

Biometrics & Secure Storage in iOS

Jason Shapiro, Intertech

A Training Division Presentation ▶▶

Welcome

- Jason S. Shapiro – jshapiro@intertech.com
 - Intertech Blog: <http://www.intertech.com/blog>
 - LinkedIn: <http://linkedin.com/in/jshapiro>
 - Stackoverflow: <http://stackoverflow.com/users/1704300/j-shapiro>
- Director of Training, Senior Instructor, and Software Architect
- 20+ Years of Professional Software Development and Architecture Experience (Java EE, iOS, Web, Agile, etc.)

Intertech

Instructors Who Consult, Consultants Who Teach

Training

- Training through hands-on, real world business examples.
- Our site, your site, or live online – globally.
- Agile/Scrum, Citrix, VMware, Oracle, IBM, Microsoft, Java/Open Source, and web and mobile technologies.

Consulting

- Design and develop software that powers businesses and governments of all sizes.
- On-site consulting, outsourcing, and mentoring.
- Agile, .NET, Java, SQL Server, mobile development including iPhone and Android platforms and more....

Our Company

- Over 45 awards for growth, innovation and workplace best practices.
- 99.7% satisfaction score from our consulting and training customers.
- Yearly “Best Places to Work” winner in Minnesota.

Welcome

■ About this Presentation:

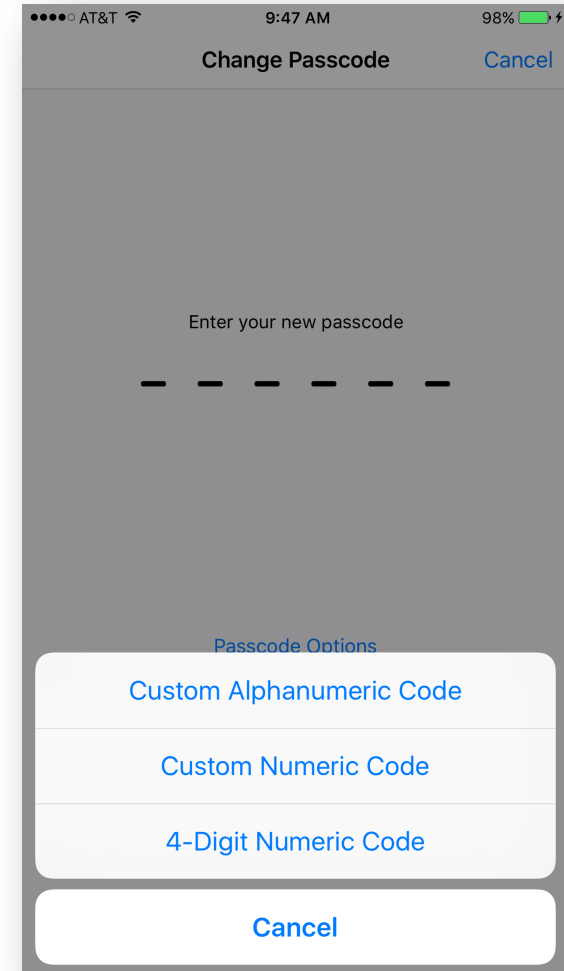
- This is an intermediate level presentation.
- The assumption is that you have experience working with iOS (either Objective-C or Swift).
- Inspired by a project at Intertech (Time Entry).
 - Needed to securely store credentials for a RESTful Web Service.
 - More details on this project at:
[http://www.intertech.com/Blog/2015-team-project-introduction-time-entry-app/`](http://www.intertech.com/Blog/2015-team-project-introduction-time-entry-app/)
- Code examples for this presentation are available at:
<https://github.com/IntertechInc/keychain-touchid-demo.git>

Agenda

- Device Security
- Touch ID (Biometrics)
- Local Authentication
- Keychain Overview
- Keychain Items (CRUD)
- Sharing Keychain Items Between Apps
- Recap
- Resources
- Questions

Device Security

- First and foremost: Enable Passcode!
 - Many security features rely on the passcode to either enable or strengthen their protection.
 - For example: “Data Protection” on the device uses the passcode to create a temporary symmetric key that is thrown away when the device is locked.
- Brute Force Attacks
 - Device can be configured to erase data after 10 failed attempts.
 - After half a dozen failed attempts, the device adds a delay.



Touch ID (Biometrics)

- Announced September 2013; first available on the iPhone 5S.
- The home button doubles as a fingerprint sensor (500 ppi).
- Fingerprints are stored and evaluated in lieu of a passcode.
- However, a passcode IS required for Touch ID to be enabled.

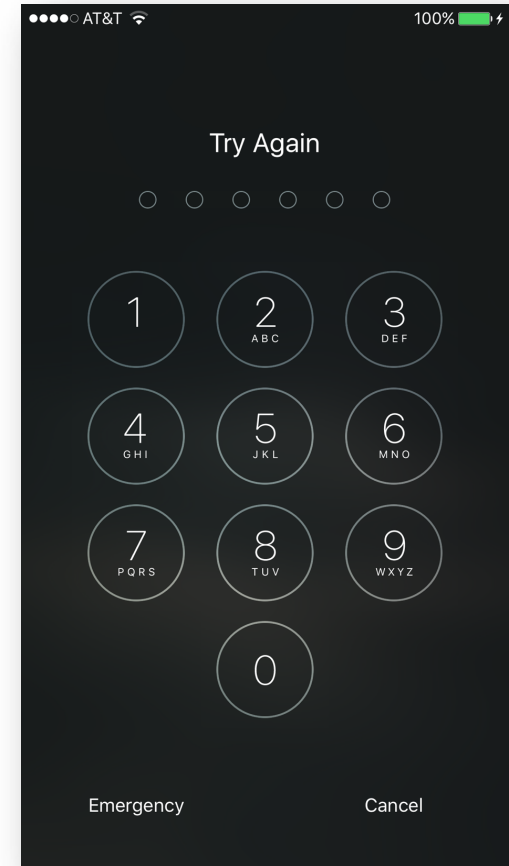


Touch ID (Biometrics)

- Applications do not have direct access to Touch ID or any stored fingerprint data.
 - Application calls are proxied by Security APIs which only return Boolean or Success/Error messages.
- Fingerprints are processed with a “lossy” algorithm prior to persistence in the Secure Enclave.
 - This prevents the data from being used to reconstruct (and steal) a fingerprint.
 - Does reduce the security – 1 in 50,000 chance of having a fingerprint “collision.”

Touch ID (Biometrics)

- Use of the passcode is required in the following situations:
 - The device is rebooted.
 - The user makes 5 unsuccessful Touch ID attempts.
 - The device has been in an unlocked state for more than 48 hours.
 - The device receives a “remote lock” request (www.icloud.com).
 - Modifying Touch ID & Passcode settings.



Touch ID (Biometrics)

- A Touch ID challenge is launched through two different ways:
 - A request is sent from an application to LAContext.
 - An application attempts to retrieve a Keychain Item that has an ACL requirement (User Must Be Present).
- Not without criticism:
 - Many in security warn against using a secret key that never changes (i.e. your fingerprint).
 - Especially as a key for multiple systems and/or resources.
 - Once it's compromised... that's it!
- Successfully hacked by Starbug of the Chaos Computer Club (Berlin) - <https://vimeo.com/75324765>

Local Authentication

- Proxy for an application to request a Touch ID challenge from Secure Enclave.
 - If Touch ID is not available, it defaults to an application determined passcode (iOS 8), or the user's passcode (iOS 9).
 - Cannot be started while an application is in the background.
 - Although it sends requests to the Secure Enclave to provide authentication challenges, there is no API to access secret information stored by the Enclave.

Local Authentication

- The process to request an authentication challenge:
 1. Check to see if the user has the desired challenge type (Touch ID, Passcode) enabled.
 2. If they do, present the authentication challenge.
 3. A return code tells the client whether or not the challenge was successful.

Local Authentication Demo

Keychain & Keychain Items

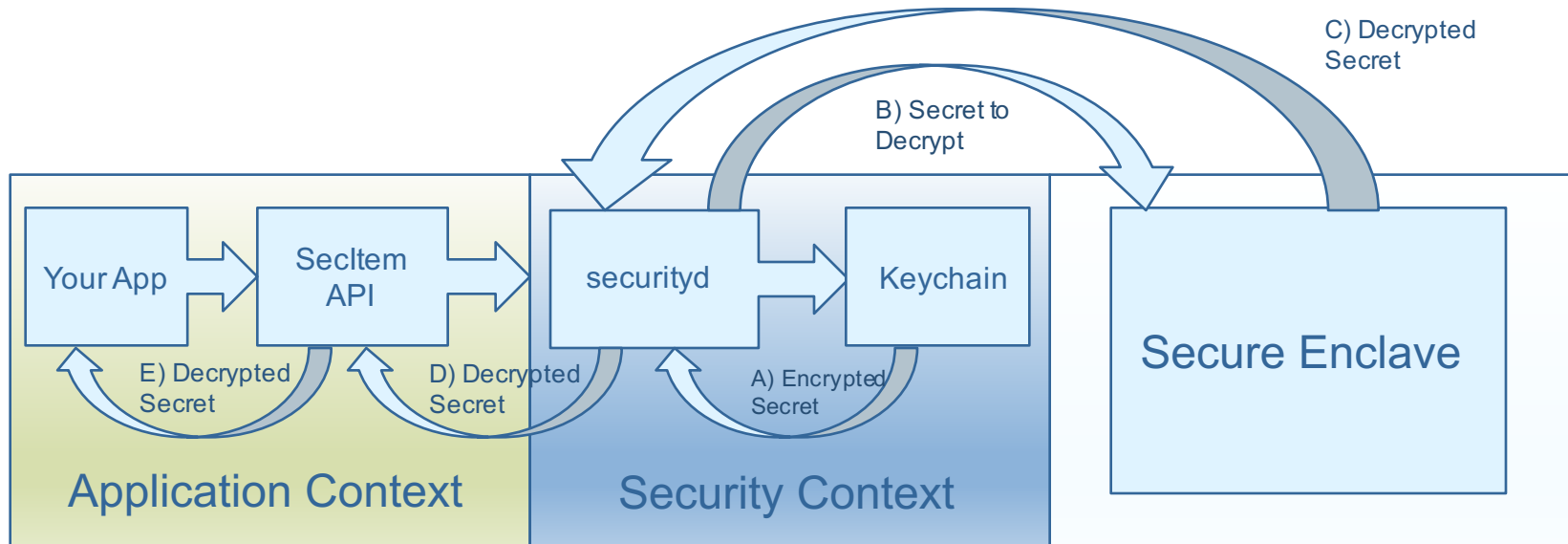
- Keychain
 - A single, encrypted SQLite database that is shared among all application on an iOS device.
 - Keychain is only available when a device is unlocked.
 - Protected, in part, by a “device secret” which is necessary to decrypt the entire Keychain.
 - Used to store small pieces of sensitive information: keys, certificates, notes, etc.
 - Each of these pieces of information are referred to as a “Keychain Item.”

Keychain & Keychain Items

- Keychain Items
 - A Keychain Item is essentially a row in the database.
 - Each Keychain Item has searchable, but hashed, “attributes.”
 - The specific Keychain Item class chosen determines what attributes are configurable (i.e. Generic Password vs. Internet Password).
 - The “secret” of the Keychain Item is encrypted with AES 128 in Galois/Counter Mode
 - Unless explicitly shared through “Entitlement” settings, a Keychain Item is only available to the application that created it.
 - A Keychain Item can be further secured through ACLs – requiring the user to be present before retrieving the key (Touch ID challenge).

Keychain Items (CRUD)

- API Access
 - Requires Security.framework
 - Items accessed or manipulated through SecItem APIs



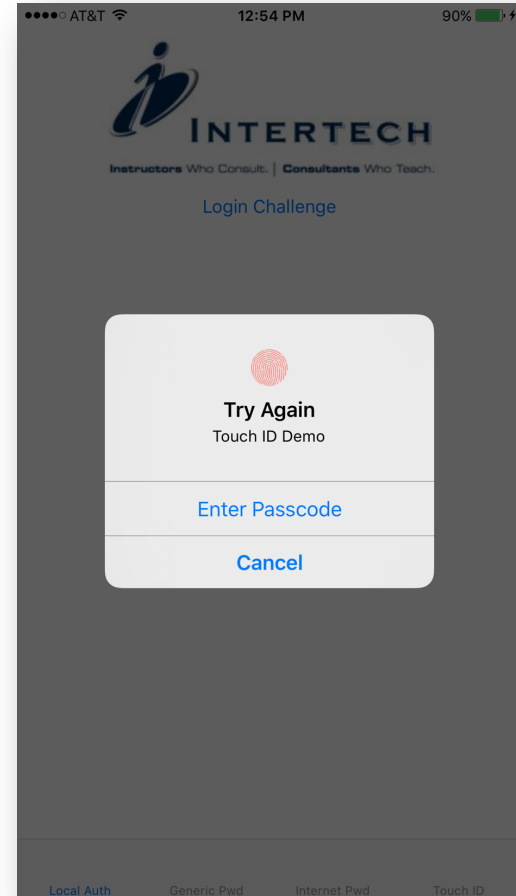
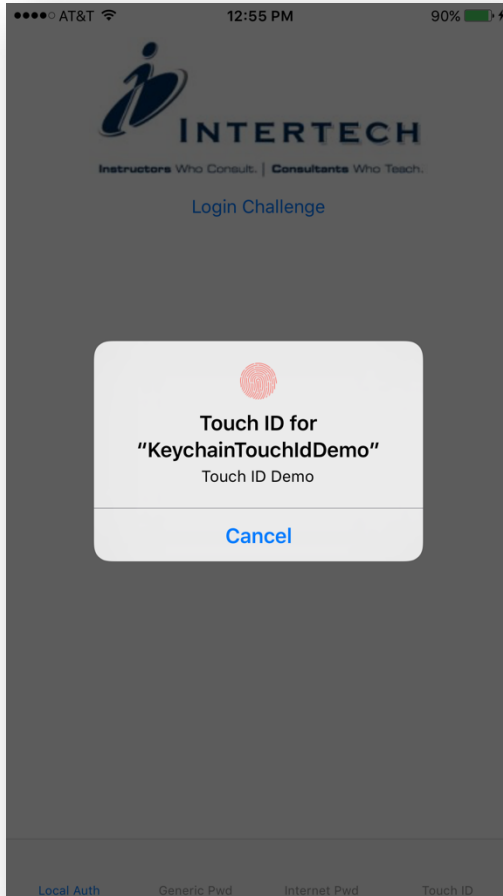
Keychain Item (CRUD)

- Four CRUD methods:
 - SecItemCopyMatching
 - SecItemAdd
 - SecItemUpdate
 - SecItemDelete
- Apple encourages the use of SecItemUpdate rather than a successive SecItemDelete -> SecItemAdd.
 - SecItemUpdate maintains the list of attributes.

Keychain Item (CRUD)

- Five Keychain Item classes:
 - `kSecClassGenericPassword`
(Account, Service, etc.)
 - `kSecClassInternetPassword`
(Server, Protocol, Port, Path, etc.)
 - `kSecClassCertificate`
(Issuer, CertificateType, PublicKeyHash, etc.)
 - `kSecClassKey`
(KeyType, KeySizeInBits, CanSign, etc.)
 - `kSecClassIdentity`
(Key, Certificate)

Keychain Item (CRUD) Demo



Sharing Keychain Items Between Apps

- Keychain Items can be shared by applications that are made by the same development team.
- The process for sharing is:
 - You need to create two "explicit id" provisioning profiles (one for each target).
 - The App ID for both profiles must use the SAME AppID-prefix, though the bundle IDs will be unique, by matching the bundle id specified in each target.
 - For example if X123X is the AppID-Prefix...
App Id #1: X123X.com.intertech.KeychainTouchIdDemo
 - App Id #2: X123X.com.intertech.KeychainItemSharingDemo

Sharing Keychain Items Between Apps

App ID Description

Name:

You cannot use special characters such as @, &, *, ', "

App ID Prefix

Value: (Team ID) ▾

App ID Suffix

Explicit App ID

If you plan to incorporate app services such as Game Center, In-App Purchase, Data Protection, and iCloud, or want a provisioning profile unique to a single app, you must register an explicit App ID for your app.

To create an explicit App ID, enter a unique string in the Bundle ID field. This string should match the Bundle ID of your app.

Bundle ID:

We recommend using a reverse-domain name style string (i.e., com.domainname.appname). It cannot contain an asterisk (*).

Sharing Keychain Items Between Apps



Select App ID.

If you plan to use services such as Game Center, In-App Purchase, and Push Notifications, or want a Bundle ID unique to a single app, use an explicit App ID. If you want to create one provisioning profile for multiple apps or don't need a specific Bundle ID, select a wildcard App ID. Wildcard App IDs use an asterisk (*) as the last digit in the Bundle ID field. Please note that iOS App IDs and Mac App IDs cannot be used interchangeably.

App ID:

Sharing Keychain Items Between Apps

- The process for sharing is (continued):
 - Make sure each target is set to use the appropriate Provisioning Profile
 - Turn on the Keychain Sharing entitlement in each app (Target -> Capabilities), specifying the same group in each.
 - For example: com.intertech.keysharing
 - Add the following item to the attribute dictionary in kSecAttrAccessGroup : "<APPID-PREFIX>.<SHARING-GROUP>"
 - For example:
kSecAttrAccessGroup : "X123X.com.intertech.keysharing"

Sharing Keychain Items Demo

Recap

- Device Security
- Touch ID (Biometrics)
- Local Authentication
- Keychain Overview
- Keychain Items (CRUD)
- Sharing Keychain Items Between Apps

Resources

- **App Distribution Guide – Adding Capabilities**
<https://developer.apple.com/library/ios/documentation/IDEs/Conceptual/AppDistributionGuide/AddingCapabilities/AddingCapabilities.html>
- **iOS Security (iOS 8.3 or Later), June 2015**
https://www.apple.com/business/docs/iOS_Security_Guide.pdf
- **Keychain Services Programming Guide**
<https://developer.apple.com/library/ios/documentation/Security/Conceptual/keychainServConcepts>
- **Keychain Services Reference**
<https://developer.apple.com/library/ios/documentation/Security/Reference/keychainservices/index.html>
- **Security Overview**
https://developer.apple.com/library/ios/documentation/Security/Conceptual/Security_Overview

Thank You

Questions or Feedback?

Jason Shapiro – jshapiro@intertech.com