

Be Aware of Automatic
iOS Screenshots

Conclusion

Balance App Security with Usability

Assume Transparency to your Resources

Protect All Aspects of App

Assume Threats from Multiple Sources

Randy Beiter – Plugh Studios, Ltd.
randy@plughstudios.com