OWASP FOUNDATION PRESENTS

APPSEC USA

NOVEMBER 18 - 21
NY MARRIOTT MARQUIS, NYC
2013

Contain Yourself:
Building Mobile Secure Containers

Ron Gutierrez
Gotham Digital Science (GDS)

Outline

- What are Security Containers?
- How are Secure Containers Created?
- Authentication Design Patterns and Data Encryption
- Assessing the Strength of a Secure Container
- Limitations of Secure Containers

Outline

What are Security Containers?

How are Secure Containers Created?

Authentication Design Patterns and Data Encryption

Assessing the Strength of a Secure Container

Limitations of Secure Containers

Bring Your Own Device (BYOD)

- Enterprises are embracing "Bring Your Own Device" (BYOD)
- Employees use personal devices to access company resources
- Unlike managed devices, device policies **cannot** be enforced

Why BYOD?

What are Secure Containers?

- ✓ Data storage protection performed at the application level
- ✓ Does not rely on OS security features being activated
- ✓ Allows security policies to be enforced at the application level

Commercial Solutions

Good

Enter password

Forgot password?

CITRIX®

NUKONA

mocana®

securityevolves™

and many more……

Why Secure Containers?

- Orgs want employees to have **convenient** access to sensitive resources (email, documents, apps with sensitive data, etc)
- Allows them to have control of their data on unmanaged devices

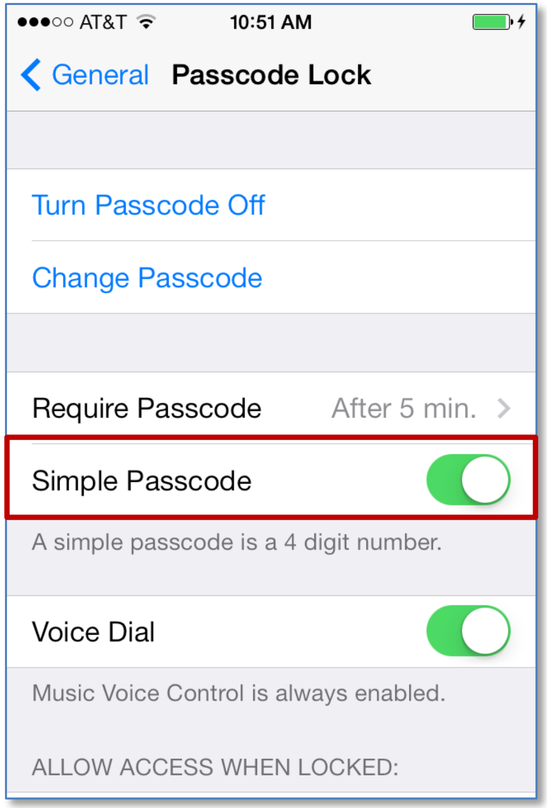
Why Secure Containers?

Unmanaged Devices

- Organizations **cannot** enforce
  - Device is passcode protected (Data Protection)
  - Device Passcode Policies
  - Remote Wipes
  - Device is not Jailbroken

Data Protection (DP)

- Developers can opt-in to use DP APIs
- Must crack device passcode to access data



Require Passcode: After 5 min.

Simple Passcode: On

A simple passcode is a 4 digit number.

Voice Dial: On

Music Voice Control is always enabled.

ALLOW ACCESS WHEN LOCKED:

Outline

What are Security Containers?

How are Secure Containers Created?

Authentication Design Patterns and Data Encryption

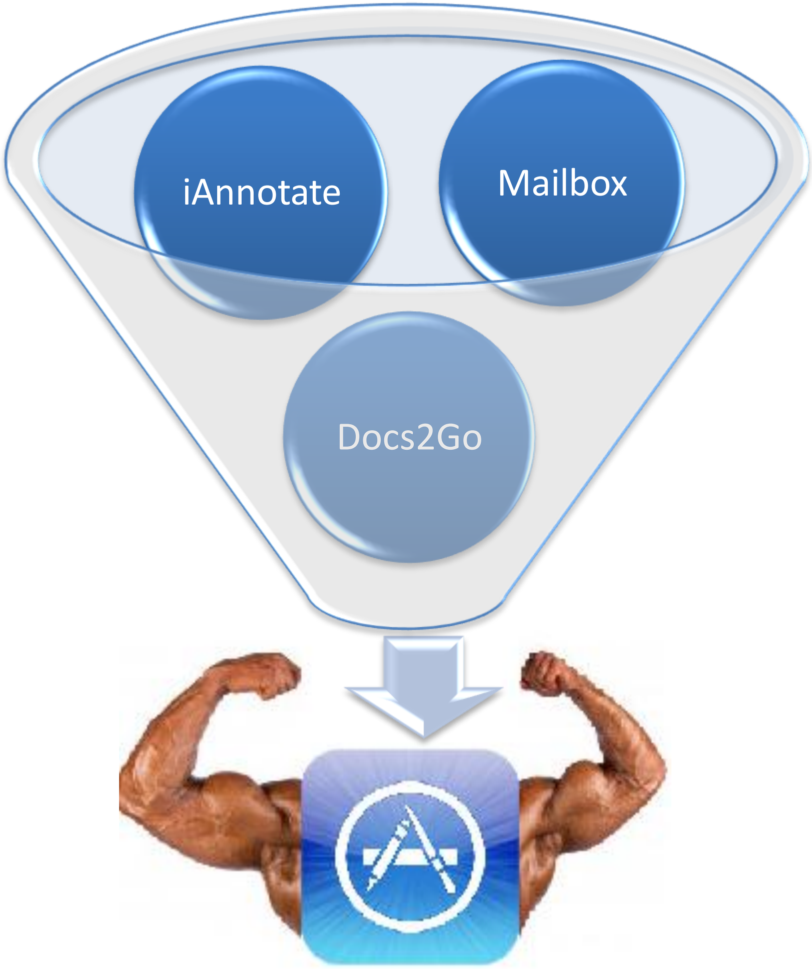
Assessing the Strength of a Secure Container

Limitations of Secure Containers

How Are Secure Containers Made?

- Application Wrapping
- Functionality injected into existing applications
- Enforces security at the application level
  - Data encryption at rest
  - Authentication
  - Policy enforcement
- No code changes required by developer

Application Wrapping



iOS App Wrapping Analysis

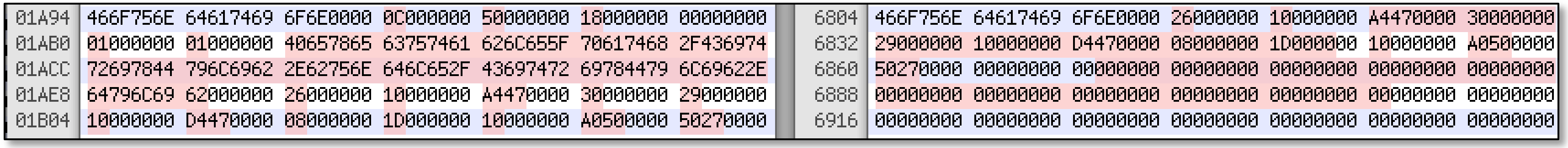
Citrix Cloud Gateway MDX Application Wrapping Analysis

- Tool accepts IPA files
- Application is re-signed using Distribution Certificate
- Outputs a new wrapped IPA file

Let's analyze the output

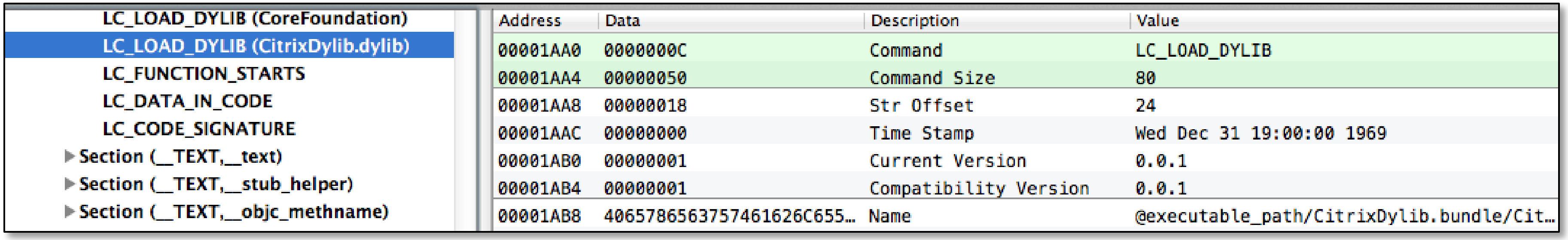
iOS App Wrapping Analysis

Diffing a pre-wrapped and post-wrapped iOS binary with HexFiend



```
01A94  466F756E 64617469 6F6E0000 0C000000 50000000 18000000 00000000     6804  466F756E 64617469 6F6E0000 26000000 10000000 A4470000 30000000
01AB0  01000000 01000000 40657865 63757461 626C655F 70617468 2F436974     6832  29000000 10000000 D4470000 08000000 1D000000 10000000 A0500000
01ACC  72697844 796C6962 2E62756E 646C652F 43697472 69784479 6C69622E     6860  50270000 00000000 00000000 00000000 00000000 00000000 00000000
01AE8  64796C69 62000000 26000000 10000000 A4470000 30000000 29000000     6888  00000000 00000000 00000000 00000000 00000000 00000000 00000000
01B04  10000000 D4470000 08000000 1D000000 10000000 A0500000 50270000     6916  00000000 00000000 00000000 00000000 00000000 00000000 00000000
```

View address offset with MachOView tool to see what was changed



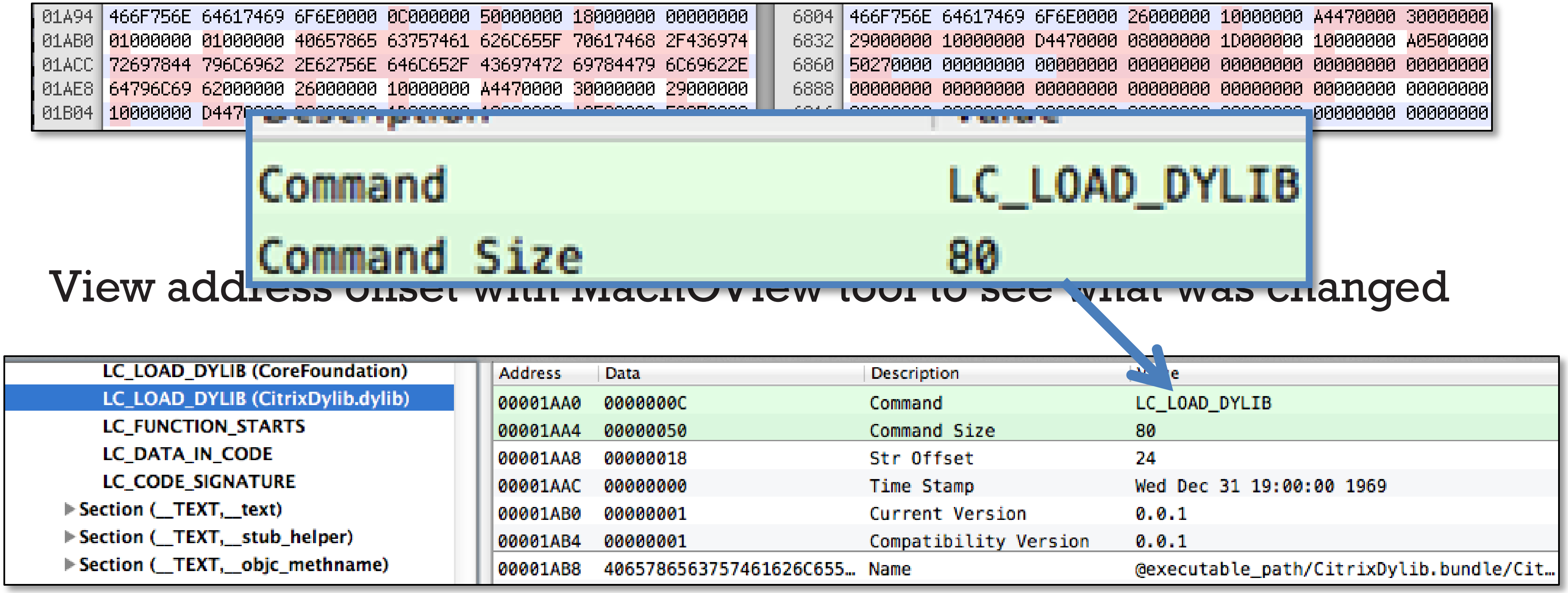
LC_LOAD_DYLIB (CoreFoundation)
LC_LOAD_DYLIB (CitrixDylib.dylib)
LC_FUNCTION_STARTS
LC_DATA_IN_CODE
LC_CODE_SIGNATURE
▶ Section (__TEXT,__text)
▶ Section (__TEXT,__stub_helper)
▶ Section (__TEXT,__objc_methname)

Address	Data	Description	Value
00001AA0	0000000C	Command	LC_LOAD_DYLIB
00001AA4	00000050	Command Size	80
00001AA8	00000018	Str Offset	24
00001AAC	00000000	Time Stamp	Wed Dec 31 19:00:00 1969
00001AB0	00000001	Current Version	0.0.1
00001AB4	00000001	Compatibility Version	0.0.1
00001AB8	4065786563757461626C655…	Name	@executable_path/CitrixDylib.bundle/Cit…

A LC_LOAD_DYLIB is added to the App's Mach-O Load Commands

iOS App Wrapping Analysis

Diffing a pre-wrapped and post-wrapped iOS binary with HexFiend



```
01A94  466F756E 64617469 6F6E0000 0C000000 50000000 18000000 00000000
01AB0  01000000 01000000 40657865 63757461 626C655F 70617468 2F436974
01ACC  72697844 796C6962 2E62756E 646C652F 43697472 69784479 6C69622E
01AE8  64796C69 62000000 26000000 10000000 A4470000 30000000 29000000
01B04  10000000 D447
```

```
6804  466F756E 64617469 6F6E0000 26000000 10000000 A4470000 30000000
6832  29000000 10000000 D4470000 08000000 1D000000 10000000 A0500000
6860  50270000 00000000 00000000 00000000 00000000 00000000 00000000
6888  00000000 00000000 00000000 00000000 00000000 00000000 00000000
                                                   00000000 00000000
```

Command	LC_LOAD_DYLIB
Command Size	80

View address offset with MachOView tool to see what was changed

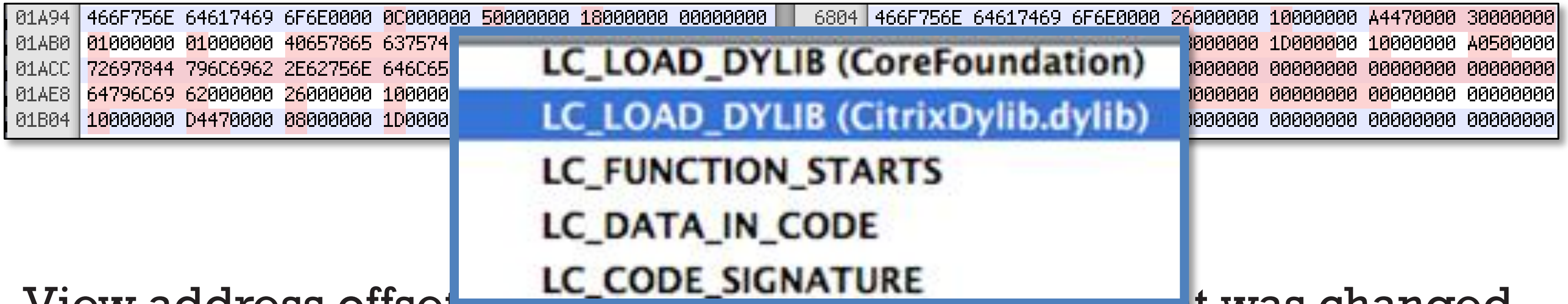
- LC_LOAD_DYLIB (CoreFoundation)
- LC_LOAD_DYLIB (CitrixDylib.dylib)
- LC_FUNCTION_STARTS
- LC_DATA_IN_CODE
- LC_CODE_SIGNATURE
- ▶ Section (__TEXT,__text)
- ▶ Section (__TEXT,__stub_helper)
- ▶ Section (__TEXT,__objc_methname)

Address	Data	Description	Value
00001AA0	0000000C	Command	LC_LOAD_DYLIB
00001AA4	00000050	Command Size	80
00001AA8	00000018	Str Offset	24
00001AAC	00000000	Time Stamp	Wed Dec 31 19:00:00 1969
00001AB0	00000001	Current Version	0.0.1
00001AB4	00000001	Compatibility Version	0.0.1
00001AB8	4065786563757461626C655…	Name	@executable_path/CitrixDylib.bundle/Cit…

A LC_LOAD_DYLIB is added to the App's Mach-O Load Commands

iOS App Wrapping Analysis

Diffing a pre-wrapped and post-wrapped iOS binary with HexFiend

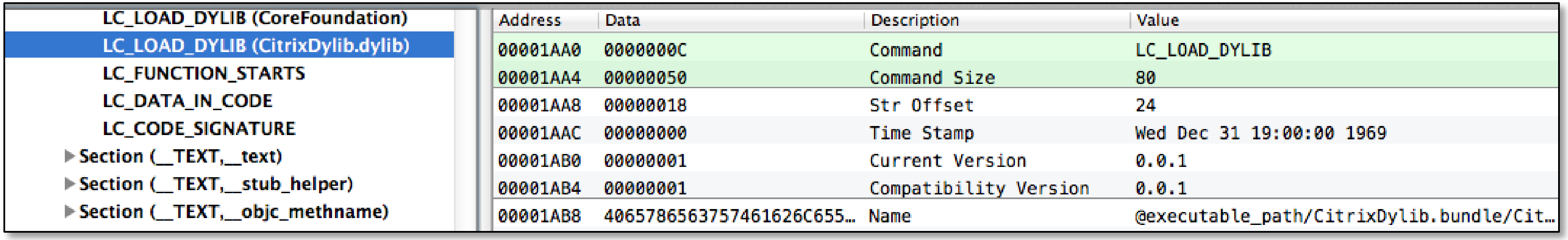


```
01A94  466F756E 64617469 6F6E0000 0C000000 50000000 18000000 00000000
01AB0  01000000 01000000 40657865 637574
01ACC  72697844 796C6962 2E62756E 646C65
01AE8  64796C69 62000000 26000000 100000
01B04  10000000 D4470000 08000000 1D0000
```

```
6804  466F756E 64617469 6F6E0000 26000000 10000000 A4470000 30000000
      000000 1D000000 10000000 A0500000
      000000 00000000 00000000 00000000
      000000 00000000 00000000 00000000
      000000 00000000 00000000 00000000
```

LC_LOAD_DYLIB (CoreFoundation)
LC_LOAD_DYLIB (CitrixDylib.dylib)
LC_FUNCTION_STARTS
LC_DATA_IN_CODE
LC_CODE_SIGNATURE

View address offset with MachOView tool to see what was changed

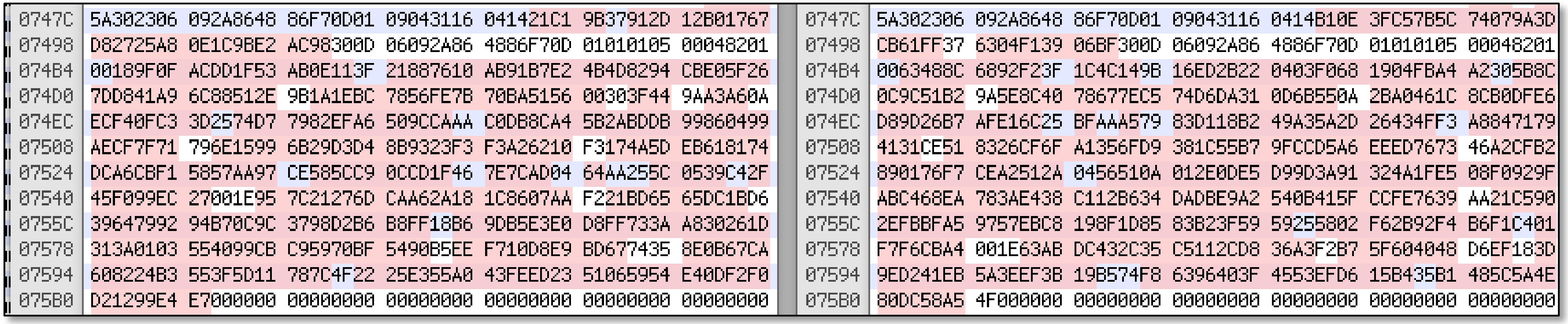


- LC_LOAD_DYLIB (CoreFoundation)
- LC_LOAD_DYLIB (CitrixDylib.dylib)
- LC_FUNCTION_STARTS
- LC_DATA_IN_CODE
- LC_CODE_SIGNATURE
- ▶ Section (__TEXT,__text)
- ▶ Section (__TEXT,__stub_helper)
- ▶ Section (__TEXT,__objc_methname)

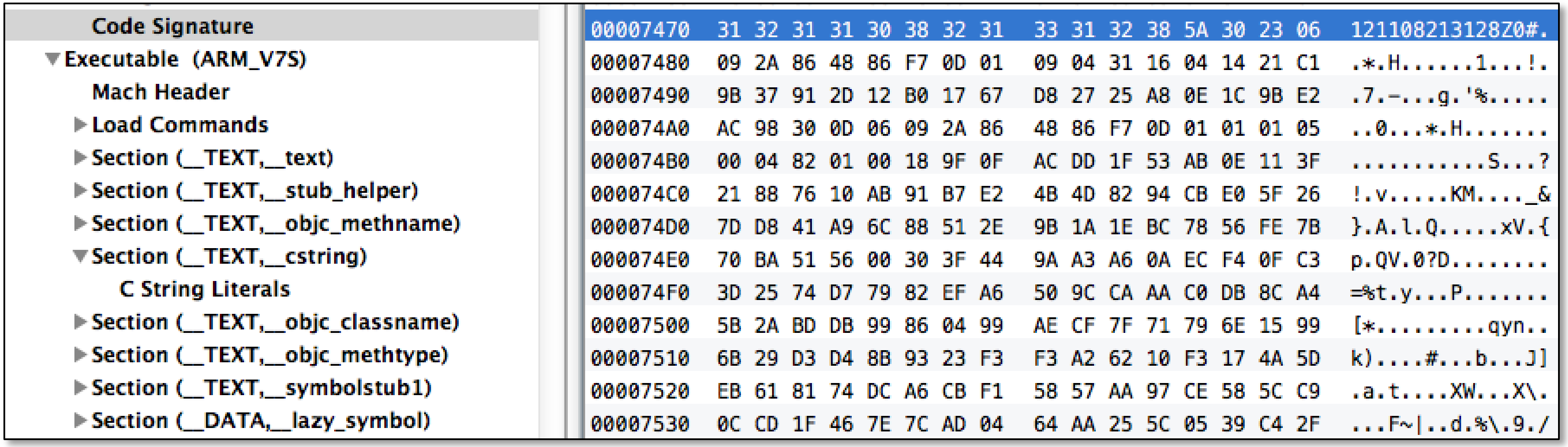
Address	Data	Description	Value
00001AA0	0000000C	Command	LC_LOAD_DYLIB
00001AA4	00000050	Command Size	80
00001AA8	00000018	Str Offset	24
00001AAC	00000000	Time Stamp	Wed Dec 31 19:00:00 1969
00001AB0	00000001	Current Version	0.0.1
00001AB4	00000001	Compatibility Version	0.0.1
00001AB8	40657865637574617461626C655…	Name	@executable_path/CitrixDylib.bundle/Cit…

A LC_LOAD_DYLIB is added to the App's Mach-O Load Commands

iOS App Wrapping Analysis



```
0747C  5A302306 092A8648 86F70D01 09043116 041421C1 9B37912D 12B01767    0747C  5A302306 092A8648 86F70D01 09043116 0414B10E 3FC57B5C 74079A3D
07498  D82725A8 0E1C9BE2 AC98300D 06092A86 4886F70D 01010105 00048201    07498  CB61FF37 6304F139 06BF300D 06092A86 4886F70D 01010105 00048201
074B4  00189F0F ACDD1F53 AB0E113F 21887610 AB91B7E2 4B4D8294 CBE05F26    074B4  0063488C 6892F23F 1C4C149B 16ED2B22 0403F068 1904FBA4 A2305B8C
074D0  7DD841A9 6C88512E 9B1A1EBC 7856FE7B 70BA5156 00303F44 9AA3A60A    074D0  0C9C51B2 9A5E8C40 78677EC5 74D6DA31 0D6B550A 2BA0461C 8CB0DFE6
074EC  ECF40FC3 3D2574D7 7982EFA6 509CCAAA C0DB8CA4 5B2ABDDB 99860499    074EC  D89D26B7 AFE16C25 BFAAA579 83D118B2 49A35A2D 26434FF3 A8847179
07508  AECF7F71 796E1599 6B29D3D4 8B9323F3 F3A26210 F3174A5D EB618174    07508  4131CE51 8326CF6F A1356FD9 381C55B7 9FCCD5A6 EEED7673 46A2CFB2
07524  DCA6CBF1 5857AA97 CE585CC9 0CCD1F46 7E7CAD04 64AA255C 0539C42F    07524  890176F7 CEA2512A 0456510A 012E0DE5 D99D3A91 324A1FE5 08F0929F
07540  45F099EC 27001E95 7C21276D CAA62A18 1C8607AA F221BD65 65DC1BD6    07540  ABC468EA 783AE438 C112B634 DADBE9A2 540B415F CCFE7639 AA21C590
0755C  39647992 94B70C9C 3798D2B6 B8FF18B6 9DB5E3E0 D8FF733A A830261D    0755C  2EFBBFA5 9757EBC8 198F1D85 83B23F59 59255802 F62B92F4 B6F1C401
07578  313A0103 554099CB C95970BF 5490B5EE F710D8E9 BD677435 8E0B67CA    07578  F7F6CBA4 001E63AB DC432C35 C5112CD8 36A3F2B7 5F604048 D6EF183D
07594  608224B3 553F5D11 787C4F22 25E355A0 43FEED23 51065954 E40DF2F0    07594  9ED241EB 5A3EEF3B 19B574F8 6396403F 4553EFD6 15B435B1 485C5A4E
075B0  D21299E4 E7000000 00000000 00000000 00000000 00000000 00000000    075B0  80DC58A5 4F000000 00000000 00000000 00000000 00000000 00000000
```



Code Signature
▼ Executable (ARM_V7S)
Mach Header
▶ Load Commands
▶ Section (__TEXT,__text)
▶ Section (__TEXT,__stub_helper)
▶ Section (__TEXT,__objc_methname)
▼ Section (__TEXT,__cstring)
C String Literals
▶ Section (__TEXT,__objc_classname)
▶ Section (__TEXT,__objc_methtype)
▶ Section (__TEXT,__symbolstub1)
▶ Section (__DATA,__lazy_symbol)

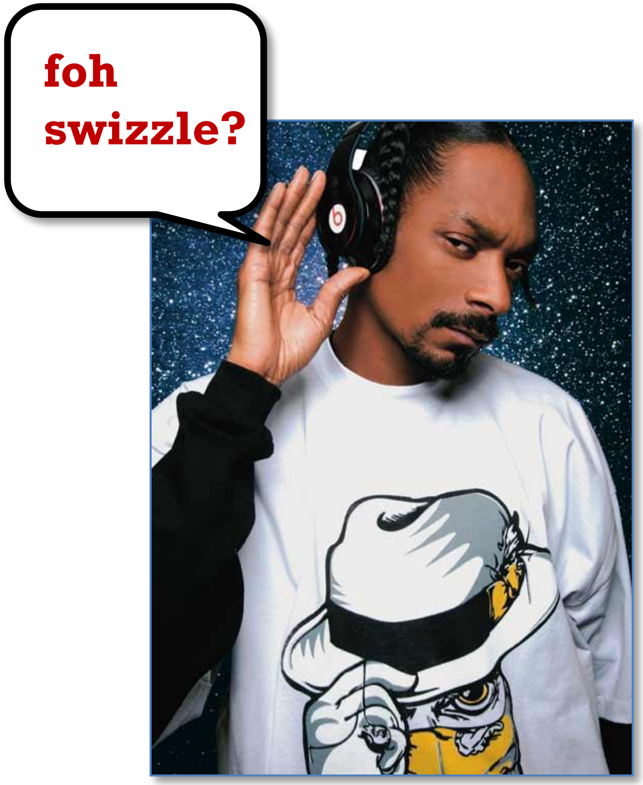
```
00007470  31 32 31 31 30 38 32 31  33 31 32 38 5A 30 23 06  121108213128Z0#.
00007480  09 2A 86 48 86 F7 0D 01  09 04 31 16 04 14 21 C1  .*.H......1...!.
00007490  9B 37 91 2D 12 B0 17 67  D8 27 25 A8 0E 1C 9B E2  .7.-...g.'%.....
000074A0  AC 98 30 0D 06 09 2A 86  48 86 F7 0D 01 01 01 05  ..0...*.H.......
000074B0  00 04 82 01 00 18 9F 0F  AC DD 1F 53 AB 0E 11 3F  ...........S...?
000074C0  21 88 76 10 AB 91 B7 E2  4B 4D 82 94 CB E0 5F 26  !.v.....KM...._&
000074D0  7D D8 41 A9 6C 88 51 2E  9B 1A 1E BC 78 56 FE 7B  }.A.l.Q.....xV.{
000074E0  70 BA 51 56 00 30 3F 44  9A A3 A6 0A EC F4 0F C3  p.QV.0?D........
000074F0  3D 25 74 D7 79 82 EF A6  50 9C CA AA C0 DB 8C A4  =%t.y...P.......
00007500  5B 2A BD DB 99 86 04 99  AE CF 7F 71 79 6E 15 99  [*.........qyn..
00007510  6B 29 D3 D4 8B 93 23 F3  F3 A2 62 10 F3 17 4A 5D  k)....#...b...J]
00007520  EB 61 81 74 DC A6 CB F1  58 57 AA 97 CE 58 5C C9  .a.t....XW...X\.
00007530  0C CD 1F 46 7E 7C AD 04  64 AA 25 5C 05 39 C4 2F  ...F~|..d.%\.9./
```

Updates to the Code Signature of the Binary

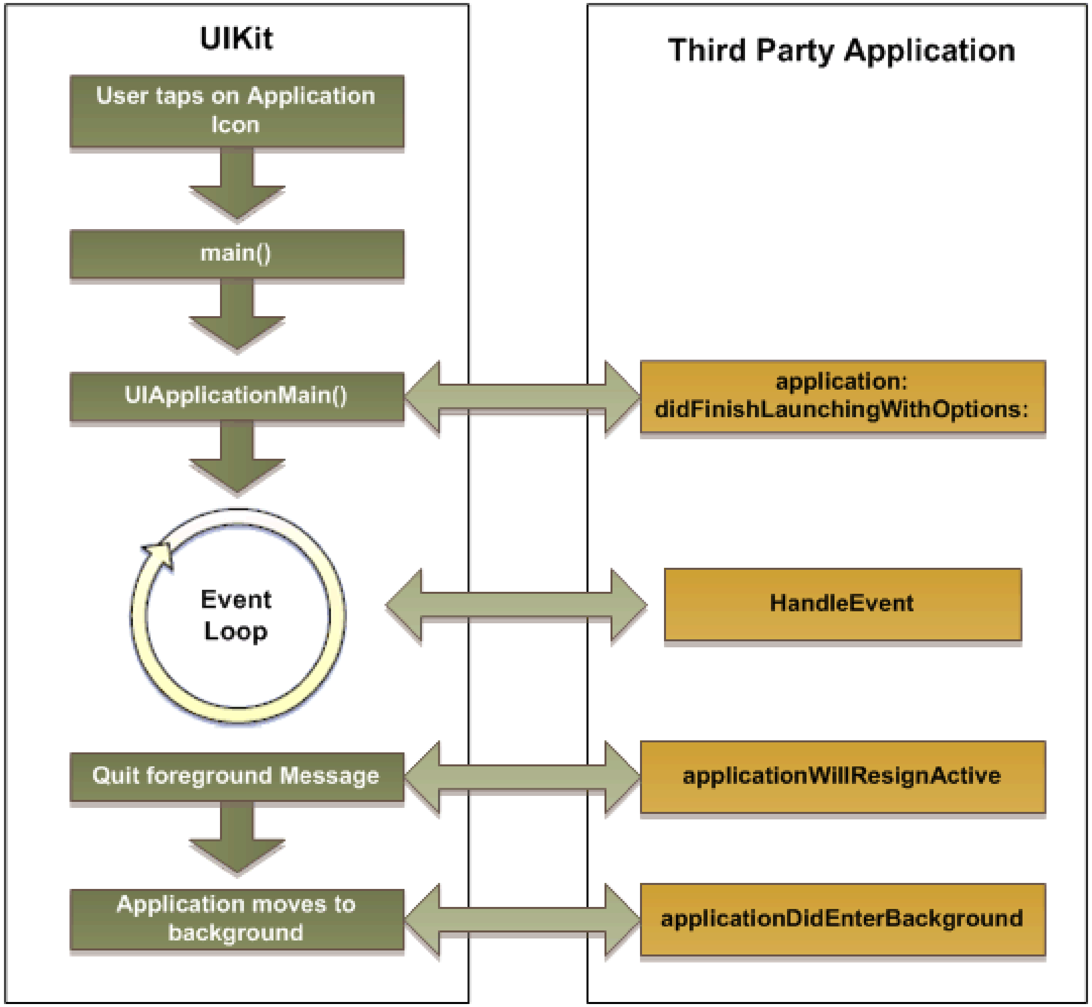
iOS Method Swizzling

- Can modify implementations of iOS Objective-C methods
  - http://cocoadev.com/wiki/MethodSwizzling

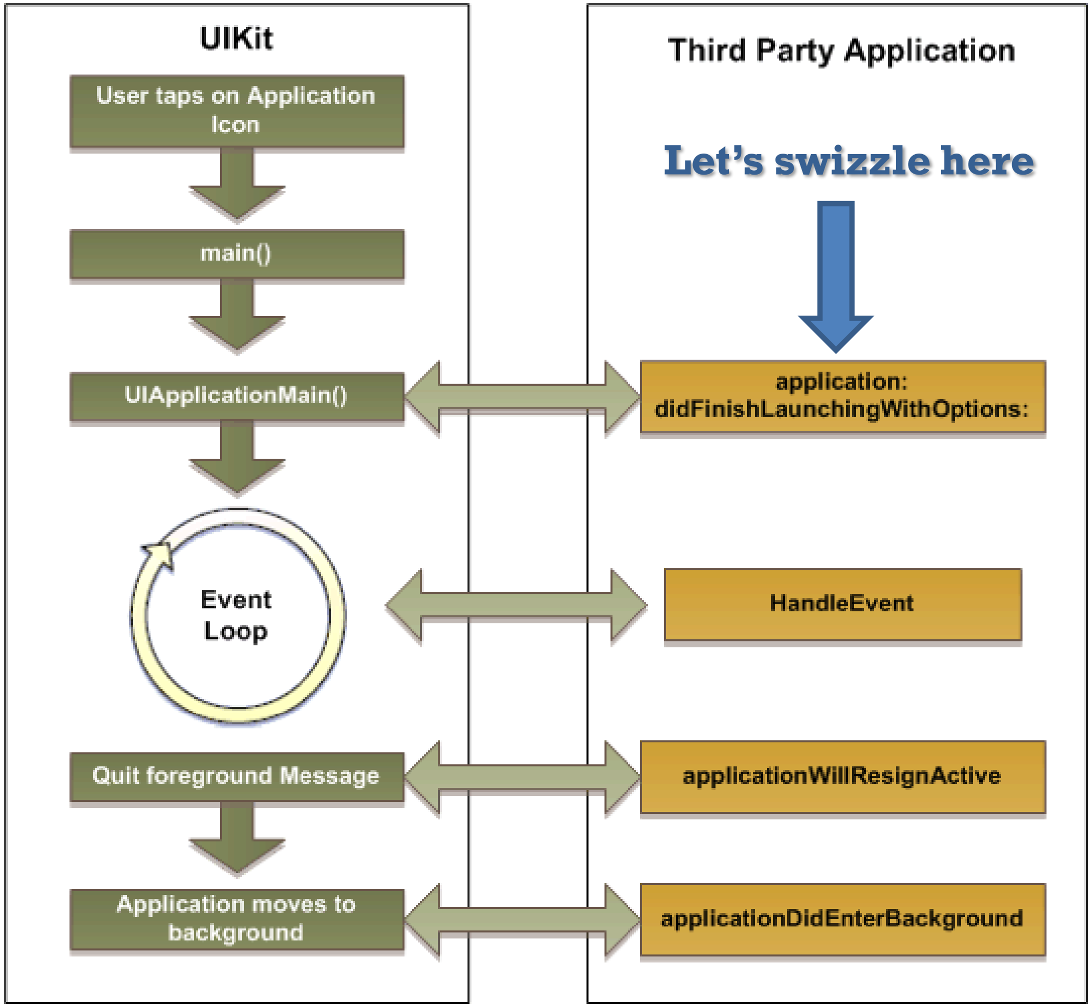
- Seen in Cydia applications
  - MobileSubstrate Tweaks
  - Cycript



iOS App Life Cycle 101


UIKit
User taps on Application Icon
main()
UIApplicationMain()
Event Loop
Quit foreground Message
Application moves to background
Third Party Application
application: didFinishLaunchingWithOptions:
HandleEvent
applicationWillResignActive
applicationDidEnterBackground

iOS App Life Cycle 101


UIKit
User taps on Application Icon
main()
UIApplicationMain()
Event Loop
Quit foreground Message
Application moves to background
Third Party Application
Let's swizzle here
application: didFinishLaunchingWithOptions:
HandleEvent
applicationWillResignActive
applicationDidEnterBackground

Swizzle Early

- Static/Dynamic libraries can overwrite implementations upon startup
- Implement swizzling within load method on Obj-C objects

+(void) load
Invoked whenever a class or category is added to the Objective-C runtime; implement this method to perform class-specific behavior upon loading.

The load message is sent to classes and categories that are both dynamically loaded and statically linked, but only if the newly loaded class or category implements a method that can respond.

https://developer.apple.com/library/ios/documentation/Cocoa/Reference/Foundation/Classes/nsobject_Class/Reference/Reference.html#//apple_ref/occ/clm/NSObject/load

Objective-C Swizzling 101

```
Method original, swizzled;

original = class_getInstanceMethod(class,
@selector(application:didFinishLaunchingWithOptions:));

swizzled = class_getInstanceMethod(self,
@selector(swizzled_application:didFinishLaunchingWithOptions:));

method_exchangeImplementations(original, swizzled);
```

Objective-C Swizzling 101

```
Method original, swizzled;

original = class_getInstanceMethod(class,
@selector(application:didFinishLaunchingWithOptions:));

swizzled = class_getInstanceMethod(self,
@selector(swizzled_application:didFinishLaunchingWithOptions:));

method_exchangeImplementations(original, swizzled);
```

Specify class and method to be replaced

Objective-C Swizzling 101

```
Method original, swizzled;

original = class_getInstanceMethod(class,
@selector(application:didFinishLaunchingWithOptions:));

swizzled = class_getInstanceMethod(self,
@selector(swizzled_application:didFinishLaunchingWithOptions:));

method_exchangeImplementations(original, swizzled);
```

Specify class and method containing your new implementation

So I heard we are gonna get swizzled up in this piece?.. Nah mean?

Contain Yourself: Building Mobile Secure Containers

DEMO: METHOD SWIZZLING

I Can Swizzle.. Now What?

- That was a simple POC on how to implement a secure container solution using a static library
- What now?
  - Org-wide static library can solve the various common iOS security issues
- Apparently there is a market for these things as well

Outline

What are Security Containers?

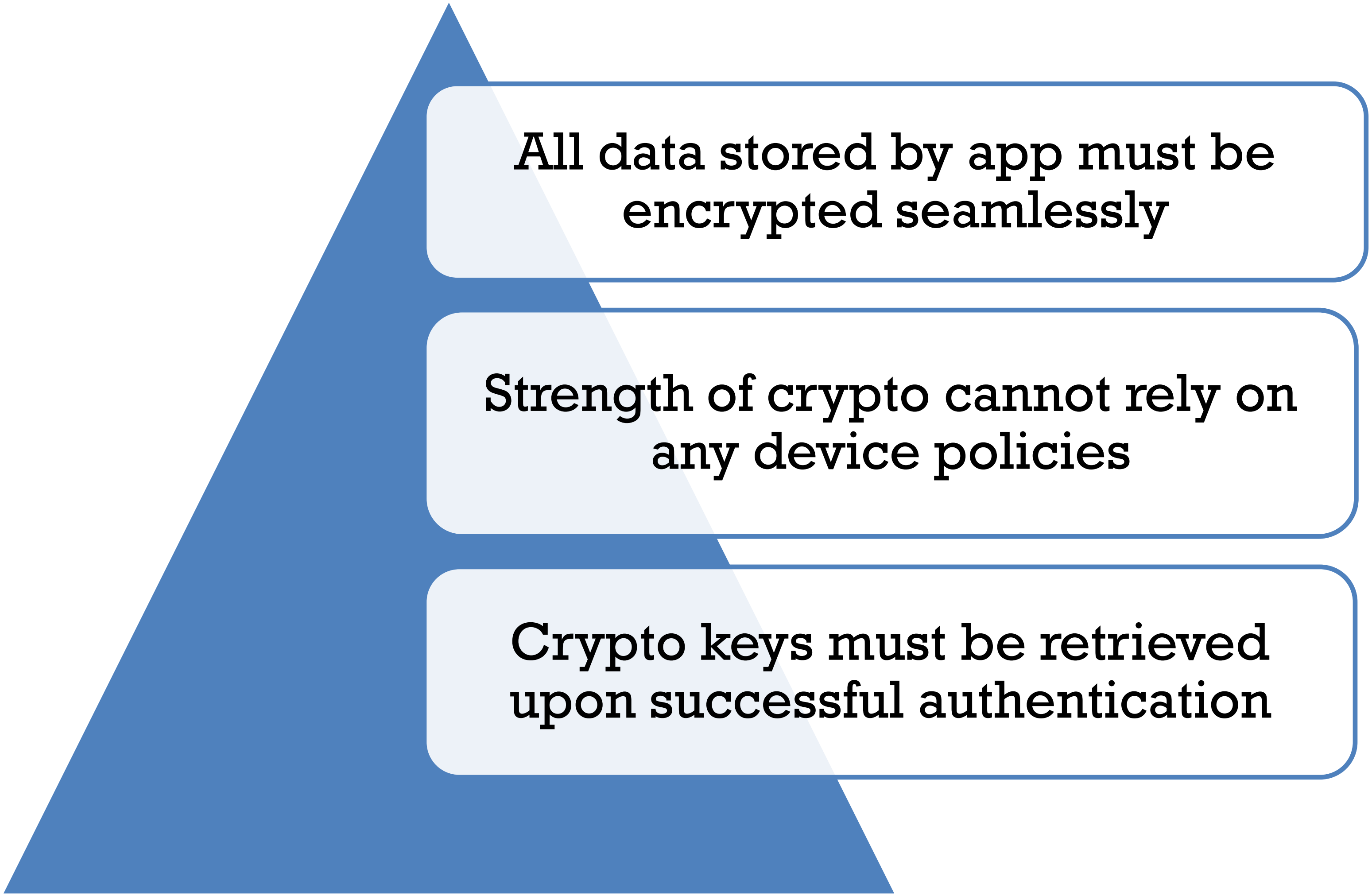
How are Secure Containers Created?

Authentication Design Patterns and Data Encryption

Assessing the Strength of a Secure Container

Limitations of Secure Containers

Principles To Live By



All data stored by app must be encrypted seamlessly

Strength of crypto cannot rely on any device policies

Crypto keys must be retrieved upon successful authentication

Authentication Designs

- Broken By Design
  - Storing crypto key on the device
  - Crypto key derivation material stored on device
  - Data storage not protected by app authentication passcode

Might as well start encrypting with ROT13+1

@YOLOCrypto approved algorithm

Essentially Security By Obscurity

Real World Example

- Mint - Financial Management Application
- Supports passcode protection
- Passcode is not used to protect any application data
- Susceptible to client-side bypass via Runtime Manipulation

Let’s bypass it

Bypassing Mint Pin Screen

- Decrypt AppStore Binary using Clutch
- Run class-dump on the decrypted binary
  - Prints out class information from Mach-O files
- Identify some methods which might control the lock screen

“Mach-O Man” Randy Savage

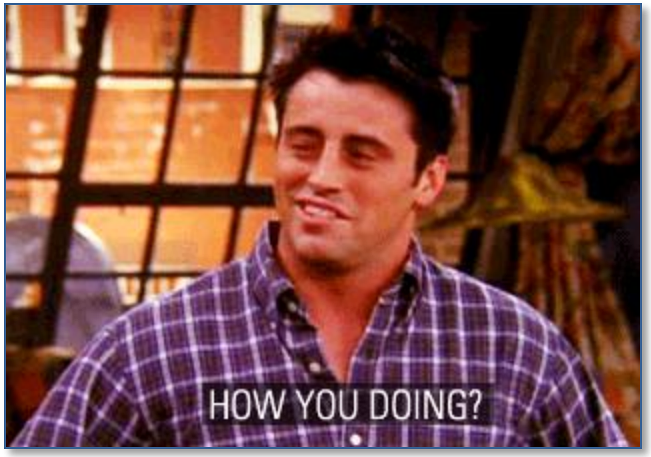
```
@interface GalaAppDelegate : NSObject
<UIApplicationDelegate, WebServiceDelegate,
UIAlertViewDelegate, BWQuincyManagerDelegate>
{
    [..snip..]

+ (id)sharedController;
[..snip..]
- (void)logInWithUsername:(id)arg1 password:(id)arg2;
- (void)logInUsingStoredMintToken;
- (void)popAwayLogin;
- (void)popUpFirstRunView;
- (void)popUpWelcomeView;
- (void)updateStatusString:(id)arg1;
- (void)setStatusCode:(int)arg1;
- (void)popAwayPasscode;
- (void)popUpPasscode;
[..snip..]
```

Mint.app class-dump results snippet

```
@interface GalaAppDelegate : NSObject
<UIApplicationDelegate, WebServiceDelegate,
UIAlertViewDelegate, BWQuincyManagerDelegate>
{
    [..snip..]

+ (id)sharedController;
[..snip..]
- (void)logInWithUsername:(id)arg1 password:(id)arg2;
- (void)logInUsingStoredMintToken;
- (void)popAwayLogin;
- (void)popUpFirstRunView;
- (void)popUpWelcomeView;
- (void)updateStatusString:(id)arg1;
- (void)setStatusCode:(int)arg1;
- (void)popAwayPasscode;
- (void)popUpPasscode;
[..snip..]
```



Mint.app class-dump results snippet

Analyze Function via Mobile Substrate

- Allows you to hook Obj-C methods on any app
  - Uses similar approach as described earlier
- Requires jailbroken device
- I like to use Theos for quick and dirty hooking

Goal is to identify when passcode related methods are called

http://iphonedevwiki.net/index.php/Theos

Simple Theos Tweak to Identify When Calls Are Made

```
%hook GalaAppDelegate
- (void)popAwayLogin {
    %log;
    %orig;
}
- (void)popAwayPasscode {
    %log;
    %orig;
}
- (void)popUpPasscode {
    %log;
    %orig;
}
%end
%ctor {
   NSLog(@"Application is now hooked by RG");
   %init;
}
```

Simple Theos Tweak to Identify When Calls Are Made

```
%hook GalaAppDelegate
- (void)popAwayLogin {
    %log;
    %orig;
}
- (void)popAwayPasscode {
    %log;
    %orig;
}
- (void)popUpPasscode {
    %log;
    %orig;
}
%end
%ctor {
   NSLog(@"Application is now hooked by RG");
   %init;
}
```

Logs method call to console

Calls original method

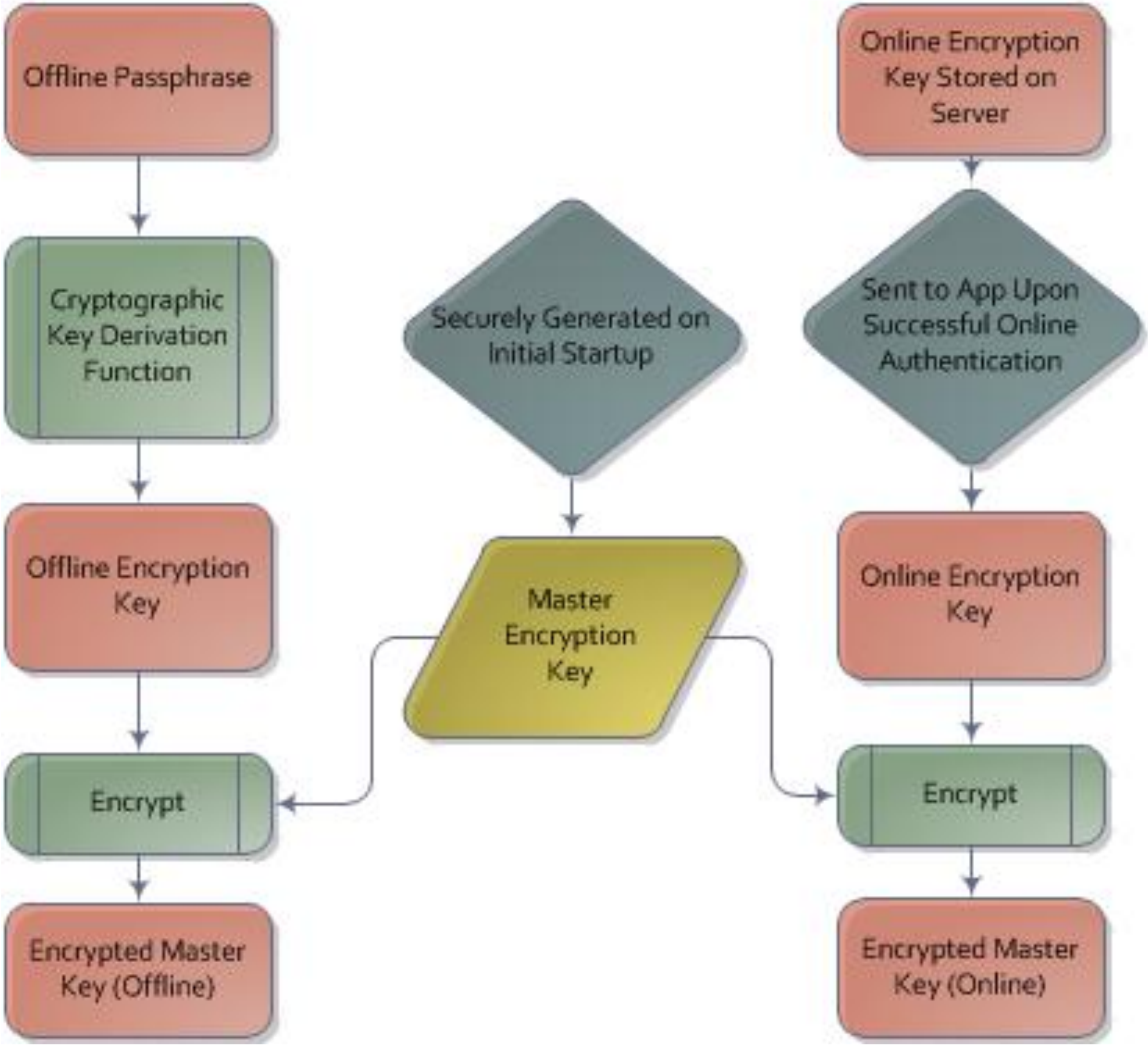
Contain Yourself: Building Mobile Secure Containers

DEMO: USING CYCRIPT TO BYPASS PINCODE

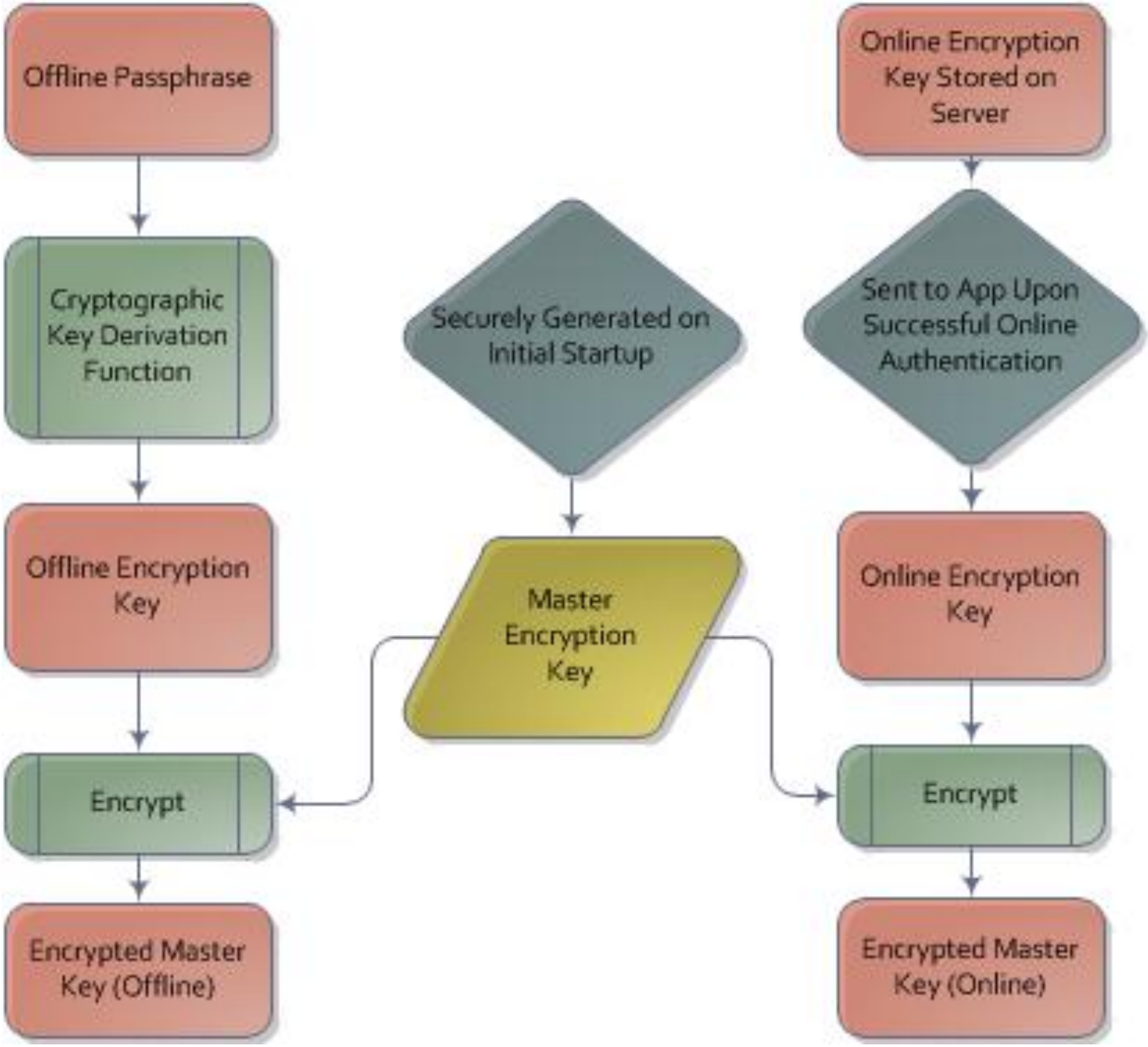
Authentication Designs

- Things get trickier due to online and offline access
- Online only apps could store keys server-side
  - Key returned only after successfully authenticating
  - Must handle server-side key storage.. this may be a pain
- What about offline access?
  - App might need access to data with even with no network access

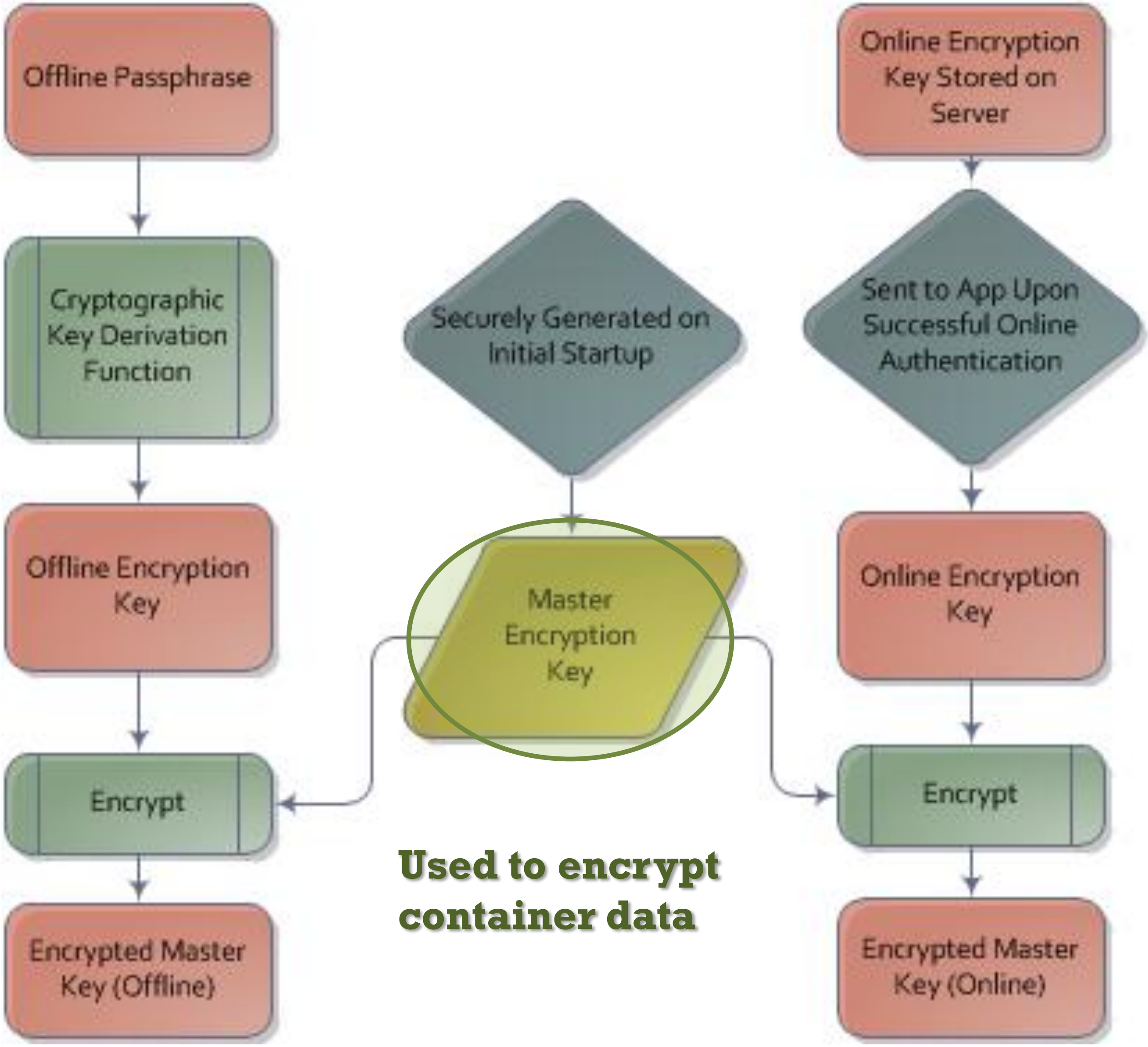
Authentication Design For Online and Offline Support


Offline Passphrase
Cryptographic Key Derivation Function
Offline Encryption Key
Encrypt
Encrypted Master Key (Offline)
Securely Generated on Initial Startup
Master Encryption Key
Online Encryption Key Stored on Server
Sent to App Upon Successful Online Authentication
Online Encryption Key
Encrypt
Encrypted Master Key (Online)

Authentication Design For Online and Offline Support

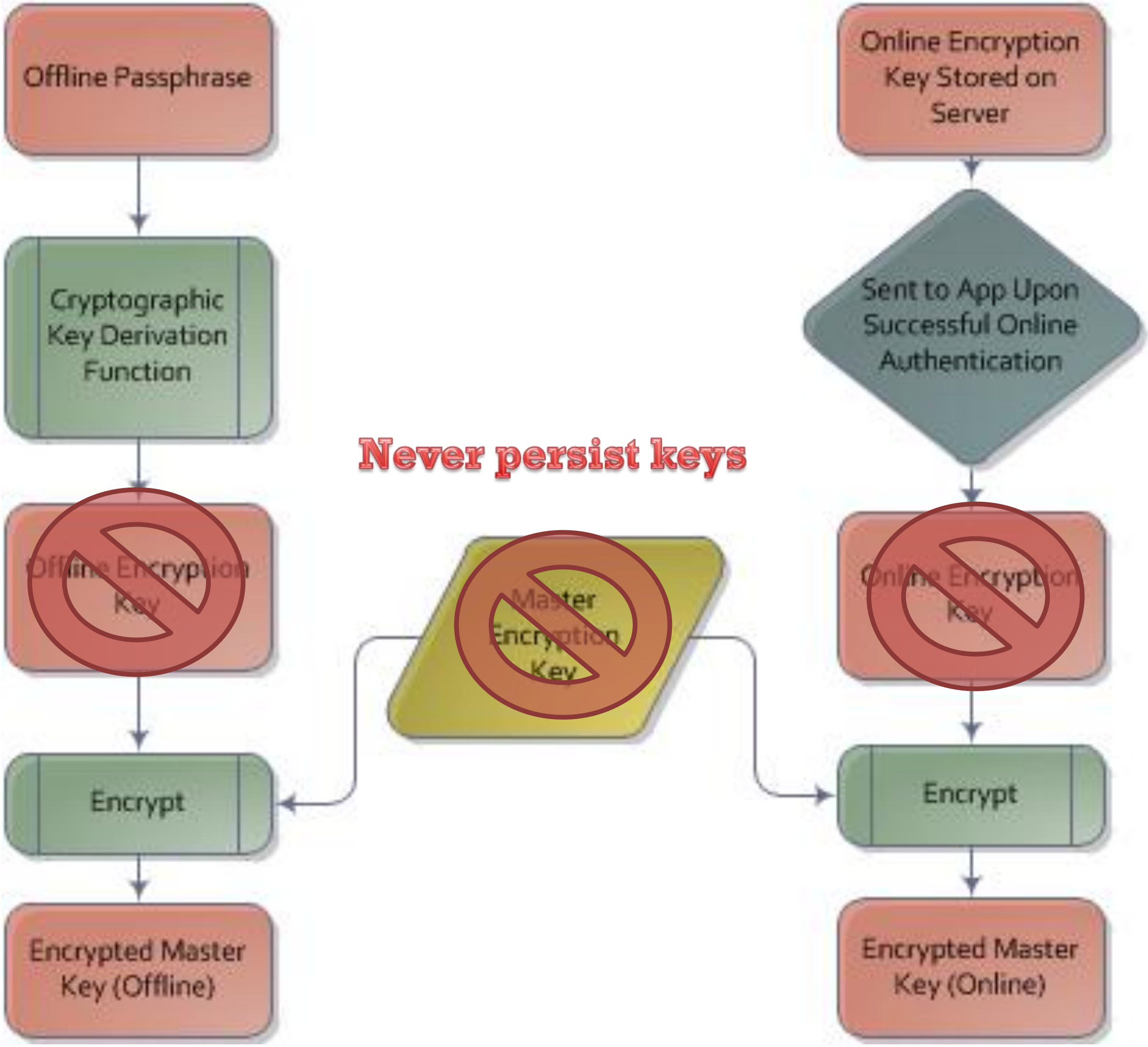

Offline Passphrase
Cryptographic Key Derivation Function
Offline Encryption Key
Encrypt
Encrypted Master Key (Offline)
Securely Generated on Initial Startup
Master Encryption Key
Online Encryption Key Stored on Server
Sent to App Upon Successful Online Authentication
Online Encryption Key
Encrypt
Encrypted Master Key (Online)

Authentication Design For Online and Offline Support

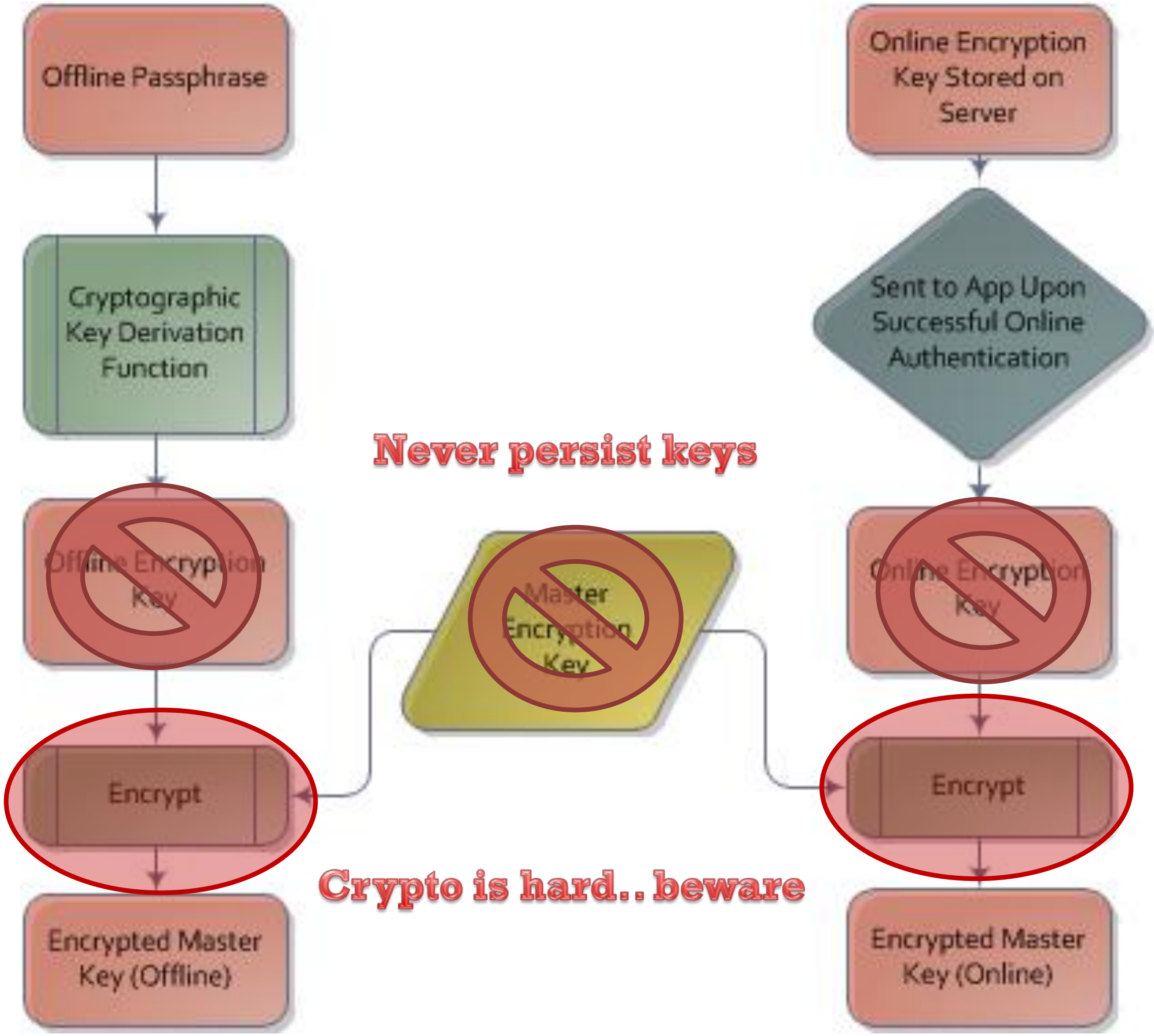

Offline Passphrase
Cryptographic Key Derivation Function
Offline Encryption Key
Encrypt
Encrypted Master Key (Offline)
Securely Generated on Initial Startup
Master Encryption Key
Online Encryption Key Stored on Server
Sent to App Upon Successful Online Authentication
Online Encryption Key
Encrypt
Encrypted Master Key (Online)

Used to encrypt container data

Authentication Design For Online and Offline Support


Offline Passphrase
Cryptographic Key Derivation Function
Offline Encryption Key
Encrypt
Encrypted Master Key (Offline)
Online Encryption Key Stored on Server
Sent to App Upon Successful Online Authentication
Online Encryption Key
Encrypt
Encrypted Master Key (Online)
Master Encryption Key
Never persist keys

Authentication Design For Online and Offline Support


Offline Passphrase
Cryptographic Key Derivation Function
Offline Encryption Key
Encrypt
Encrypted Master Key (Offline)
Master Encryption Key
Online Encryption Key Stored on Server
Sent to App Upon Successful Online Authentication
Online Encryption Key
Encrypt
Encrypted Master Key (Online)

Never persist keys

Crypto is hard.. beware

Offline Authentication

Common Issues

- Stored data not encrypted using passphrase derived key
  - Causes offline authentication to be susceptible to bypass
- Weak Key Derivation Function Used
  - PBKDF2 (minimum 4096 iterations) recommended
- Insufficient passphrase complexity enforcement

Weak Real World Example

Password & Data Vault

- Password vault authentication uses bcrypt
- Bcrypt is fairly resistant to offline brute force attacks

BUT…
HOW IS THE SYMMETRIC KEY GENERATED?

Weak Real World Examples

```
public Encryption(String paramString){
  byte[] arrayOfByte = new byte[16];
  arrayOfByte[0] = 0;
  [..snip..]
  arrayOfByte[15] = 0;
  [..snip..]

try {
    this.key = new SecretKeySpec(this.sha1HashBytes, "AES");
    this.ips = new IvParameterSpec(arrayOfByte);
[..snip..]
```

Single iteration SHA256 hash of passphrase

Decompiled Android Code From Application

Brute Force Time Comparisons

Time to brute force 1000 passwords

Algorithm	Time (s)
PBKDF2 (4096 iters)	317.647
SHA256	0.001
SHA256 + AES Decrypt	0.080

- Merkle-Damgard hash functions and AES are fast
- Susceptible to offline brute forcing

Outline

What are Security Containers?

How are Secure Containers Created?

Authentication Design Patterns and Data Encryption

Assessing the Strength of a Secure Container

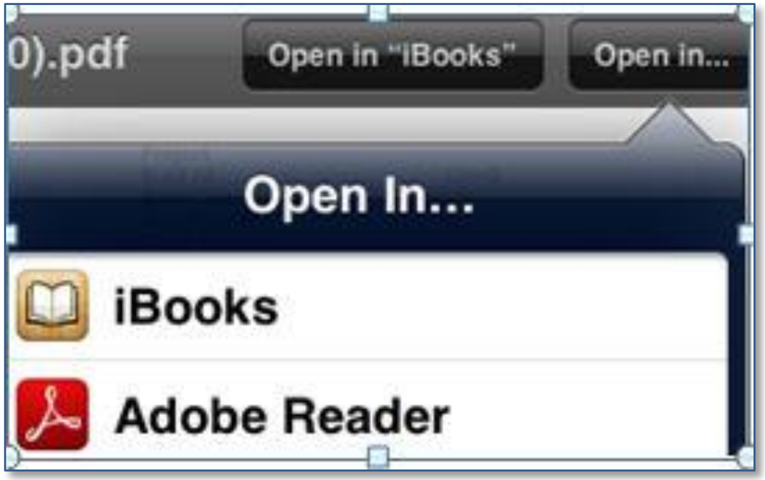
Limitations of Secure Containers

Completeness of Implementation

- The solution must not only consider the obvious APIs
- How about the subtle OS “features” that cache data?
  - NSHTTPCookieStorage on Persistent Cookies
  - NSURLRequest Caches
  - Document Interactions API
  - iOS Snapshots
  - Keyboard Caching
- How about Keychain Data?
- Are filenames also encrypted?

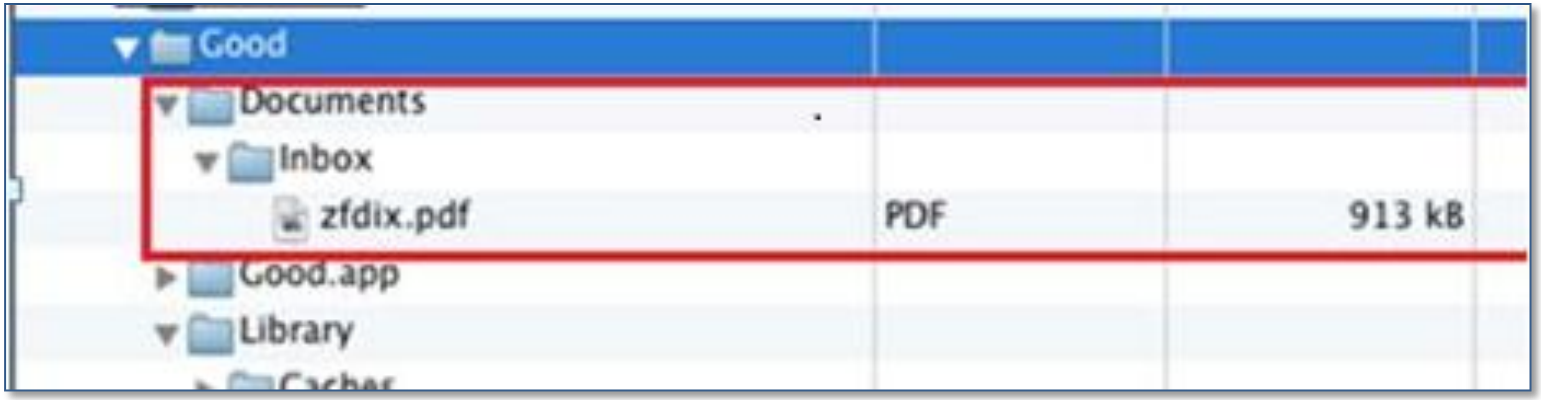
Good For Enterprise (GFE)

- iOS Document Interaction API used for handling documents
  - Save document into the GFE Container
  - Email document through corporate email within GFE
- Allows GFE to be used to open specific file types in iOS



Good For Enterprise (GFE)

- Open-in must bypass iOS Sandbox
- iOS System writes the file to GFE App Container
  - Documents/Inbox folder
- File is not protected by the GFE data encryption
- File persists unencrypted for an extended period



```
▼ Good
  ▼ Documents
    ▼ Inbox
        zfdix.pdf                PDF        913 kB
  ▶ Good.app
  ▼ Library
    ▶ Caches
```

Client's Custom Secure Container

- Missed wrapping calls to NSHTTPCookieStorage
- Apps store persistent cookies in plaintext
  - Library/Cookies/Cookies.binarycookies
- The cookie store also not removed on data wipes

Outline

- What are Security Containers?
- How are Secure Containers Created?
- Authentication Design Patterns and Data Encryption
- Assessing the Strength of a Secure Container
- Limitations of Secure Containers

Remote Wipes

- The traditional iOS MDM remote wipes cannot be used
- Remote wipes must be triggered at app level
- iOS limitations allow apps to only enforce wipe while active

Isn't iOS 7 Supposed To Have Some New Background Features?

Remote Wipes in iOS 7

- iOS 7 added Background Modes Capability
- Background fetch to poll for remote resets periodically
  - Requires wrapping to modify Info.plist file
  - Sounds promising, more research needed
- Remote Notifications
  - Not practical for application wrapping solutions
  - Requires:
    - APNS Service Setup
    - Application specific APNS certificates

Client-side Policy Enforcement

- All client-side restrictions can be bypassed
- Pick your poison
  - Intercept & modify policies as they are sent to the device
  - Modify policies while cached on the device
  - Runtime hooking of policy handling methods
- Performed checks server-side where possible
- Move critical pieces of code to low level code
  - Obj-C is easy to reverse.. make it harder for the bad guys

Jailbreak Detection

- Most of the exploits pointed out require a Jailbreak
- Preventing secure containers from running on Jailbroken device goes a long way
- If you can't bypass Xcon Cydia app, epic fail
- Detection should be low-level and difficult to reverse

http://appminder.nesolabs.de/

Jailbreak Detection Generator in ASM + AntiDebugging

Thanks For Coming!

- Come visit the GDS & Send Safely Booth!
- Check out the GDS Blog for updates on this topic
- Slides and Code will be posted to GDS Github page
  - https://github.com/GDSSecurity

Contact Info:
email: rgutierrez@gdssecurity.com
twitter: @rgutie01
github: https://github.com/rongutierrez