iOS applications auditing

AppSec Forum

Western Switzerland

Julien Bachmann / julien@scrt.ch

SCRT Information Security

- Motivations
- Quick review of the environment
- Common flaws
- Information gathering
- Network analysis
- Software reverse engineering

Preamble

- Security engineer @ SCRT
- Teacher @ HEIG-VD
  - Areas of interest focused on reverse engineering, software vulnerabilities, mobile devices security and OS internals
- Not an Apple fanboy
  - But like all the cool kids...
- Goals
  - This presentation aims at sharing experience and knowledge in iOS apps pentesting
- Contact
  - @milkmix_

motivations | why ?

- More and more applications
  - Most of Fortune-500 are deploying iPads
  - Growth in mobile banking
  - Mobile eShop
  - Internal applications
- Need for security
  - Access and storage of sensitive information
  - Online payments

environment | devices

- Latest devices
  - Apple A5 / A5X / A6 / A6X
  - Based on ARMv7 specifications
- Processor
  - RISC
  - Load-store architecture
  - Fixed length 32-bits instructions

environment | simulator

- Beware
  - Simulator != emulator
- More like a sandbox
  - Code compiled for Intel processors
  - 32-bits
  - ~/Library/Application Support/iPhone Simulator/<*v*>/Applications/<*id*>/

environment | applications

- Localisation
  - ~/Music/iTunes/iTunes Music/Mobile Applications/
  - /var/mobile/Applications/<*guid*>/<*appname*>.app/
- .ipa
  - Used to deploy applications
  - Zip file

environment | applications

- .plist
  - Used to store properties
  - XML files, sometimes in a binary format
  - Associates keys (CFString, CFNumber, …) with values
- plutil (1)
  - Convert binary plist file to its XML representation

flaws | communication snooping

- Secure by default
  - Well... at least if the developer is using URLs starting with HTTP**S**://
- Even if a fake certificate is presented !
  - The *DidFailWithError* method is called

```
NSURLRequest *theRequest =
    [NSURLRequest requestWithURL:[NSURL URLWithString:@"https://epreuves1.insomni.hack/getChallenge.php"]
                cachePolicy:NSURLRequestReloadIgnoringLocalCacheData
            timeoutInterval:10.0];
NSURLConnection *theConnection=[[NSURLConnection alloc] initWithRequest:theRequest delegate:self];
```

flaws | communication snooping

- Ok, but what about real life ?
  - A lot of development environments are using self-signed certificates
  - No built-in method to include certificates in the simulator
- Obviously, what did the developers ?
  - *Let's check what's on stackoverflow.com...*

flaws | communication snooping

- How to (potentialy) wreak havoc
  - Implement the two following delegates only to bypass certificates validation
  - *CanAuthenticateAgainstProtectionSpace*
  - *DidReceiveAuthenticationChallenge*

```
- (BOOL)connection:(NSURLConnection *)connection
canAuthenticateAgainstProtectionSpace:(NSURLProtectionSpace *)protectionSpace {

    return [protectionSpace.authenticationMethod isEqualToString:NSURLAuthenticationMethodServerTrust];
}

- (void)connection:(NSURLConnection *)connection
didReceiveAuthenticationChallenge:(NSURLAuthenticationChallenge *)challenge {

    [challenge.sender useCredential:[NSURLCredential credentialForTrust:challenge.protectionSpace.serverTrust]
         forAuthenticationChallenge:challenge];
}
```

flaws | data storage

- Most applications are working connected
  - Still some information locally stored
    - plist
    - SQLite3 databases
    - ...
  - Could include sensitive data
- Built-in protection
  - Data Protection API since iOS 4.0
    - http://code.google.com/p/iphone-dataprotection/
  - New attributes when working on files, Keychain entries or databases
  - Automatically used when calling NSURLCredentialStorage:setDefaultCredential but could not change protection type

flaws | data storage

Attribute	Definition
kSecAttrAccessibleWhenUnlocked	Only if unlocked
kSecAttrAccessibleAfterFirstUnlock	Unlocked at least once
kSecAttrAccessibleAlways	Do not use Data Protection API
kSecAttrAccessibleWhenUnlockedThisDeviceOnly	Only if unlocked, but do not store in backups
kSecAttrAccessibleAfterFirstUnlockThisDeviceOnly	Unlocked at least once, but do not store in backups
kSecAttrAccessibleAlwaysThisDeviceOnly	Do not store in backups

flaws | data storage

- Cryptographic primitives
  - Common Crypto Library
    - *CCCrypt()*
      - *kCCEncrypt*
      - *kCCDecrypt*

flaws | information disclosure

- The previous seems obvious, but...
  - Logs ?
  - Automagically created files ?

flaws | external interactions

- Files handling
  - *CFBundleDocumentTypes* in Info.plist
- IPC-like mechanism
  - URIs handlers
  - *CFBundleURLTypes* in Info.plist
  - Implementation of *handleOpenURL* or *OpenURL*

flaws | external interactions

- Memory management vulnerabilities
  - Objective-C classes are well protected
  - Still possible to introduce vulnerabilities if developing custom parsing functions for homegrown protocol
  - Beware to the old threats : format strings
  - Most likely result : app crash due to software exploitation protections
- HTML / Javascript injection
  - UIWebView controller used to render web pages
  - More related to server side vulnerabilities

flaws | server side

- Most of the time, included in the scope of the audit
  - Lot of applications are communicating with web-services
- Common flaws
  - No need to present the Top10

info gathering | apple's tools

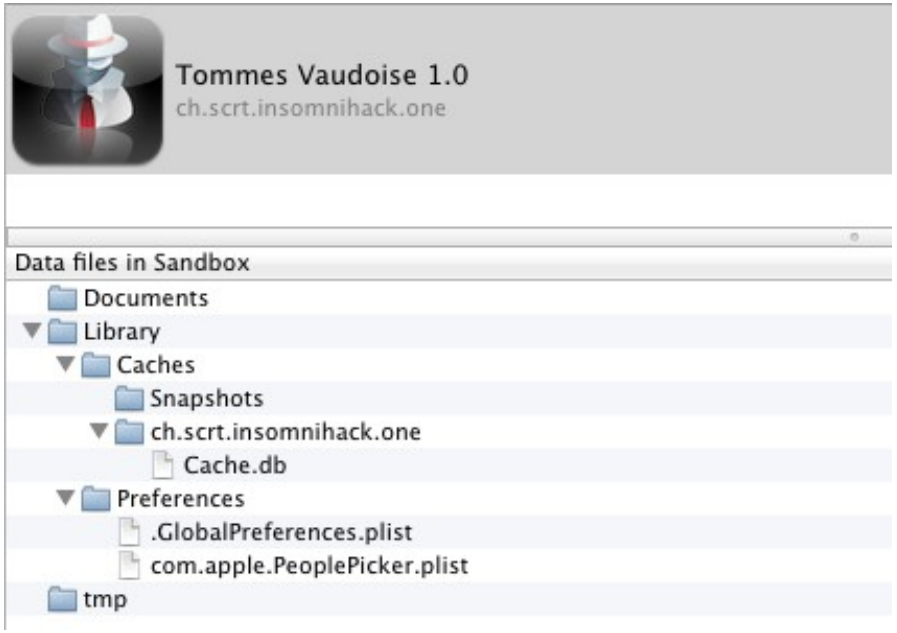
- First idea most people will have
  - *Let's jailbreak it !*
- There is another way
  - Stealthier to do a first recon
  - Still, jailbreaking the auditor's device is mandatory
  - Kudos to the jailbreakers teams for their work !

info gathering | apple's tools

› Activate developer mode

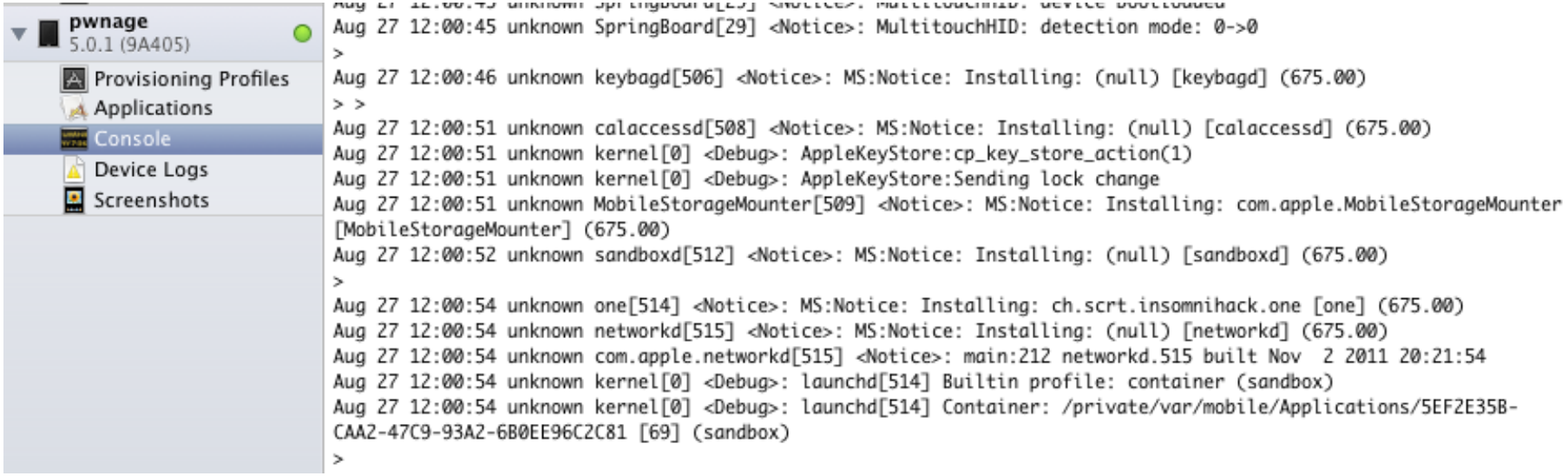
info gathering | apple's tools

- Access application's files
  - Only works for applications compiled in debug mode



info gathering | apple's tools

› Console / Application's logs



info gathering | getting access to the device

- Now you can do it
  - Enough documentation on *jailbreaking* online
- Personnal choice
  - Create a firmware with the smalest footprint as jailbreak detection mechanisms mostly check for Cydia presence
  - Use device that can be pwned using bootloader vulnerability in DFU mode
  - Use *tcprelay.py* relying on *usbmux* to ssh to the device through the usb cable

info gathering | keychain items

- Keychain Dumper
  - https://github.com/ptoomey3/Keychain-Dumper

Terminal — zsh — 80×40

```
pwnage:~ root# ./keychain_dumper
Generic Password
----------------
Service: com.apple.managedconfiguration
Account: Private
Entitlement Group: apple
Label: (null)
Generic Field: (null)
Keychain Data: (null)

Generic Password
----------------
Service: AirPort
Account: sholmes
Entitlement Group: apple
Label: (null)
Generic Field: (null)
Keychain Data: everyday

Generic Password
----------------
Service: push.apple.com
Account:
Entitlement Group: com.apple.apsd
Label: APSPublicTokens
Generic Field: (null)
Keychain Data: (null)

Generic Password
----------------
Service: AirPort
Account: plop
Entitlement Group: apple
Label: (null)
Generic Field: (null)
Keychain Data: a

No Internet Password Keychain items found.
pwnage:~ root# exit
logout
```

network analysis | communication snooping

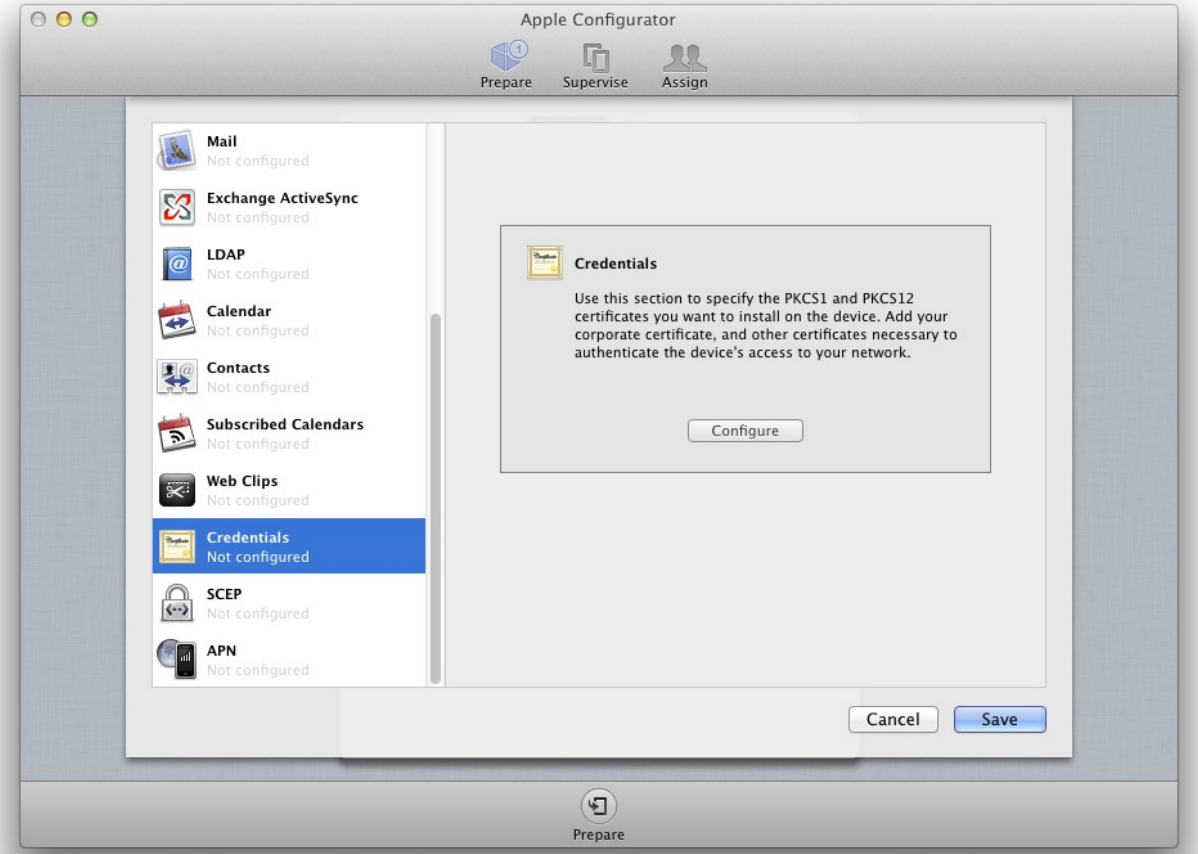
- Main idea
  - Use {burp ;zap ;...} to intercept the trafic and manipulate it
- Problem
  - *What about if the developers are using SSL and best-practices ?*

network analysis | communication snooping

- If you are doing an assignment mixing pentest and code review
  - Use the Simulator
- Certificates store
  - Based on a SQLite3 database
  - ~/Library/Application Support/iPhone Simulator/<*sdk*>/Library/Keychains/TrustStore.sqlite3
  - GDSSecurity released a script automating the insertion of x509 certificates in the database
    - https://github.com/GDSSecurity/Add-Trusted-Certificate-to-iOS-Simulator

network analysis | communication snooping

- Using a device
  - Generate CA and sign certificate
  - Upload the CA to the device using *Apple Configurator*



network analysis | communication snooping

- Won't go further on this subject
  - Joins *classic* web service pentesting
  - Except you are using a specific application and not a browser

reverse engineering | why ?

- Pentesting is not code review
  - If you want to understand an application behavior you have to reverse it
- Static
  - Hexdump
  - otool
  - IDA Pro
  - Hopper
- Dynamic
  - GDB

reverse engineering | ida pro

```
ADD         R1, PC ; selRef_class
MOVT.W      R0, #0x9F
MOV         R4, R2
ADD         R0, PC ; classRef_NSKeyedArchiver
LDR         R1, [R1] ; "class"
LDR         R0, [R0] ; _OBJC_CLASS_$_NSKeyedArchiver
BLX.W       _objc_msgSend
MOV         R2, R0
MOV         R0, (selRef_isKindOfClass_ - 0x25AA70) ; selRef_isKindOfClass_
ADD         R0, PC ; selRef_isKindOfClass_
LDR         R1, [R0] ; "isKindOfClass:"
MOV         R0, R4
BLX.W       _objc_msgSend
TST.W       R0, #0xFF
BEQ         loc_25AB1E
```

```
MOV         R0, (selRef_projectName - 0x25AA88) ; selRef_projectName
ADD         R0, PC ; selRef_projectName
LDR         R1, [R0] ; "projectName"
MOV         R0, R5
BLX.W       _objc_msgSend
MOV         R2, R0
MOV         R0, (selRef_encodeObject_forKey_ - 0x25AAA0) ; selRef_encodeObject_forKey_
MOVW        R8, #0x9DCE
ADD         R0, PC ; selRef_encodeObject_forKey_
MOVT.W      R8, #0xA3
ADD         R8, PC ; __MergedGlobals_33
LDR         R6, [R0] ; "encodeObject:forKey:"
MOV         R0, R4
LDR.W       R3, [R8]
MOV         R1, R6
BLX.W       _objc_msgSend
MOV         R0, (selRef_serverName - 0x25AABE) ; selRef_serverName
ADD         R0, PC ; selRef_serverName
LDR         R1, [R0] ; "serverName"
MOV         R0, R5
BLX.W       _objc_msgSend
LDR.W       R3, [R8,#(dword_C94878 - 0xC94874)]
MOV         R1, R6
MOV         R2, R0
MOV         R0, R4
BLX.W       _objc_msgSend
MOV         R0, (selRef_serverPort - 0x25AADE) ; selRef_serverPort
ADD         R0, PC ; selRef_serverPort
LDR         R1, [R0] ; "serverPort"
MOV         R0, R5
BLX.W       _objc_msgSend
MOV         R2, R0
MOV         R0, (selRef_encodeInt32_forKey_ - 0x25AAF6) ; selRef_encodeInt32_forKey_
LDR.W       R3, [R8,#(dword_C9487C - 0xC94874)]
ADD         R0, PC ; selRef_encodeInt32_forKey_
LDR         R1, [R0] ; "encodeInt32:forKey:"
```

```
loc_25AB1E
MOVW        R0, #0x149C
MOV         R2, R6
MOVT.W      R0, #0x9F
ADD         R0, PC ; selRef_doesNotRecognizeSelector_
LDR         R1, [R0] ; "doesNotRecognizeSelector:"
MOV         R0, R5
BLX.W       _objc_msgSend
```

reverse engineering | hopper

one-app.hop

Read Executable | Back | Follow | D Mark As Data | A Mark As ASCII | C Mark As Code | P Mark As Procedure | N Symbol Names | S Segments | X Cross References | Show CFG | Pseudo Code | G GDB

Label Name

EntryPoint
_main_25a8
-[InsomniOneAppDelegate applicati...
-[InsomniOneAppDelegate applicati...
-[InsomniOneAppDelegate applicati...
-[InsomniOneAppDelegate applicati...
-[InsomniOneAppDelegate applicati...
-[InsomniOneAppDelegate applicati...
-[InsomniOneAppDelegate window]_...
-[InsomniOneAppDelegate setWindo...
-[InsomniOneAppDelegate viewCont...
-[InsomniOneAppDelegate setViewC...
-[InsomniOneViewController didRec...
-[InsomniOneViewController connec...
-[InsomniOneViewController connec...
-[InsomniOneViewController connec...
-[InsomniOneViewController mysetL...
-[InsomniOneViewController viewDi...
-[InsomniOneViewController viewDi...
-[InsomniOneViewController viewWil...
-[InsomniOneViewController viewDi...
-[InsomniOneViewController viewWil...
-[InsomniOneViewController viewDi...
-[InsomniOneViewController should...
-[InsomniOneViewController receive...
-[InsomniOneViewController setRec...
-[Tomme connection:didReceiveRes...
-[Tomme connection:didReceiveDat...
-[Tomme connectionDidFinishLoadi...
-[Tomme initialize]_2d74
-[Tomme one:]_2d80
-[Tomme two:]_2ef8

```
                                    ; Range 0x257c - 0x4438 (7868 bytes)
                                    ; File offset 5500 (7868 bytes)
                                    ; Flags : 0x0880000400
                                    ;
                                EntryPoint:
0000257c 00009DE5                       ldr     r0, [sp]
00002580 04108DE2                       add     r1, sp, #0x4
00002584 014080E2                       add     r4, r0, #0x1                    ; XREF=
00002588 042181E0                       add     r2, r1, r4 lsl #2
0000258c 07D0CDE3                       bic     sp, sp, #0x7
00002590 0230A0E1                       mov     r3, r2
00002594 044093E4                       ldr     r4, [r3], #0x4                  ; XREF=
00002598 000054E3                       cmp     r4, #0x0
0000259c FCFFFF1A                       bne     0x2594
000025a0 000000FA                       blx     _main_25a8
000025a4 8D0A00EA                       b       imp___symbolstub1__exit

================ B E G I N   O F   P R O C E D U R E ================

                                    ; Basic Block Input Regs: r0 r1 r7 sp lr pc -  Killed Regs
                                _main_25a8:
000025a8 80B5                           push    {r7, lr}                        ; XREF=
000025aa 6F46                           mov     r7, sp
000025ac 8AB0                           sub     sp, #0x28
000025ae 0122                           movs    r2, #0x1
000025b0 C0F20002                       movt    r2, #0x0
000025b4 0023                           movs    r3, #0x0
000025b6 C0F20003                       movt    r3, #0x0
000025ba 0993                           str     r3, [sp, #0x28 + var_36]
000025bc 0890                           str     r0, [sp, #0x28 + var_32]
000025be 0791                           str     r1, [sp, #0x28 + var_28]
000025c0 0592                           str     r2, [sp, #0x28 + var_20]
000025c2 0493                           str     r3, [sp, #0x28 + var_16]
000025c4 02F010ED                       blx     imp___symbolstub1__objc_autoreleasePoolPush
000025c8 0899                           ldr     r1, [sp, #0x28 + var_32]
000025ca 079A                           ldr     r2, [sp, #0x28 + var_28]
000025cc 114B                           ldr     r3, = 0x3176
000025ce 7B44                           add     r3, pc                          ; 0x574
000025d0 1B68                           ldr     r3, [r3]                        ; @0x57
000025d2 DFF83C90                       ldr.w   r9, = 0x30d6
000025d6 F944                           add     r9, pc                          ; 0x56b
000025d8 D9F80090                       ldr.w   r9, [r9]                        ; @sele
```

Analysis segment __nl_symbol_ptr
Analysis segment __objc_classlist
Analysis segment __objc_catlist
Analysis segment __objc_protolist
Analysis segment __objc_imageinfo
Analysis segment __objc_const
Analysis segment __objc_selrefs
Analysis segment __objc_classrefs
Analysis segment __objc_superrefs
Analysis segment __objc_data
Analysis segment __cfstring
Analysis segment __objc_ivar

Address 0x257c, Segment __text, EntryPoint + 0, file offset 0x157c

SCRT Information Security

APPLICATION SECURITY FORUM WESTERN SWITZERLAND

reverse engineering | hopper

Pseudo Code

```
function -[Tomme one:]_2d80 {
    *(r13 - 0x38 + 0x34) = r0;
    *(r13 + 0x30) = r1;
    *(r13 + 0x2c) = r2;
    *(r13 + 0x18) = 0x0;
    r0 = imp___symbolstub1__NSStringFromSelector();
    *(r13 + 0x28) = r0;
    *(r13 + 0x14) = *(r13 + 0x28);
    r0 = imp___symbolstub1__objc_msgSend();
    r0 = imp___symbolstub1__objc_msgSend();
    *(r13 + 0x28) = r0;
    r0 = imp___symbolstub1__objc_msgSend();
    *(r13 + 0x34).challenge = r0;
    *(r13 + 0x34).caller = *(r13 + 0x2c);
    r0 = imp___symbolstub1__objc_msgSend();
    *(r13 + 0x24) = r0;
    *(r13 + 0x10) = *bind__OBJC_CLASS_$_NSURLRequest;
    r0 = imp___symbolstub1__objc_msgSend();
    *(r13 + 0x4) = 0x40240000;
    *r9 = 0x0;
    *(r13 + 0xc) = r0;
    r0 = imp___symbolstub1__objc_msgSend();
    *(r13 + 0x20) = r0;
    imp___symbolstub1__objc_msgSend();
    *(r13 + 0x8) = *(r13 + 0x34);
    r0 = imp___symbolstub1__objc_msgSend();
    *(r13 + 0x1c) = r0;
    if (*(r13 + 0x1c) != *(r13 + 0x18)) {
            imp___symbolstub1__objc_msgSend();
            r0 = imp___symbolstub1__objc_msgSend();
            **0x505c = r0;
    }
    return r0;
}
```

SCRT Information Security

APPLICATION SECURITY FORUM WESTERN SWITZERLAND

reverse engineering | need to know

- Architecture
  - File format for Objective-C executables
  - ARM basics
- Language
  - Objective-C basics
  - ARM assembly basics
- AppStore
  - How to decrypt AppStore binaries

reverse engineering | appstore

- Applications from the AppStore are encrypted
  - DRM
  - *Fair Play* like
- Do it manually
  - GDB, set, go !
- Automatic
  - Crackulous (won't work on executables compiled with PIE)
  - Clutch

reverse engineering | obj-c to arm

- Objective-C
  - [ObjectPointer Method:42]
- C++ equivalent
  - ObjectPointer->Method(42)
- Pseudo C generated by the compiler
  - objc_msgSend(ObjectPointer, "Method", 42)
- ARM assembly
  -

```
LDR     R1, =off_1337   ; offset to "Method"
MOV     R2, 0x2a
MOV     R0, R4          ; R4 contains Object @
BL      _objc_msgSend
```

reverse engineering | obj-c to arm

- Reflective language
  - Access to own definition

```
- (void) one:(InsomniOneViewController*)v {
    NSString*   tmp = NSStringFromSelector(_cmd);

    tmp = [tmp substringToIndex:[tmp length] - 1];
```

  - Call methods from names

```
s = NSSelectorFromString(dataString);
t = [Tomme new];
[t performSelector:s withObject:self];
```

reverse engineering | where to begin ?

- Main class
  - Derived from *UIApplicationDelegate*
  - Implements *applicationDidFinishLaunching* or *applicationDidFinishLaunchingWithOptions*
- Views
  - Derived from *UI*ViewController*
  - Implement *viewDidLoad*

reverse engineering | extracting class info

› class-dump

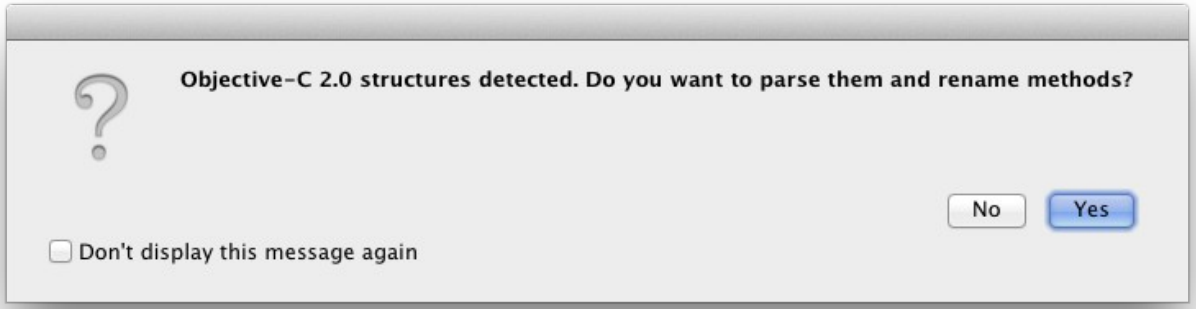
```
@interface OneAppDelegate : NSObject <UIApplicationDelegate>
{
    UIWindow *window;
    mainController *controller;
    db *db1;
}

- (BOOL)application:(id)arg1 didFinishLaunchingWithOptions:(id)arg2;
- (void)applicationDidEnterBackground:(id)arg1;
- (void)applicationWillResignActive:(id)arg1;
- (void)applicationWillTerminate:(id)arg1;
- (void)dealloc;

@end
```

reverse engineering | static analysis

- Goals
  - Understand the application's algorithms
- Tools
  - IDA Pro
  - Hopper
  - *fixobjc.idc* to resolve XREFs and parse Obj-C structures
  - Built-in functionality since version 6.2



reverse engineering | dynamic analysis

- Goals
  - Understand the application's algorithms
  - Allows to tamper data
- *But data tampering is not done with Burp ?*
  - What happens when the protocol is encrypted ?
  - Need to find the function encrypting the data
  - Set breakpoint
  - Modify the data in-memory

reverse engineering | dynamic analysis

- GDB
  - Provided by Apple as part of iOS SDK
  - Standalone version or *gdbserver* with *gdb* version for ARM targets
  - Advantage of *gdbserver* is ability to launch GUI applications
  - Highly recommend *gdbinit** by @osxreverser
- Entitlement
  - Binary will not run out-of-the-box on iDevices
  - Need to add entitlements after extracting ARMv7 binary
  - *ldid* to the rescue

* http://reverse.put.as/gdbinit/

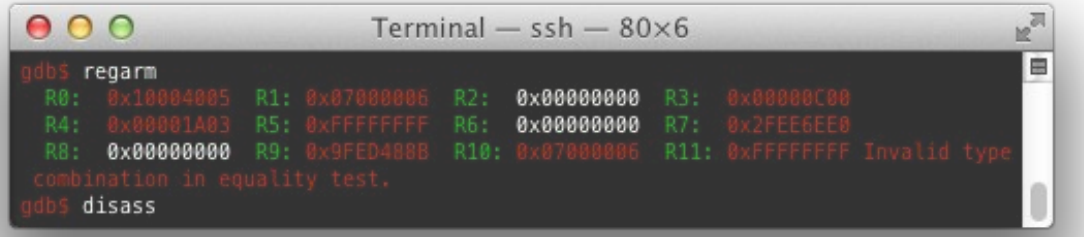
reverse engineering | dynamic analysis

Terminal — zsh — 80×18

```
.-(~/local/iOS)-----------------------------------------(milkmix@yamabushi)-
`--> cat ent-gdb.xml                                                      [0]
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/
PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
        <key>com.apple.springboard.debugapplications</key>
        <true/>
        <key>get-task-allow</key>
        <true/>
        <key>task_for_pid-allow</key>
        <true/>
        <key>run-unsigned-code</key>
        <true/>
</dict>
</plist>
.-(~/local/iOS)-----------------------------------------(milkmix@yamabushi)-
`--> ./ldid -S ent-gdb.xml gdb-arm                                        [0]
```

reverse engineering | dynamic analysis

- Startup
  - *# ~/debugserver-armv7 -x spring <app>*
  - *gdb$ set shlib-path-substitutions /*
    /Applications/Xcode.app/Contents/Developer/Platforms/iPhoneOS.platf
    orm/Developer/SDKs/iPhoneOS5.1.sdk/
  - *gdb$ target remote-macosx localhost:1999*
  - *gdb$ source ~/gdbinit8*
  - *gdb$ b [InsomniOneViewController viewDidLoad]*
  - *gdb$ c*
  - *gdb$ regarm*



reverse engineering | dynamic analysis

› Warning

Terminal — gdb-arm-apple-da — 140×23

```
gdb$ x/10hi $pc
0xa09e0 <-[InsomniOneViewController viewDidLoad]>:      push    {r7, lr}
0xa09e2 <-[InsomniOneViewController viewDidLoad]+2>:    mov     r7, sp
0xa09e4 <-[InsomniOneViewController viewDidLoad]+4>:    sub     sp, #48
0xa09e6 <-[InsomniOneViewController viewDidLoad]+6>:    movs    r2, #0
0xa09e8 <-[InsomniOneViewController viewDidLoad]+8>:    movt    r2, #0  ; 0x0
0xa09ec <-[InsomniOneViewController viewDidLoad]+12>:   str     r0, [sp, #44]
0xa09ee <-[InsomniOneViewController viewDidLoad]+14>:   str     r1, [sp, #40]
0xa09f0 <-[InsomniOneViewController viewDidLoad]+16>:   ldr     r0, [sp, #44]
0xa09f2 <-[InsomniOneViewController viewDidLoad]+18>:   str     r0, [sp, #32]
0xa09f4 <-[InsomniOneViewController viewDidLoad]+20>:   ldr     r0, [pc, #264]  (0xa0b00 <-[InsomniOneViewController viewDidLoad]+288>)
gdb$ x/10wi $pc
0xa09e0 <-[InsomniOneViewController viewDidLoad]>:      strbtmi r11, [pc], -r0, lsl #11
0xa09e4 <-[InsomniOneViewController viewDidLoad]+4>:    andcs   r11, r0, #140   ; 0x8c
0xa09e8 <-[InsomniOneViewController viewDidLoad]+8>:    andeq   pc, r0, #12     ; 0xc
0xa09ec <-[InsomniOneViewController viewDidLoad]+12>:   tstls   r10, r11
0xa09f0 <-[InsomniOneViewController viewDidLoad]+16>:   andls   r9, r8, r11, lsl #16
0xa09f4 <-[InsomniOneViewController viewDidLoad]+20>:   ldrbtmi r4, [r8], #-2114
0xa09f8 <-[InsomniOneViewController viewDidLoad]+24>:   andls   r6, r9, r0, lsl #16
0xa09fc <-[InsomniOneViewController viewDidLoad]+28>:   ldrbtmi r4, [r8], #-2111
0xa0a00 <-[InsomniOneViewController viewDidLoad]+32>:   stmdage r8, {r0, r11, sp, lr}
0xa0a04 <-[InsomniOneViewController viewDidLoad]+36>:   undefined instruction 0xf0029205
gdb$
```

reverse engineering | dynamic analysis

- Inspect / modify memory
  - *gdb$ po $r2*
  - *gdb$ set {int}0xcafebabe = 1337*
- For large amount of data
  - $ cat data.mod | hexdump -ve '1/4 "set {unsigned char *}(<*addr*> + %#2_ax) = %#02x\n"' > data.gdb

reverse engineering | instrumentation

- Automating an attack
  - Suppose you found something (SQL injection, …)
  - Possible to call methods using *gdb*'s *call* directive
  - Too slow to modify data on the fly by hands
- Solution
  - Use code injection to modify the behaviour of the application
  - Modify data automagically

reverse engineering | instrumentation

- This is where you start loving Objective-C
  - Hooking is a bundled feature
  - It's called 'swizzling'
- Principle
  - Use the functions provided by Apple, like
    - *class_replaceMethod*
    - *method_exchangeimplementations*

reverse engineering | instrumentation

intercept.m

```
#include <Foundation/Foundation.h>
#include <objc/objc.h>
#include <objc/runtime.h>

static IMP  _sevenIMP = Nil;

void    myseven(id self, SEL sel)
{
    NSLog(@"seven was called");
    (_sevenIMP)(self, sel);
}

static void __attribute__((constructor)) initialize(void)
{
    NSLog(@"hijacking bundle loaded");

    _sevenIMP = class_replaceMethod(objc_getClass("Tomme"), sel_registerName("seven:"), (IMP)myseven, "@:");
    if (!_sevenIMP)
    {
        NSLog(@"failed to set hook using swizzling");
    }
    else
    {
        NSLog(@"hook in place, waiting for the sheep...");
    }
}
```

Line 26, Column 2 Tab Size: 4 Objective-C

reverse engineering | instrumentation

Terminal — less — 80×10

```
$ /Applications/Xcode.app/Contents/Developer/Platforms/iPhoneOS.platform/Develop
er/usr/bin/arm-apple-darwin10-llvm-gcc-4.2 -c -o intercept.o intercept.m -isysro
ot /Applications/Xcode.app/Contents/Developer/Platforms/iPhoneOS.platform/Develo
per/SDKs/iPhoneOS5.1.sdk/ -fPIC

$ /Applications/Xcode.app/Contents/Developer/Platforms/iPhoneOS.platform/Develop
er/usr/bin/ld -dylib -lsystem -lobjc -framework Foundation -o intercept.dylib in
tercept.o -syslibroot /Applications/Xcode.app/Contents/Developer/Platforms/iPhon
eOS.platform/Developer/SDKs/iPhoneOS5.1.sdk/ -ios_version_min 5.0
```

reverse engineering | instrumentation

- Injecting into process
  - *DYLD_PRELOAD* for stand-alone launch
  - *DYLD_INSERT_LIBRARIES* and SpringBoard.plist modification to inject in all graphical applications

Questions ?

SCRT Information Security

APPLICATION SECURITY FORUM WESTERN SWITZERLAND