BISHOP FOX®

# Mobile Application Security Testing

## ASSESSMENT & CODE REVIEW


Sept. 31st 2014

# Presenters

ITAC 2014

## Bishop Fox

- Francis Brown
  *Partner*

- Joe DeMesy
  *Security Associate*

# Introductions

FRANCIS BROWN

- Hi, I'm Fran
- Partner at Bishop Fox
- You may remember me from such hacks as:
  - RFID Thief
  - Diggity Search Tool Suite
  - Sharepoint Hacking

# Introductions

JOE DEMESY

- Hi, I'm Joe
- Associate at Bishop Fox
- I like computers
  - That run a POSIX OS*
  - Phones are cool too
- Open source projects:
  - Root the Box
  - iSpy

# Agenda

COVERED TODAY

**Breaking iOS Apps –**

- Static analysis
- Dynamic analysis
- The future of iOS assessments
- Protections & counter-measures

**Breaking Android Apps –**

- Static Analysis
- Dynamic Analysis
- Protections & counter-measures

# App Security Requirements

OUR TARGETS

## Scenarios

- Online Finance
- Point of Sale
- Streaming Media
- Mobile Device Management (MDM)
- Games (cheating, etc.)

# THE GOLDEN RULE

APPLICATION SECURITY

# Users are Evil

EVERY LAST ONE OF ‘EM

- They have complete control
- Do not trust them
- Design applications and APIs accordingly

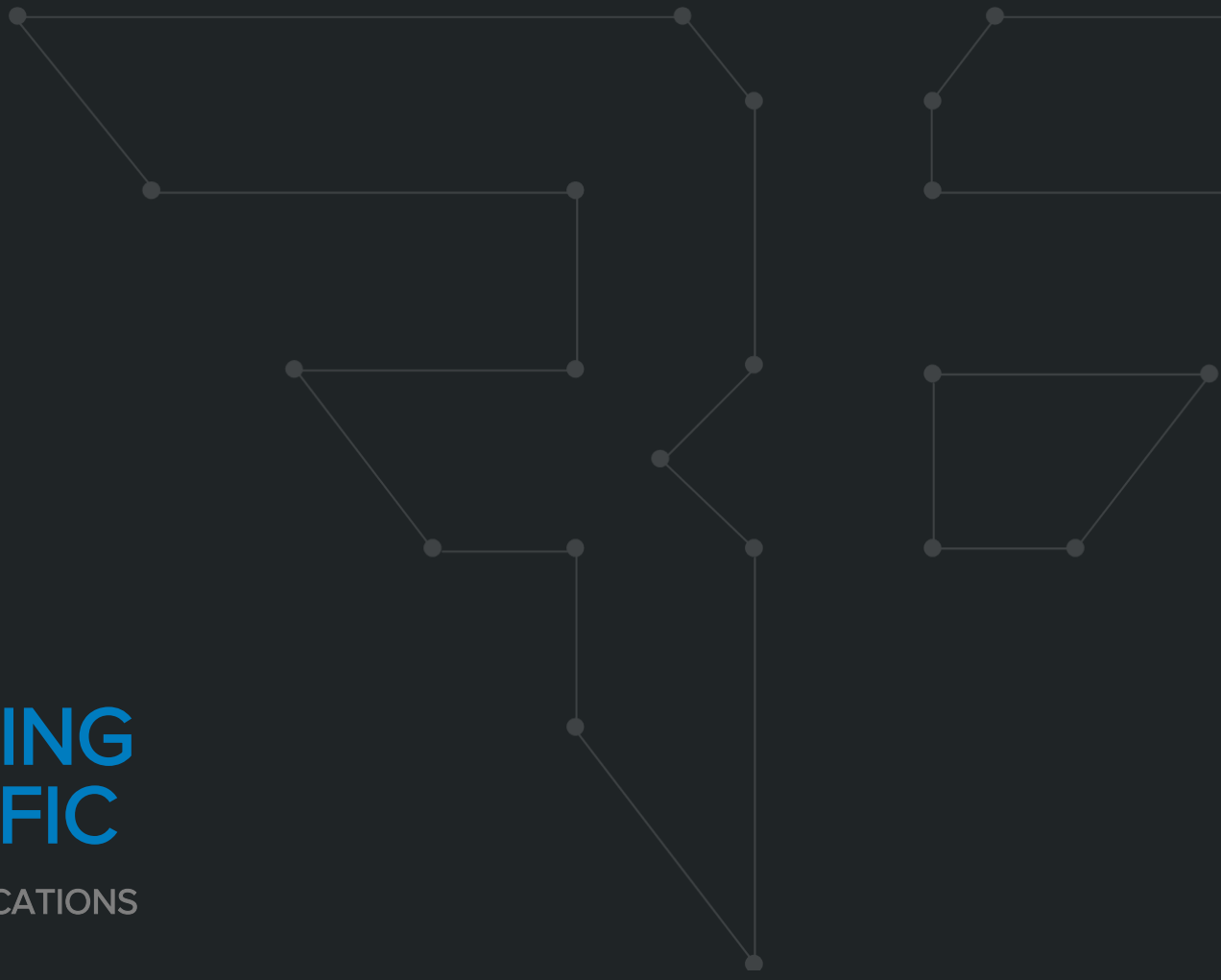
# IOS DYNAMIC ANALYSIS

## BREAKING IOS APPLICATIONS

# iOS Prerequisites

WHAT YOU NEED TO START

- Mac & Xcode
- HTTP Proxy
  - Burp Suite Pro ($300)
  - MitM Proxy ($0)
- ARM Disassembler *(optional)*
  - Hopper ($90)
  - IDA Pro ($600+)
- Jailbroken iOS Device
  - SSH access



# INTERCEPTING HTTP TRAFFIC

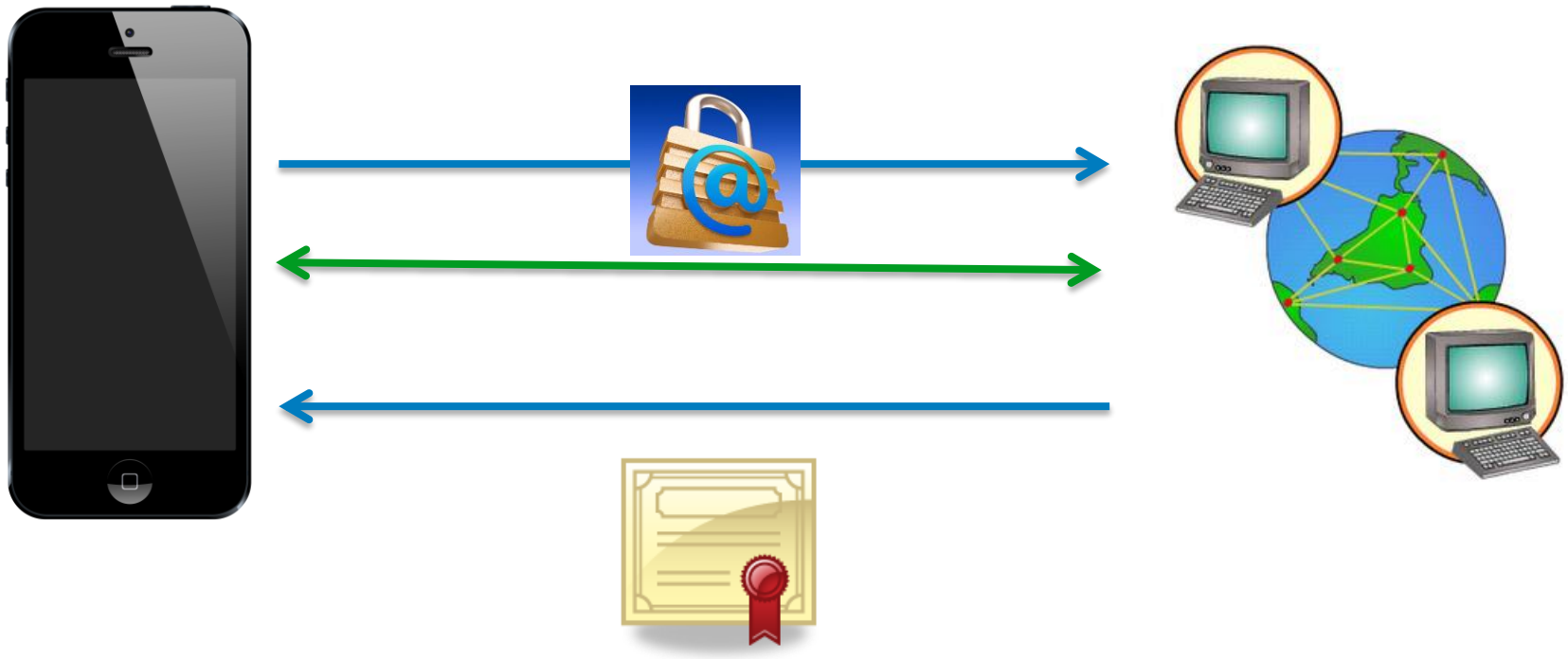
BREAKING MOBILE APPLICATIONS

# HTTP Proxy Setup

PROXY SETTINGS

# HTTP Proxy Setup

## INTERCEPTING HTTPS TRAFFIC

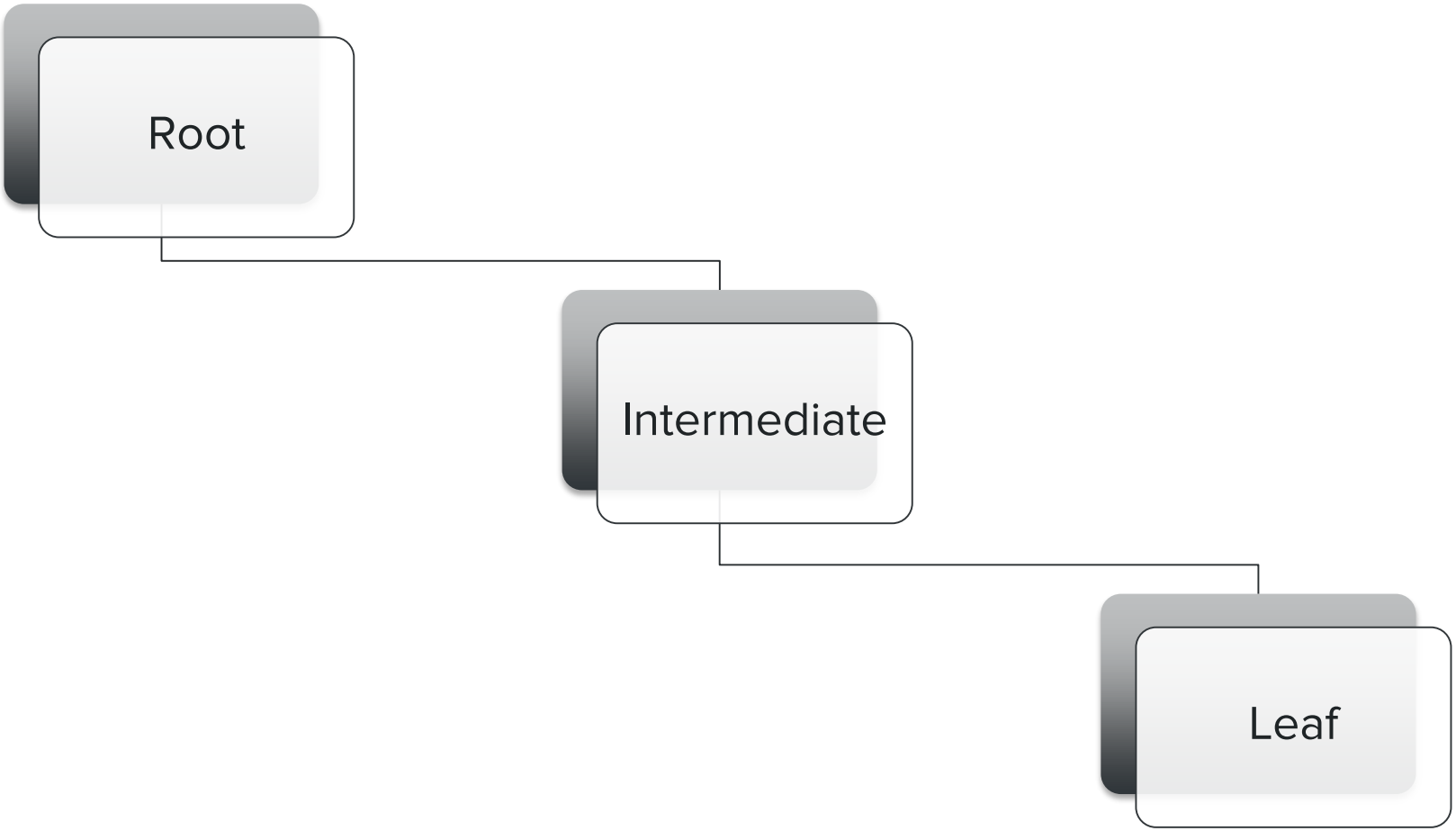


# HTTP Proxy Setup

## INTERCEPTING HTTPS TRAFFIC

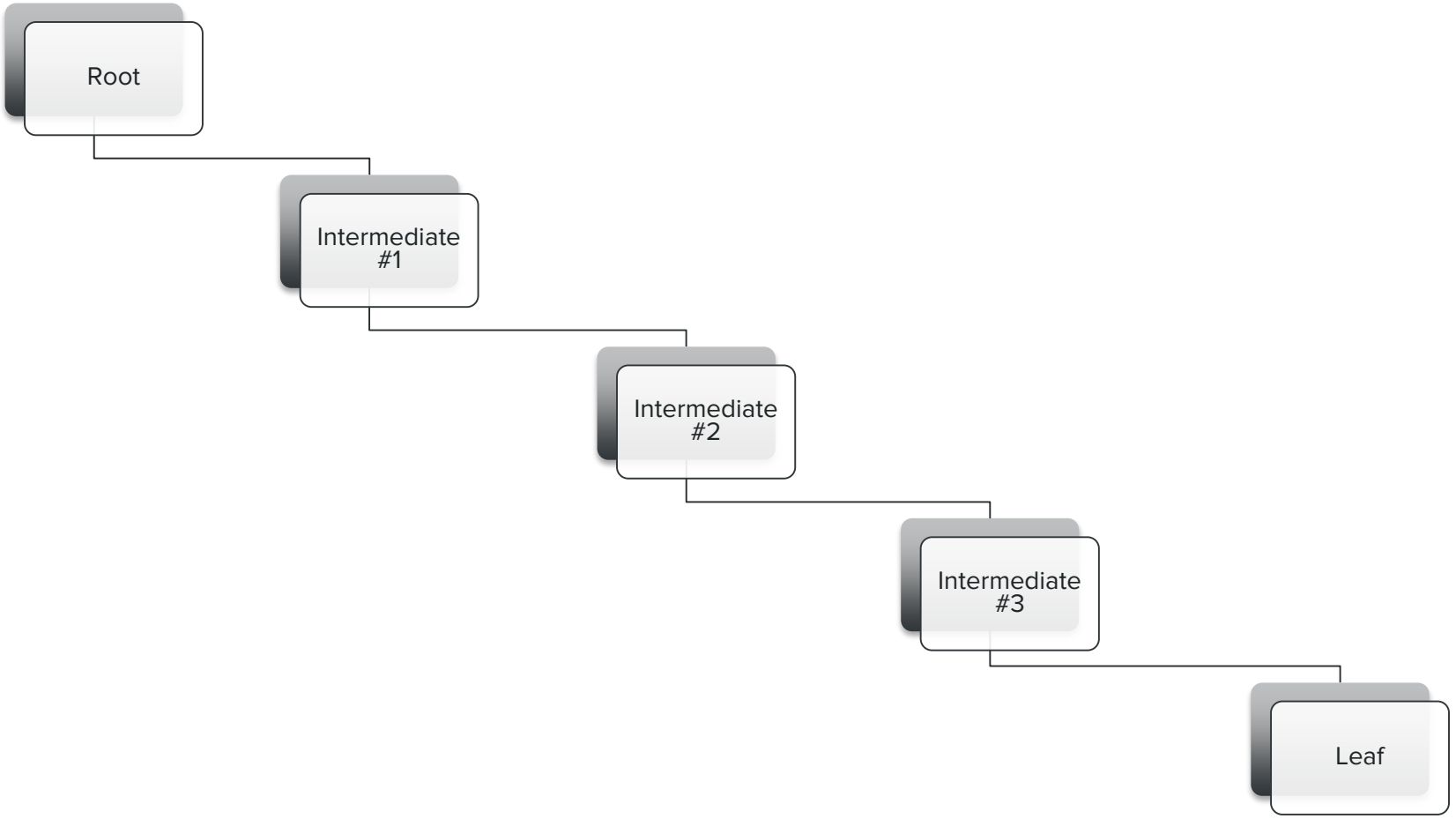
# The SSL Certificate Chain

CERTIFICATE VALIDATION



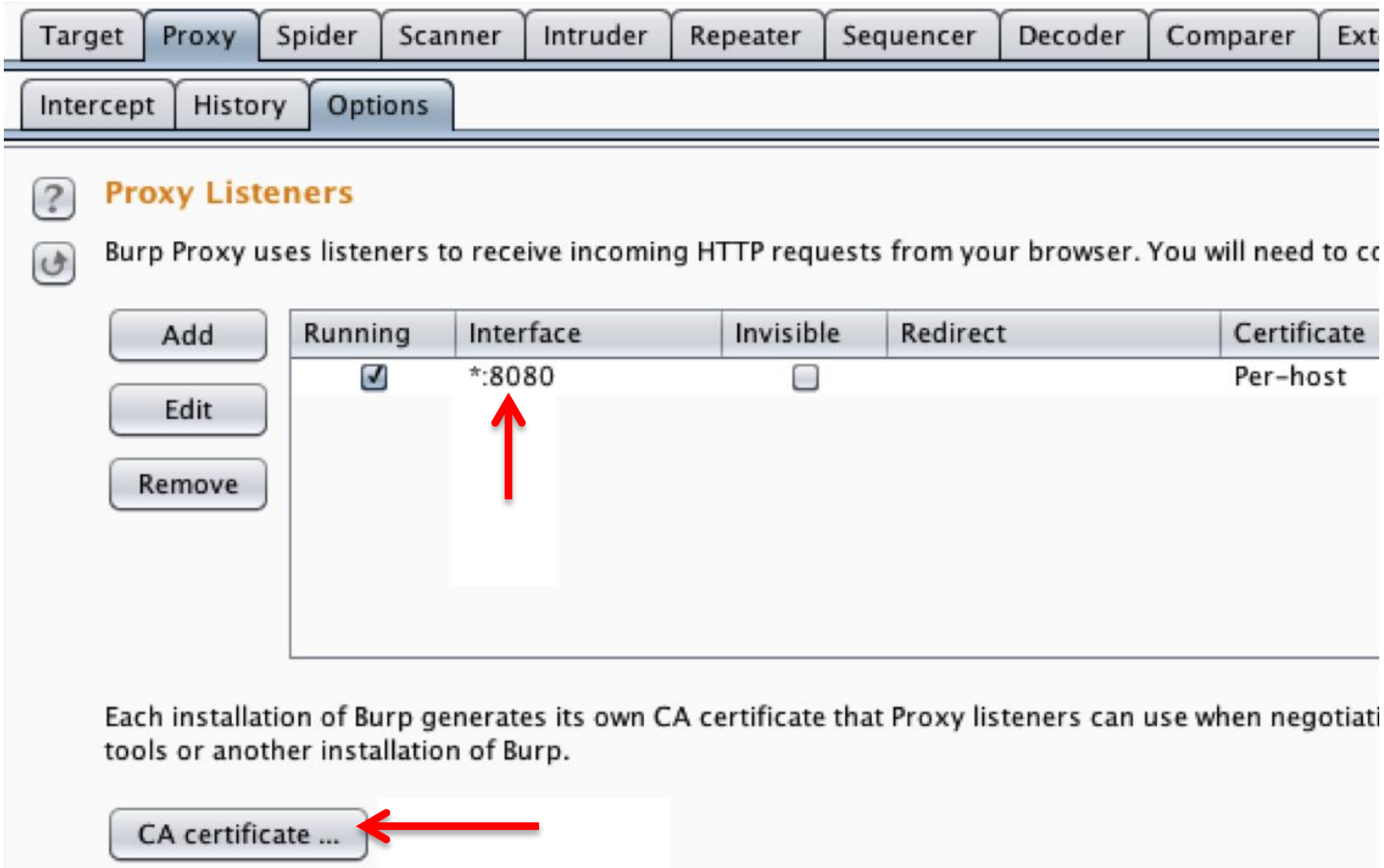
# SSL Certificate Chain

## CERTIFICATE VALIDATION



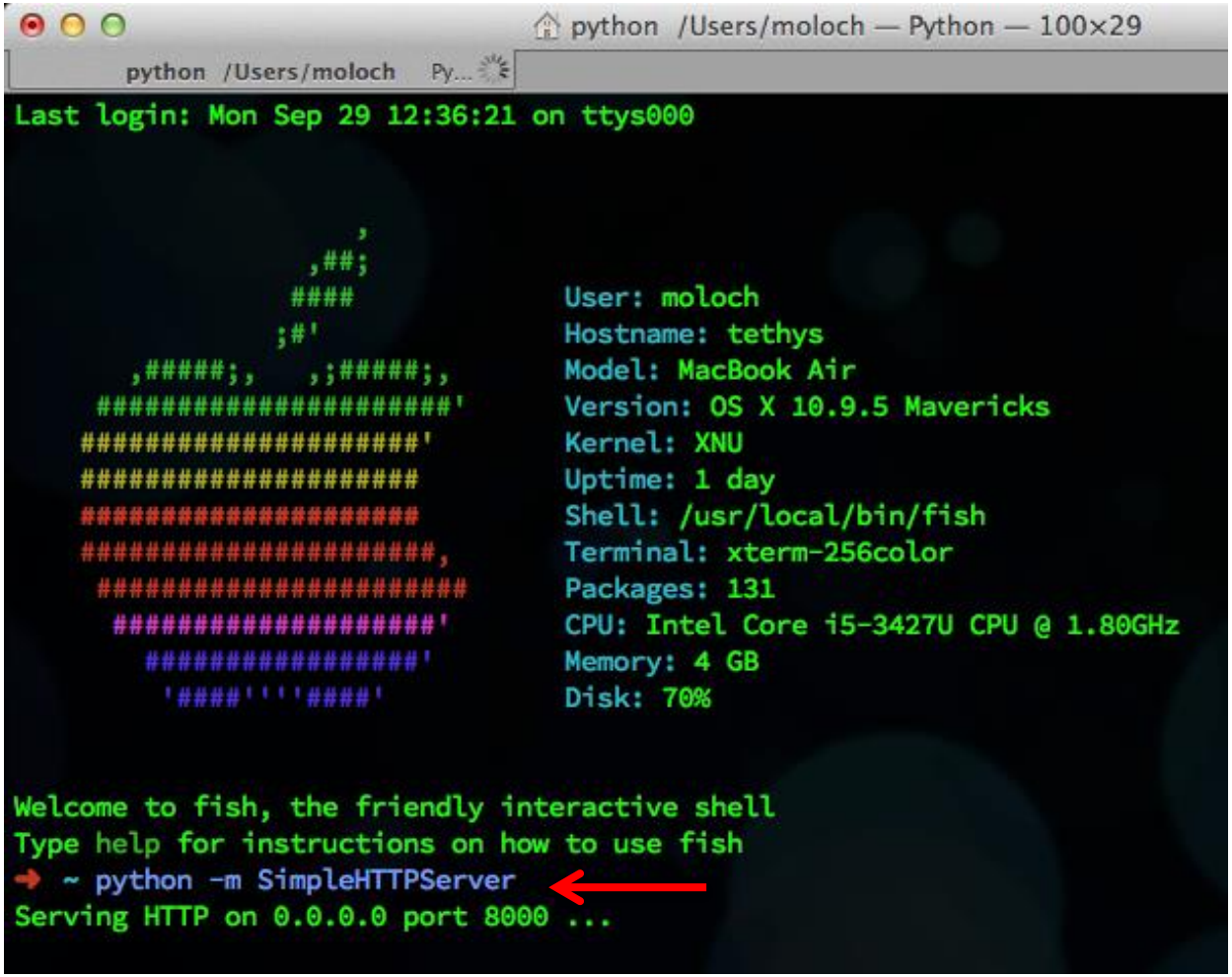
# Burp Suite Pro

## ADDING A TRUSTED ROOT CERTIFICATE



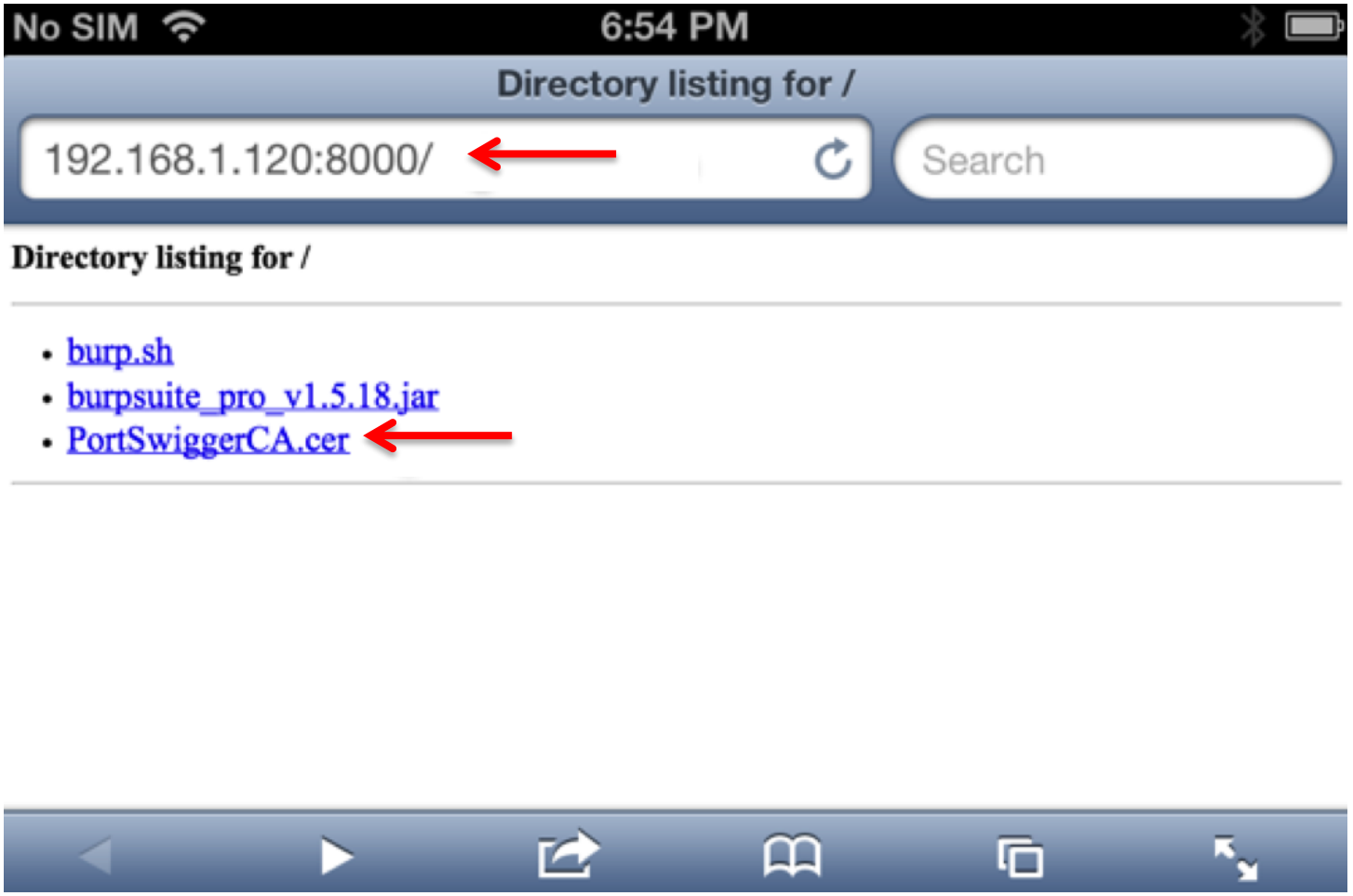
# Burp Suite Pro

## ADDING A TRUSTED ROOT CERTIFICATE



# Burp Suite Pro

## ADDING A TRUSTED ROOT CERTIFICATE



# Burp Suite Pro

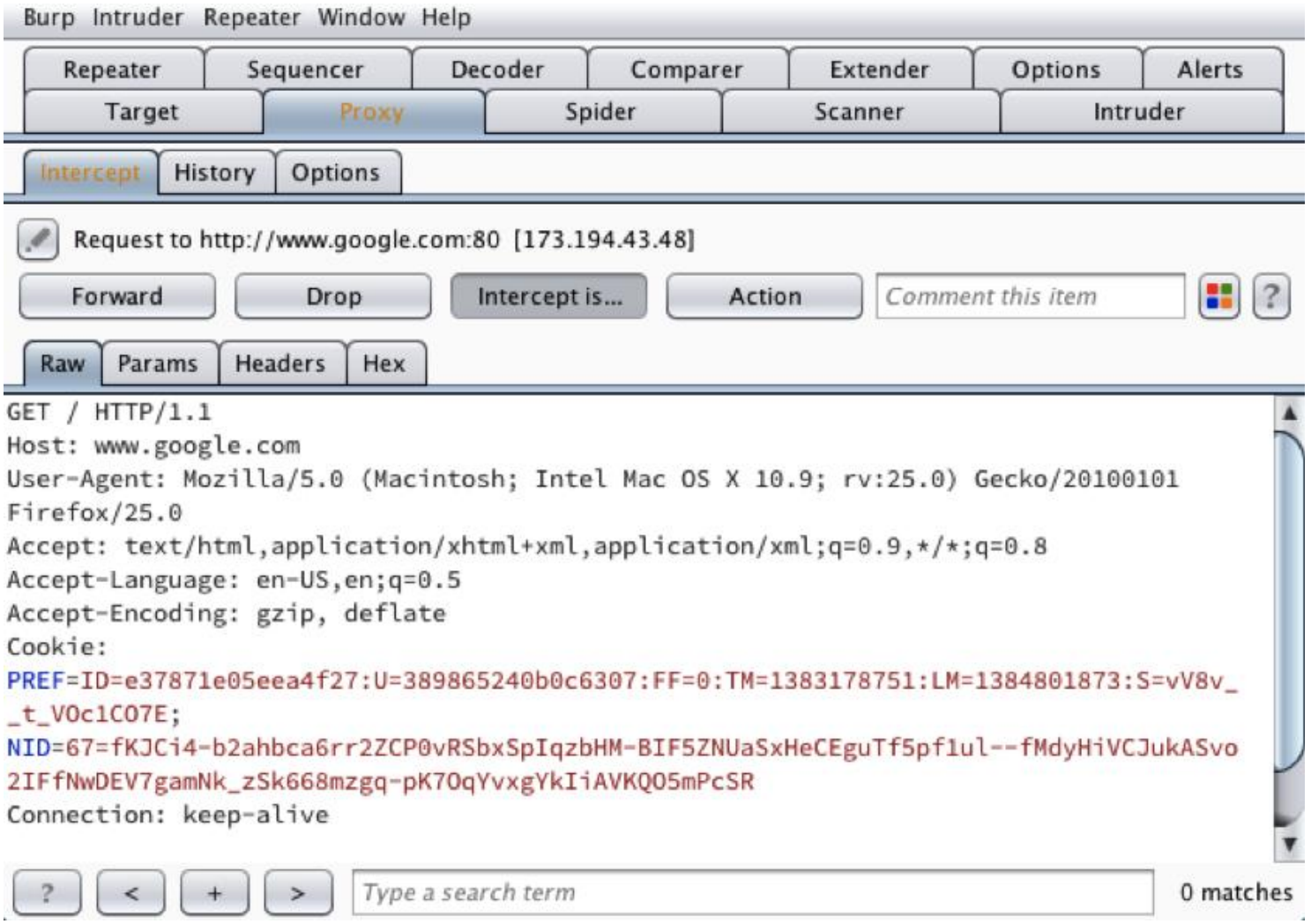
ADDING A TRUSTED ROOT CERTIFICATE

# HTTP Proxy Setup

## INTERCEPTING HTTPS TRAFFIC

# HTTP Proxy Setup

SECURE TRAFFIC INTERCEPTION

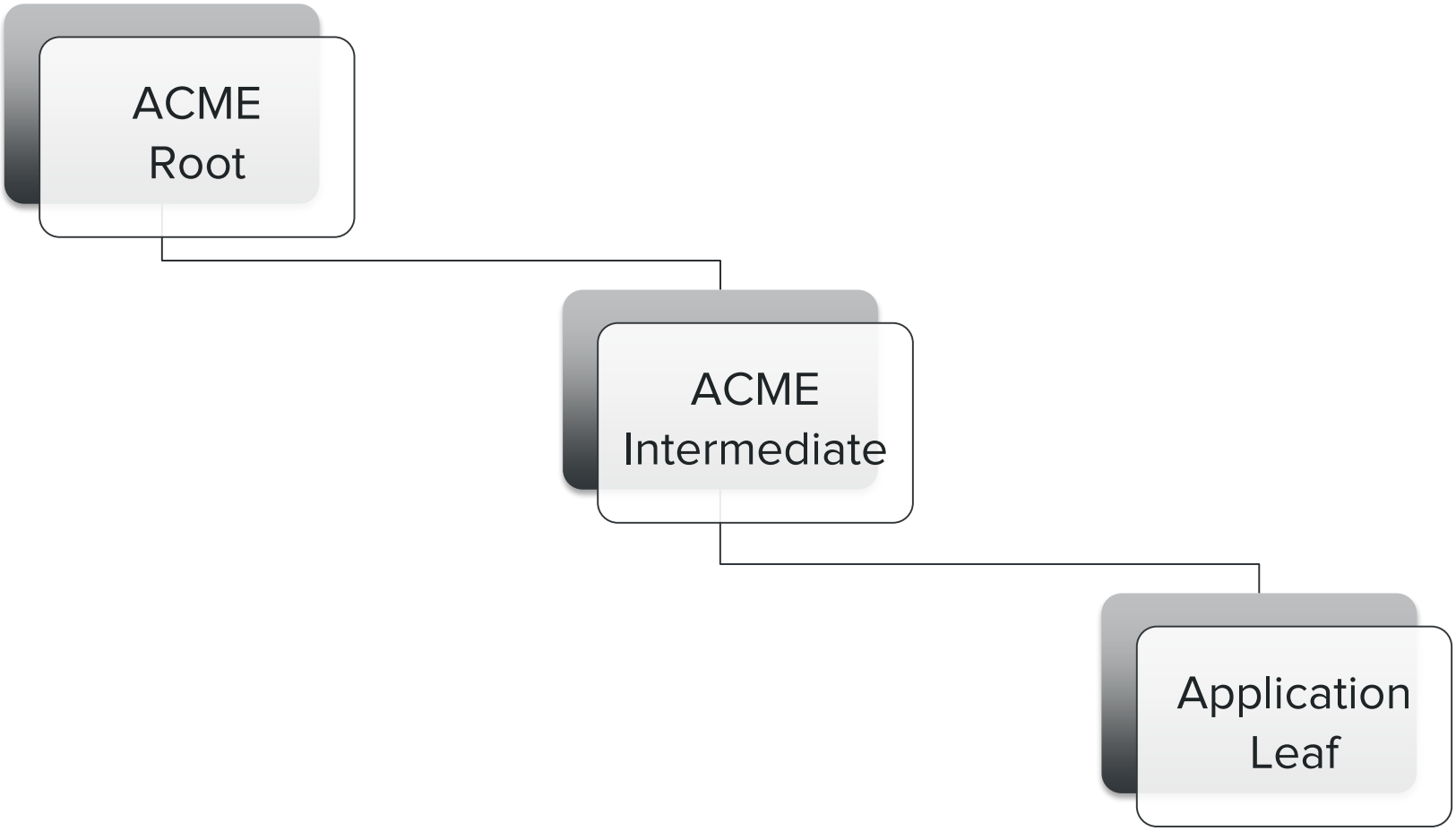


# HTTP Proxy Setup

## INTERCEPTING HTTPS TRAFFIC

# ACME Certificate Pinning

NON-BROWSER CERTIFICATE VALIDATION



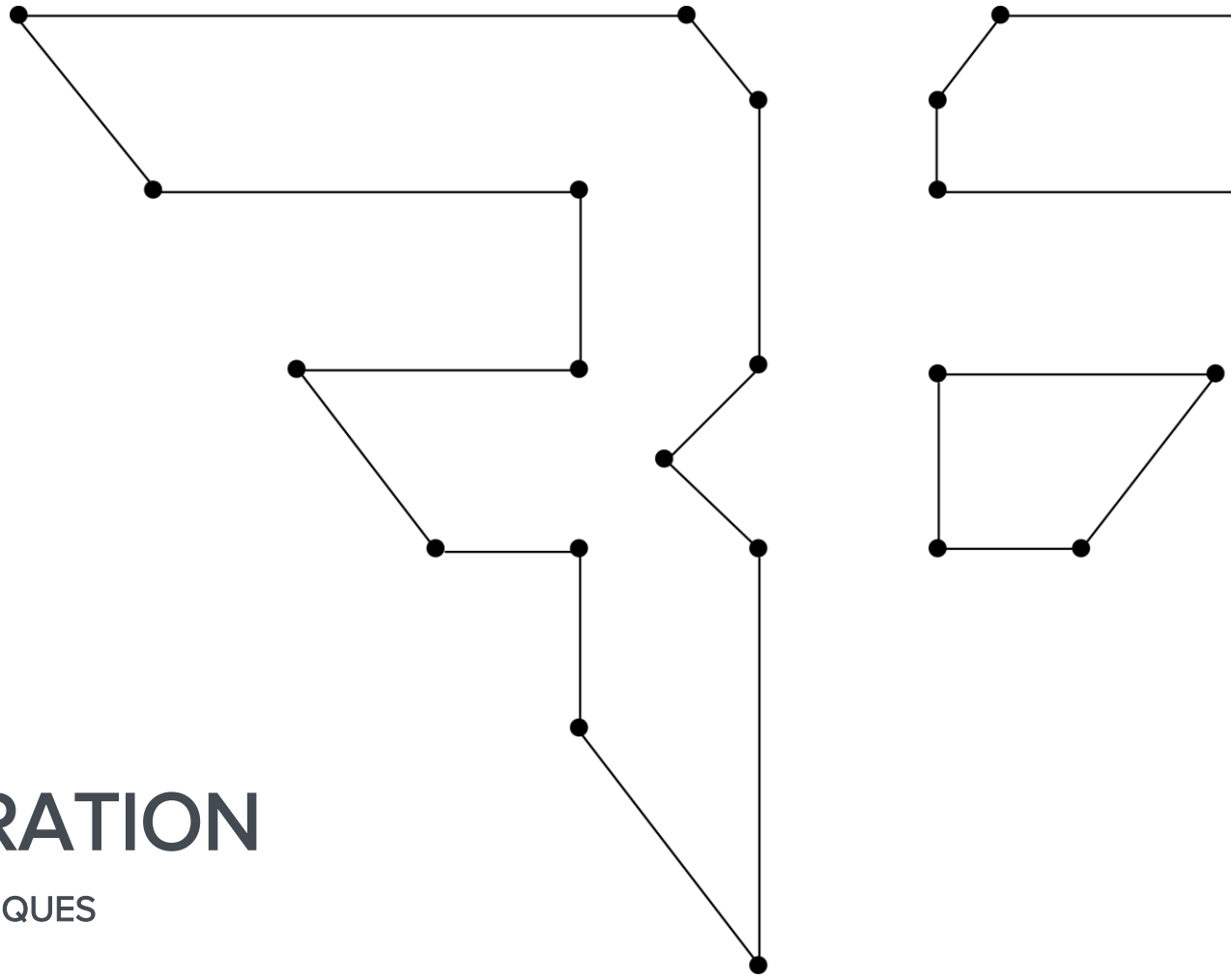
# IOS DYNAMIC ANALYSIS

## BREAKING IOS APPLICATIONS

# Operating System Security Model

## WHY WE NEED TO JAILBREAK

- Signed Binaries
  - Modifying binaries
  - Code injection
  - Runtime modification
- App Sandbox
  - Debugging
  - Filesystem access



# DEMONSTRATION

## CODE INJECTION TECHNIQUES

# APP STORE ENCRYPTION

## BREAKING IOS APPLICATIONS

# Binary Encryption

GETTING PLAINTEXT BINS

- Encrypted Binaries
  - AppStore
    - Clutch
    - Rasticrac
- No Encryption
  - Provisioned Device
  - Test Flight, etc.

# Clutch Usage

## DECRYPTING IOS BINARIES

- Open source (GitHub)
- Decrypts iOS applications and repackages them
- Saves apps in:
  - /var/root/Documents/Cracked
- Saves apps as **.ipa** files (they're just ZIPs)
- Use: clutch <app name>

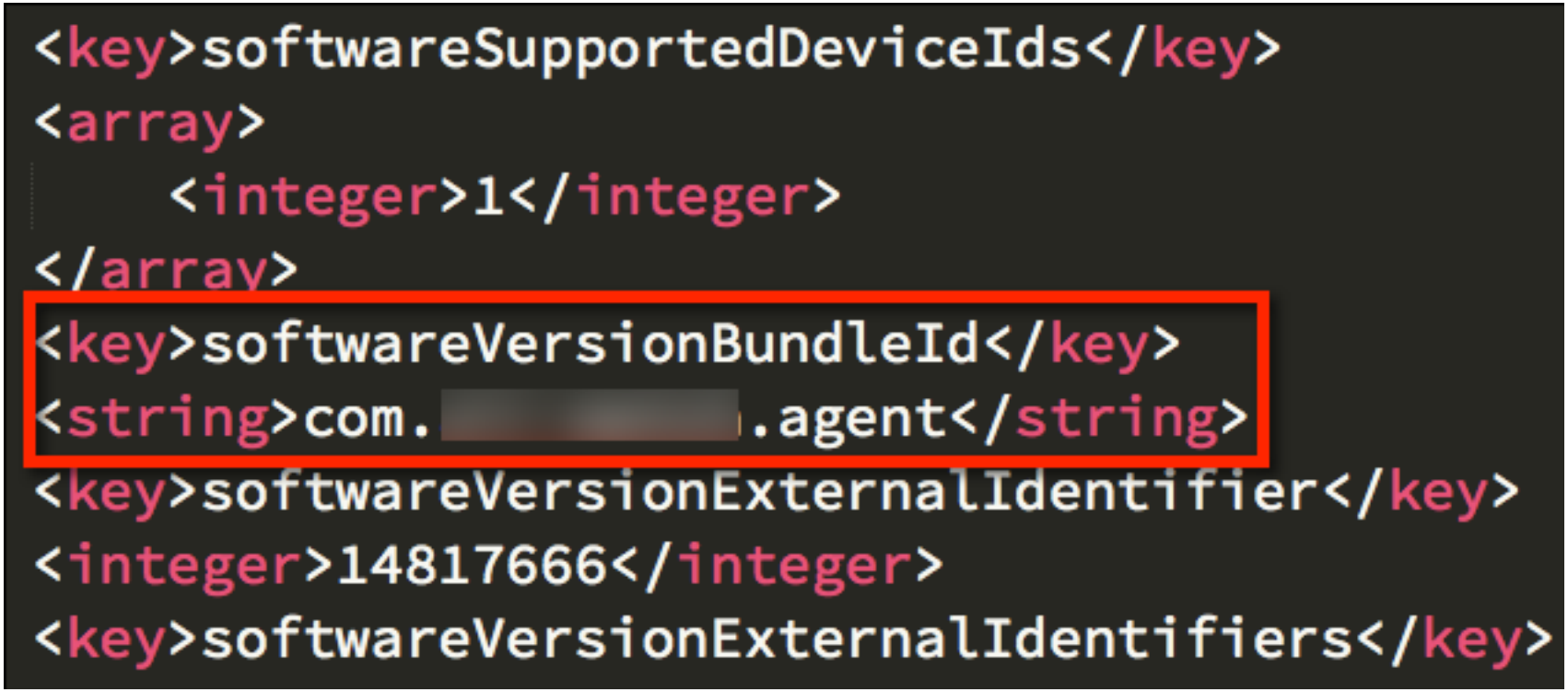
# The IPA Archive Format

NOT DELICIOUS BEER

- **Foobar.ipa**
  - **iTunesMetadata.plist**
  - iTunesArtwork
  - Payload/
    - Foobar.app
      - **Foobar**
      - …

# iTunes Metadata

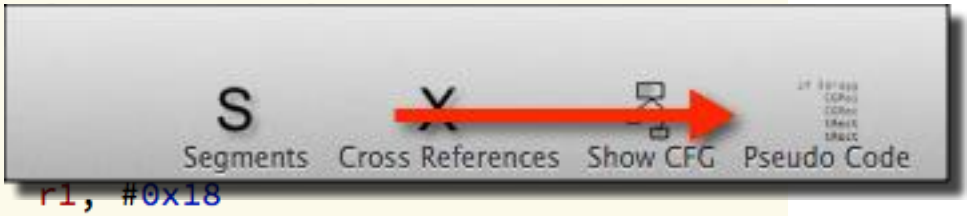
SOFTWARE VERSION BUNDLE ID



# ARM Disassembly

I AM IN YOUR BINARIES CHANGING YOUR CODE

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

                +[[redacted]CompromiseDetection jailBrokenStatus]:
0x000ddf38          push        {r4, r5, r6, r7, lr}
0x000ddf3a          add         r7, sp, #0xc
0x000ddf3c          push.w
0x000ddf40          sub.w
0x000ddf44          sub
0x000ddf46          movw
0x000ddf4a          movt        r1, #0x18
0x000ddf4e          movw        r0, #0xaa7c
0x000ddf52          movt        r0, #0x18
0x000ddf56          add         r1, pc
0x000ddf58          add         r0, pc
0x000ddf5a          ldr         r1, [r1]
0x000ddf5c          ldr         r0, [r0]
0x000ddf5e          blx         imp___symbolstub1__objc_msgSend
0x000ddf62          mov         r1, r0
0x000ddf64          movs        r0, #0x2
0x000ddf66          cmp         r1, #0x0
0x000ddf68          beq.w       0xde55a
```



# ARM Decompiler

I AM IN UR BINARIES MODIF'IN UR CODEZ

```
loc_ddf6c:
    r4 = malloc(0x18);
    r0 = 0x0;
    *(int8_t *)(r4 + 0x17) = r0;
    r1 = "\xBA\xD4\xE5\xE5\xF9\xFC\xF6\xF4\xE1\xFC\xFA\xFB\xE6\xBA\xD6\xEC\xF1\xFC\x
    do {
            *(int8_t *)(r4 + r0) = *(int8_t *)(r1 + r0) ^ 0x95;
            r0 = r0 + 0x1;
    } while (r0 != 0x17);
    r5 = malloc(0xf);
    r0 = 0x0;
    *(int8_t *)(r5 + 0xe) = r0;
    r1 = "\xBA\xE3\xF4\xE7\xBA\xF9\xFC\xF7\xBA\xF6\xEC\xF1\xFC\xF4";
    do {
            *(int8_t *)(r5 + r0) = *(int8_t *)(r1 + r0) ^ 0x95;
            r0 = r0 + 0x1;
    } while (r0 != 0xe);
    r6 = malloc(0x13);
    r0 = 0x0;
    *(int8_t *)(r6 + 0x12) = r0;
    r1 = "\xBA\xE3\xF4\xE7\xBA\xE1\xF8\xE5\xBA\xF6\xEC\xF1\xFC\xF4\xBB\xF9\xFA\xF2";
    do {
            *(int8_t *)(r6 + r0) = *(int8_t *)(r1 + r0) ^ 0x95;
            r0 = r0 + 0x1;
    } while (r0 != 0x12);
```

# XOR is Not Obfuscation

JAILBREAK DETECTION BYPASSES

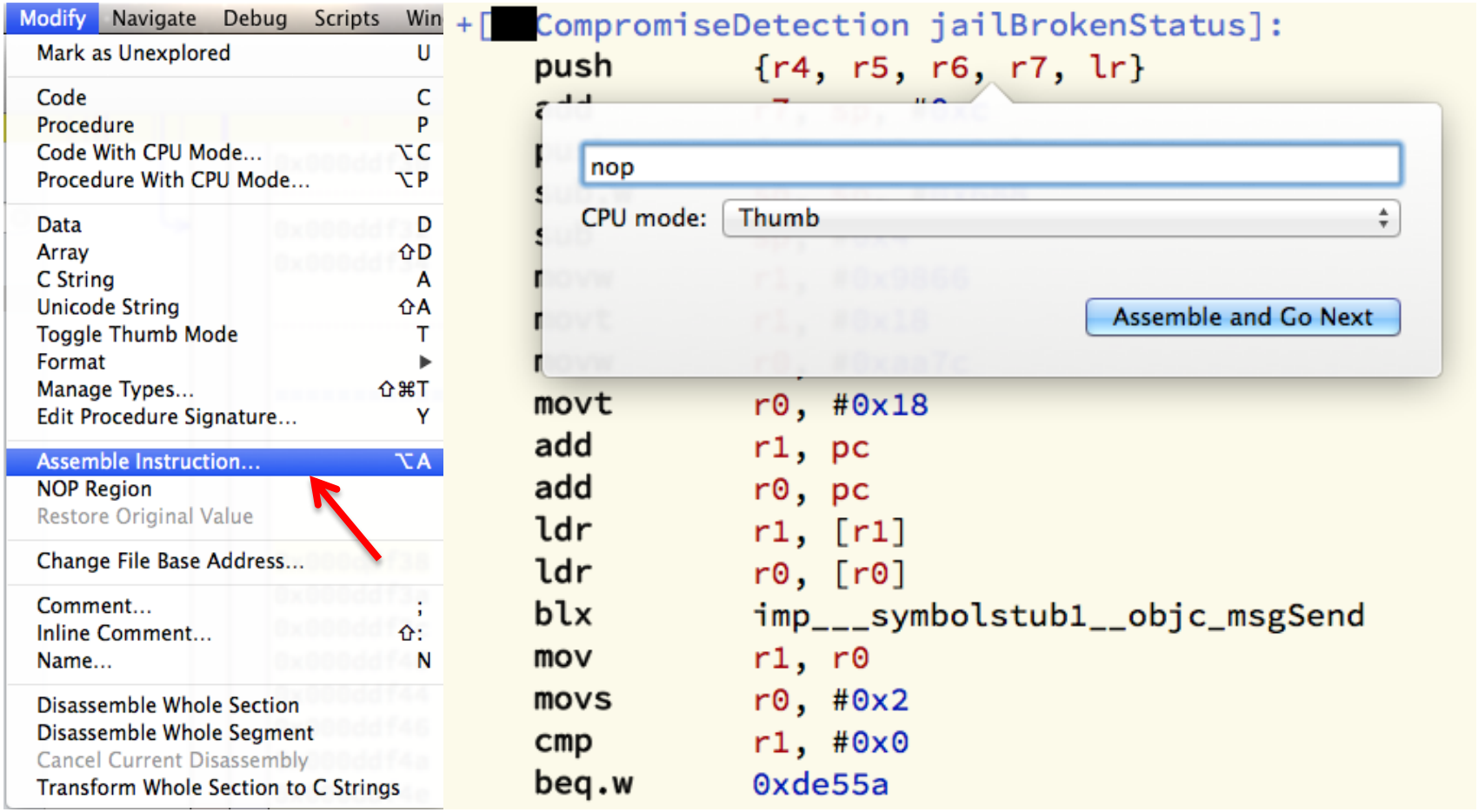
```
    do {
            *(int8_t *)(r4 + r0) = *(int8_t *)("\\xBA\\xD4\\xE5\\xE5
\xF1\\xFC\\xF4\\xBB\\xF4\\xE5\\xE5" + r0) ^ 0x95;
            r0 = r0 + 0x1;
    } while (r0 != 0x17);
    r0 = malloc(0xf);
    *(int8_t *)(r0 + 0xe) = 0x0;
```

```
In [31]: for char in a:
    foo = int(repr(char)[3:-1], 16) ^ 0x95
    out += chr(foo)
   ....:

In [32]: print out
/Applications/Cydia.app
```

```
    Pop();
    Pop();
    return r0;

loc_ddf6c:
    r0 = malloc(0x18);
    do {
            *(int8_t *)(r4 + r0) = *(int8_t *)("\\
\xF9\\xFC\\xF6\\xF4\\xE1\\xFC\\xFA\\xFB\\xE6\\xBA\
\xF4\\xBB\\xF4\\xE5\\xE5" + r0) ^ 0x95;
```

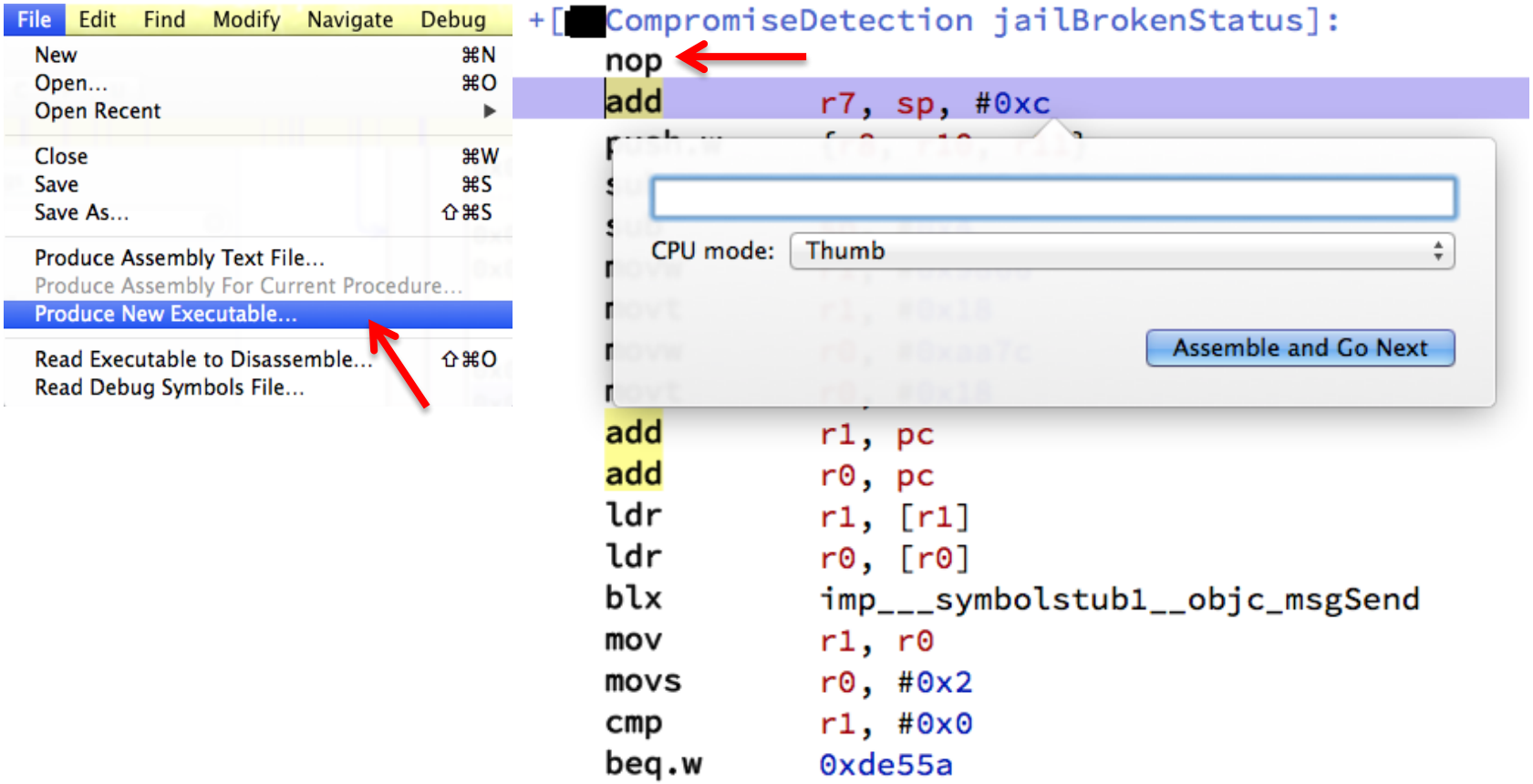
# Modifying ARM Assembly

ASSEMBLE INSTRUCTION



# Modifying ARM Assembly

## PRODUCE NEW EXECUTABLE

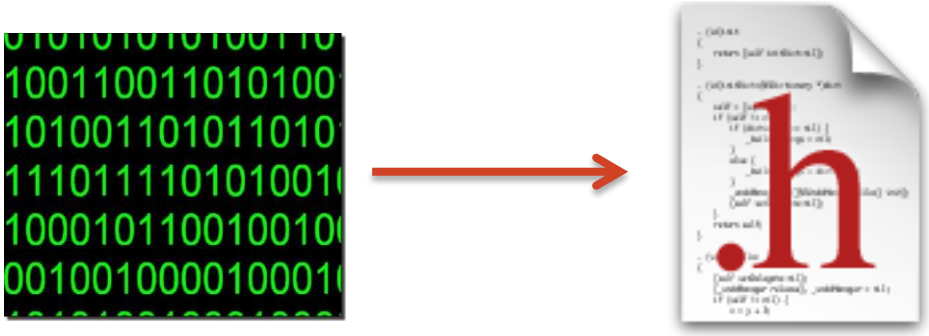


# OBJECTIVE-C HEADERS

STATIC ANALYSIS

# Class Dump

## OBJECTIVE-C CLASS INTERFACES



```
#import <XXUnknownSuperclass.h> // Unknown library

@class NSString;

@interface XXasymEncryptor : XXUnknownSuperclass {
	NSString* _publicKeyID;
	NSString* _privateKeyID;
}
@property(copy, nonatomic) NSString* privateKeyID;
@property(copy, nonatomic) NSString* publicKeyID;
-(id)privateQueryDict;
-(id)publicQueryDict;
-(void)decryptWithPrivateKey;
-(void)encryptWithPublicKey;
-(void)KeysPlease;
-(id)decryptData:(id)data;
-(id)encryptData:(id)data;
-(id)init;
@end
```

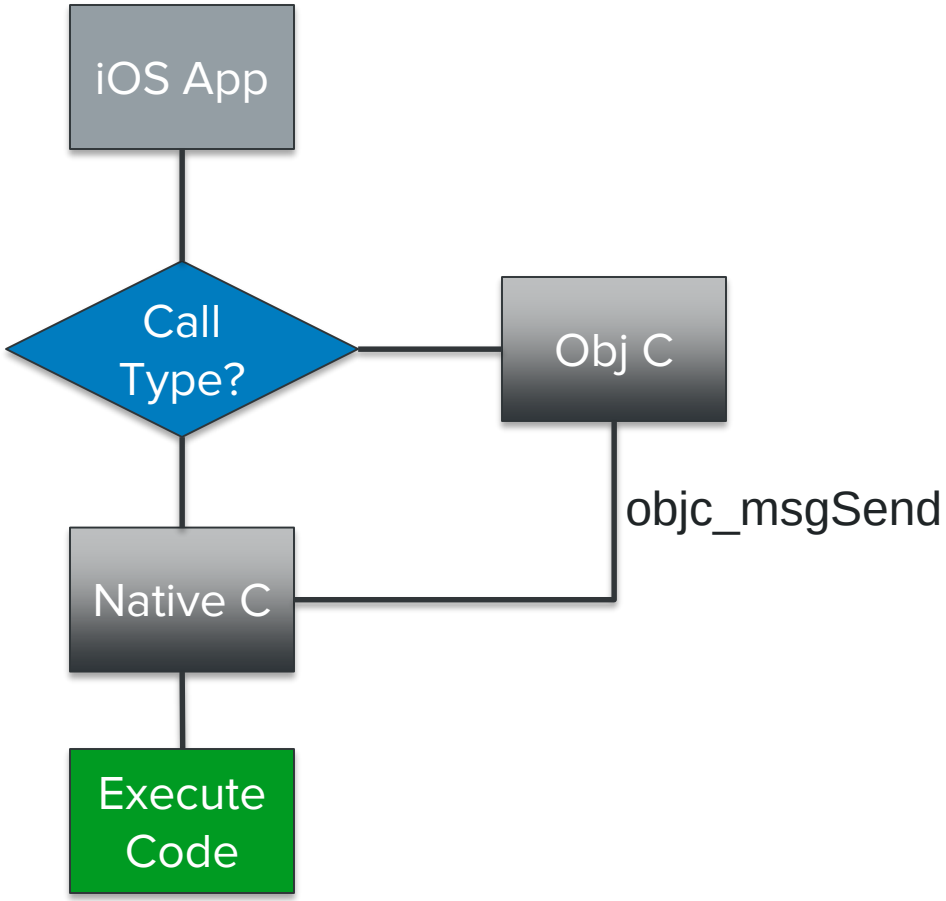
```
#import <XXUnknownSuperclass.h> // Unknown library

@class NSString;

@interface XXasymEncryptor : XXUnknownSuperclass {
    NSString* _publicKeyID;
    NSString* _privateKeyID;
}
@property(copy, nonatomic) NSString* privateKeyID;
@property(copy, nonatomic) NSString* publicKeyID;
-(id)privateQueryDict;
-(id)publicQueryDict;
-(void)decryptWithPrivateKey;
-(void)encryptWithPublicKey;
-(void)KeysPlease;
-(id)decryptData:(id)data;
-(id)encryptData:(id)data;
-(id)init;
@end
```

# MOBILE SUBSTRATE

## CODE INJECTION TECHNIQUES

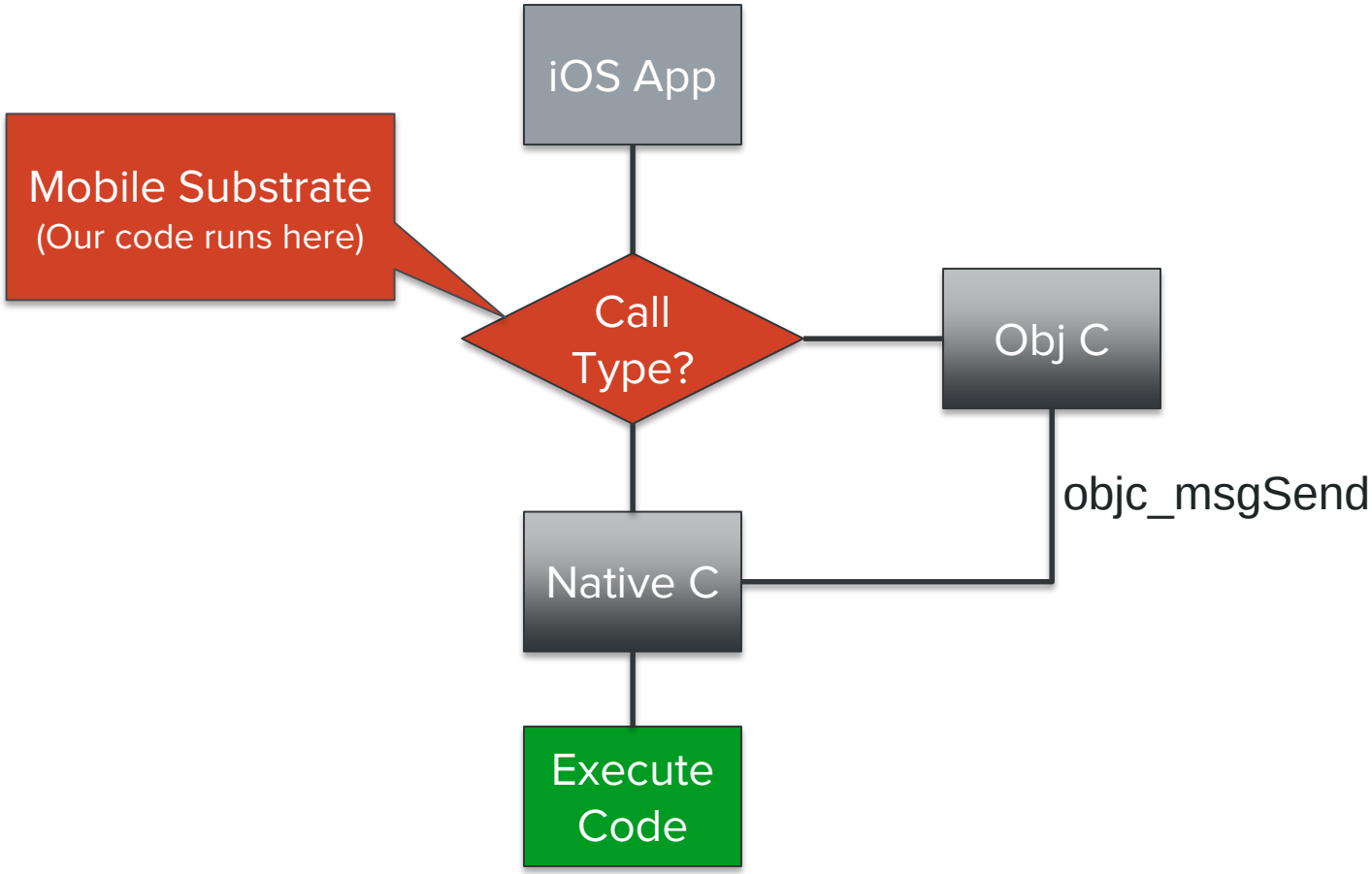
# The Objective-C Runtime

## MESSAGE PASSING



# The Objective-C Runtime

## MESSAGE PASSING



# Jailbreak Detection Code

## BYPASSING COMMON DETECTION METHODS

- Fork()
- Stat() / Lstat()
  - Cydia
  - /apt/
  - Etc
- dyld_count()
- dyld_get_image_name()

# Jailbreak Detection Code

## BYPASSING COMMON DETECTION METHODS

```
@class NSString;

@interface DeviceSecurity: {

        BOOL _jbstatus;

}

@property(assign, nonatomic) BOOL jbstatus;

+(BOOL)isJailbroken;

@end
```

# Theos + Logos + Mobile Substrate

## CLASS AND METHOD HOOKING

```
#import "substrate.h"

%hook DeviceSecurity

-(BOOL) isJailbroken {
    %log;      // Logos built-in logging
    return NO; // Return FALSE
}

%end
```

# Certificate Bypasses

## “TRUST ME” BYPASS

```
#import "substrate.h"

/* New function definition */
OSStatus new_SecTrustEvaluate(SecTrustRef trust,
                              SecTrustResultType *result) {
    *result = kSecTrustResultProceed;
    return errSecSuccess;
}

%ctor {
/* Hook the function */
MSHookFunction((void *)SecTrustEvaluate,
               (void *)new_SecTrustEvaluate,
               (void **)&original_SecTrustEvaluate);
}
```

# CYCRIPT

RUNTIME MODIFICATION

# Cycript is Black Magic

## RUNTIME MODIFICATION TECHNIQUES

- JavaScript REPL
- JavaScript + Cycript language extensions
- Objective-C runtime is merged into the REPL
- Attach to running apps

# Cycript Basics

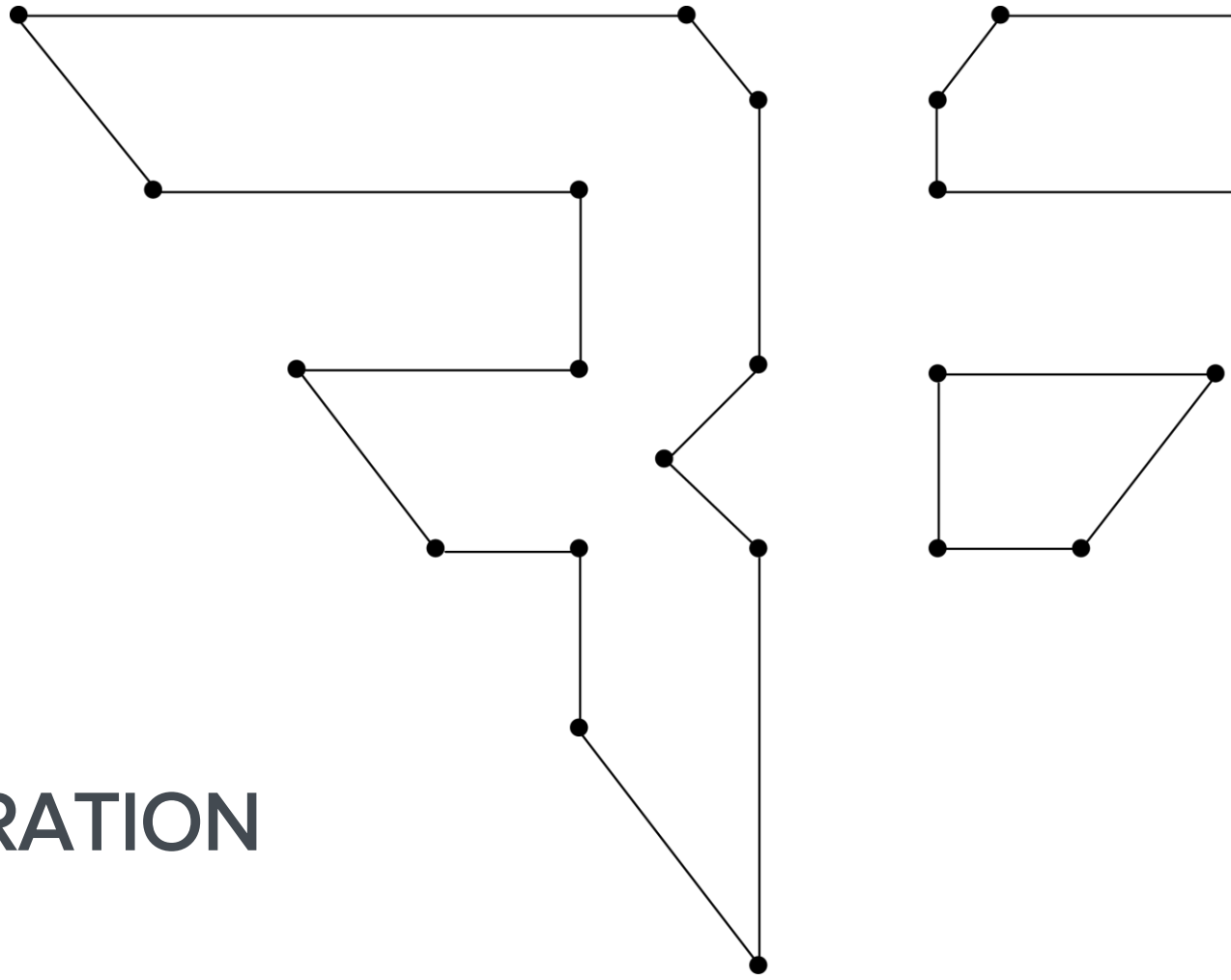
## ATTACHING TO A PROCESS

```
iphone:~root# cycript –p AlienBlue

cy# UIApp

@"<UIApplication: 0x8ba2c0>"

cy# UIApp.keyWindow.delegate

@"<CustomNavigationController: 0x836900>"

cy# ui(UIApp.keyWindow, "Foobar")

<UILabel: 0x82f0d0; frame = (132 12; 55 21); text = 'Foobar';
clipsToBounds = YES; userInteractionEnabled = NO; layer =
<CALayer: 0x82f190>>
```

# Cycript Basics

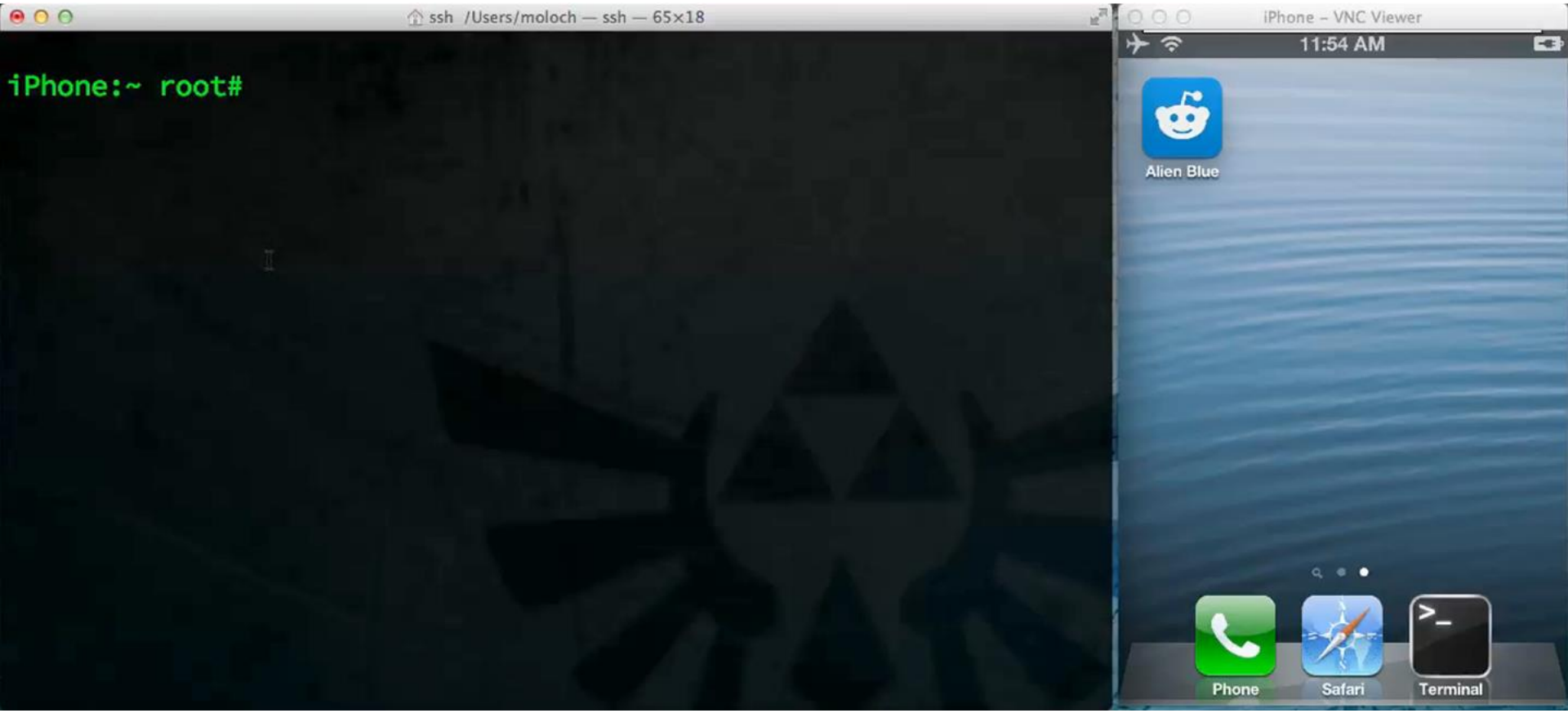
## ATTACHING TO A PROCESS

```
cy# var label = new Instance(0x82f0d0)

@"<UILabel: 0x82f0d0; frame = (132 12; 55 21); text =
'Reddits'; clipsToBounds = YES; userInteractionEnabled = NO;
layer = <CALayer: 0x82f190>>”

cy# label.text

@"Foobar"

cy# label.text = @"Barfoo"

@"Barfoo"
```



# DEMONSTRATION

CYCRIPT IN ACTION


ssh /Users/moloch — ssh — 65×18
iPhone:~ root#
iPhone – VNC Viewer
11:54 AM
Alien Blue
Phone
Safari
Terminal

# ISPY

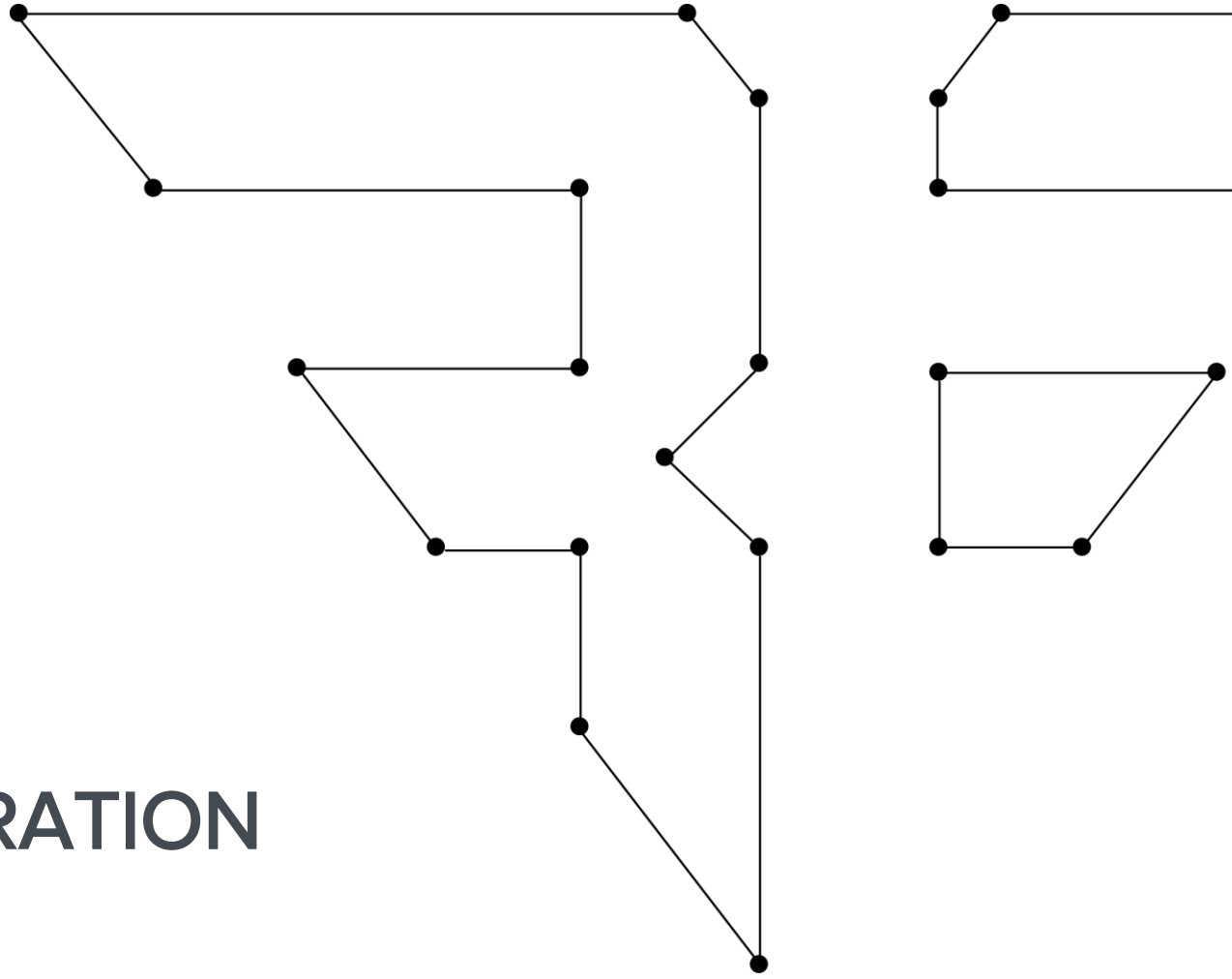
WELCOME TO THE FUTURE

# iSpy Assessment Framework

YOUR ONE-STOP-SHOP FOR IOS HACKING

## Under Active Development

- Easy to use Web GUI
- Class dumps / Instance tracking
- Automatic jailbreak-detection bypasses
- Automatic SSL certificate pinning bypasses
- Re-implemented objc_msgSend
- Automatic detection of vulnerable function calls
- Easy to use soft-breakpoints
- *More on the way!*



# DEMONSTRATION

ISPY IN ACTION

# iSpy

## GITHUB REPOS

https://github.com/BishopFox

# COUNTER-MEASURES

PROTECTING IOS APPLICATIONS

# Defense Against the Dark Arts

ON DISK & IN MEMORY

1. Security code
    1. Assembly and/or C
    2. Inline functions
    3. Objective-C obfuscation
2. Certificate pinning
    1. Whitelist root authorities
3. Change release

- Metaforic – Commercial
- AppMinder – BSD Licensed
    a) http://appminder.nesolabs.de/

# AppMinder

FREE ‘N EASY OBFUSCATION

```
#if !(TARGET_IPHONE_SIMULATOR)
  __attribute__ ((always_inline)) static void
MKJKqOBovPApokJQDajjRvdTEk (void)
{
        asm volatile ("b #4;pop {r0-r15};bx lr;mov r0, #63;mov r12, #256;asr r12,
#7;b #2;pop {r0-r15};sub r2, r2, r2;mov r0, r2;mov r0, r0;svc 0x80;mov r1, #1;b
#1;cmp r0, r10;cmp r0, r1;itt ne;movne r12, #1;swine 0x11;mov r3, r0;lsl r3, r3,
#3;add r3, r0;add r3, pc;bx r3;add sp, #120;ldmia sp, {r0-r15};bx lr;mov r3, r3;sub
r2, r2, r2;mov r0, r2;mov r12, #2;mov r2, r2;mov r0, #31;b #4;pop {r0-r15};bx lr;svc
0x80;sub r1, r1, r1;b #1;cmp r0, r10;mov r3, r1;b #2;push {r0-r12};add r3, r3, #1;b
#2;stmdb sp!, {r0-r12};cmp r0, r3;bne #1;b #4;mov r12, #1;swi 0x11;mov r2, #12;sub
r2, r2, r0;add r2, pc;bx r2;mov r0, #1;mov r12, #1;svc 0x80;" : : : "r0", "r1", "r2",
"r3", "r4", "r12", "cc", "memory");
}
#endif
```

**STEP 2:** Call the C function, where you would like to implement the jailbreak detection (e.g. in *didFinishLaunchingWithOptions:*):

```
#if !(TARGET_IPHONE_SIMULATOR)
MKJKqOBovPApokJQDajjRvdTEk ();
#endif
```

# ANDROID APP SECURITY

JAVA JAVA JAVA JAVA JAVA JAVA

# Android Prerequisites

WHAT YOU NEED TO START

- Linux / Mac / Windows
- HTTP Proxy
  - Burp Suite Pro ($300)
  - MitM Proxy ($0)
- ADT Eclipse Bundle
  - Substrate plug-in
- Procyon *(Java decompiler)*
- Dex2jar *(bytecode converter)*
- Rooted Device
  - Cydia Substrate

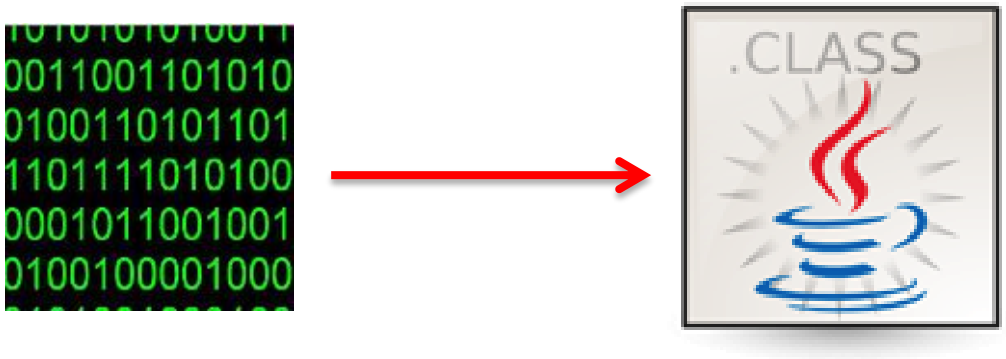
# Android Application Packages

GOOGLE PLAY STORE

- APKs are signed
  - Not encrypted
- APK Extractor
- Direct Download

# Decompiling Dalvik Bytecode

## GETTING SOURCE CODE



# Dex2jar + Procyon

## TWO STEP DECOMPILATION

```
$ dex2jar Foobar.apk

dex2jar foobar.apk -> Foobar-dex2jar.jar

$ procyon –jar Foobar-dex2jar.jar –o src/
Decompiling com/foobar/Parser...
Decompiling com/foobar/XMLWriter...
```

# Decompiled Android Code

SOURCE SORTA

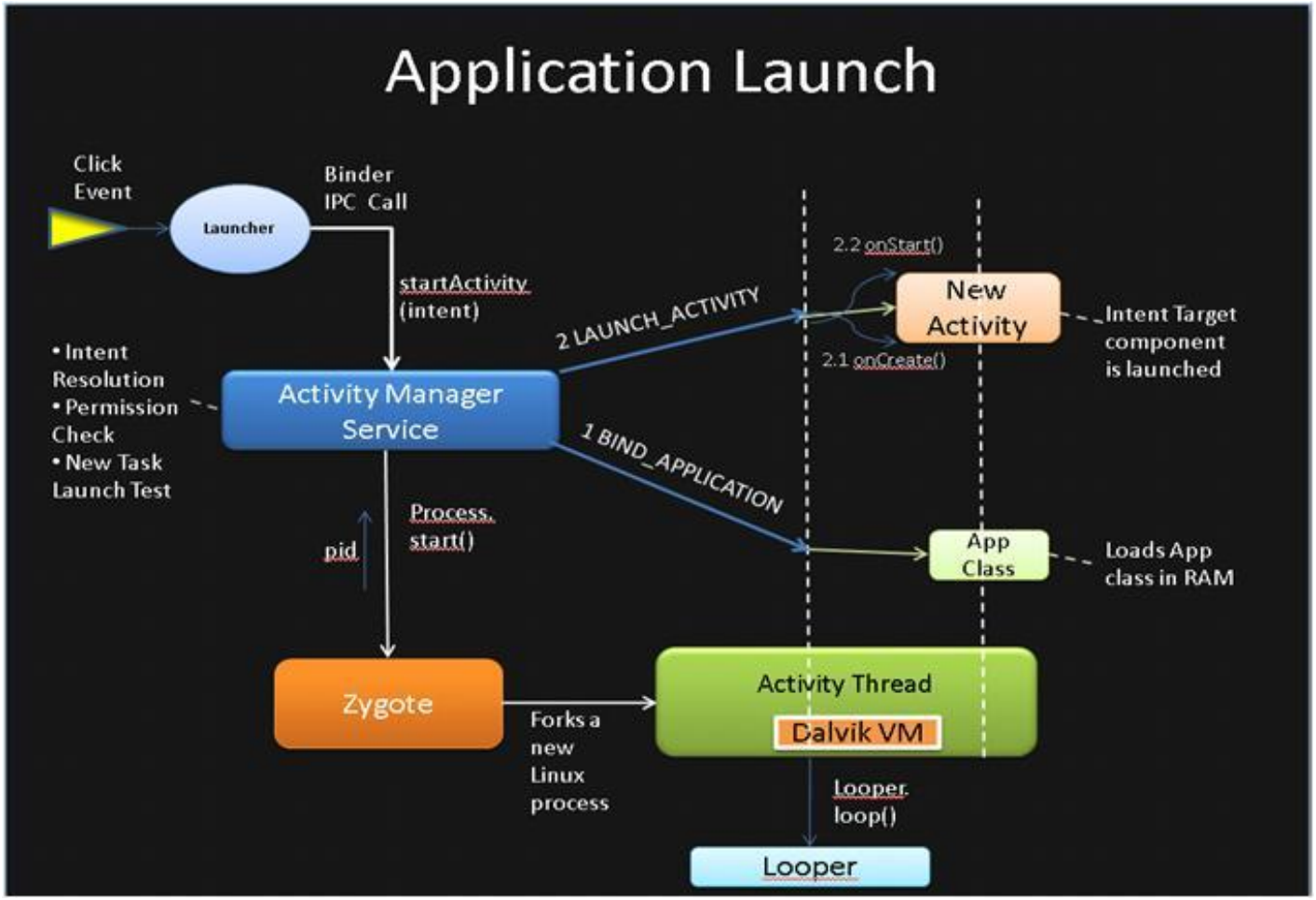
```
public MainActivity() {
    super();
    this.d = a.a();
    this.l = new IntentFilter("                                    .ACTION_DOWNLOA
    this.m = new IntentFilter("                                    .CHECK_THEME_UP
    this.n = new BroadcastReceiver() {
        public void onReceive(final Context context, final Intent intent)
                                                    f.c.a(n.confirm_restart_title, n.
                public void run() {
                    MainActivity.this.x();
                }
            }).show(MainActivity.this.getSupportFragmentManager(), "confir
        }
    };
```

# DYNAMIC ANALYSIS

ON ANDROID

# The Android Zygote

APPLICATION INSTANTIATION



# Cydia Substration

ANDROID FUNCTION HOOKING

Class to hook

```
/* Get the class we want to hook */
Class _class = Class.forName("android.net.http.AndroidHttpClient");

/* Get the method we want to hook, in this case it's the constructor */
Method execMethod = _class.getMethod(
        "execute", HttpUriRequest.class, HttpContext.class);
```

Method to hook

# Cydia Substration

ANDROID FUNCTION HOOKING

```
/* Our method alteration code */
MS.hookMethod(_class, execMethod, new MS.MethodAlteration() {
    public Object invoked(Object _this, Object... args) throws Throwable
    {
        /* Cast arguments into useful types */
        HttpUriRequest request = (HttpUriRequest) args[0];
        HttpContext context = null;
        if (args[1] != null) {
            context = (HttpContext) args[1];
        }

        /* Log pertinent information */
        Log.d("HttpInterceptor", "[" + request.getMethod() + "] -> " + r

        /* Call the original method */
        return invoke(_this, request, context);
    }
});
```

# Cydia Substration

ANDROID FUNCTION HOOKING

```
/* Our method alteration code */
MS.hookMethod(_class, execMethod, new MS.MethodAlteration() {
    public Object invoked(Object _this, Object... args) throws Throwable
    {
        /* Cast arguments into useful types */
        HttpUriRequest request = (HttpUriRequest) args[0];
        HttpContext context = null;
        if (args[1] != null) {
            context = (HttpContext) args[1];
        }

        /* Log pertinent information */
        Log.d("HttpInterceptor", "[" + request.getMethod() + "] -> " + r

        /* Call the original method */
        return invoke(_this, request, context);
    }
});
```

# Cydia Substration

ANDROID FUNCTION HOOKING

```
/* Our method alteration code */
MS.hookMethod(_class, execMethod, new MS.MethodAlteration() {
    public Object invoked(Object _this, Object... args) throws Throwable
    {
        /* Cast arguments into useful types */
        HttpUriRequest request = (HttpUriRequest) args[0];
        HttpContext context = null;
        if (args[1] != null) {
            context = (HttpContext) args[1];
        }

        /* Log pertinent information */
        Log.d("HttpInterceptor", "[" + request.getMethod() + "] -> " + r

        /* Call the original method */
        return invoke(_this, request, context);
    }
});
```

# Cydia Substration

ANDROID FUNCTION HOOKING

```
/* Our method alteration code */
MS.hookMethod(_class, execMethod, new MS.MethodAlteration() {
    public Object invoked(Object _this, Object... args) throws Throwable
    {
        /* Cast arguments into useful types */
        HttpUriRequest request = (HttpUriRequest) args[0];
        HttpContext context = null;
        if (args[1] != null) {
            context = (HttpContext) args[1];
        }

        /* Log pertinent information */
        Log.d("HttpInterceptor", "[" + request.getMethod() + "] -> " + r

        /* Call the original method */
        return invoke(_this, request, context);
    }
});
```

# Cydia Substration

ANDROID FUNCTION HOOKING

```
/* Our method alteration code */
MS.hookMethod(_class, execMethod, new MS.MethodAlteration() {
    public Object invoked(Object _this, Object... args) throws Throwable
    {
        /* Cast arguments into useful types */
        HttpUriRequest request = (HttpUriRequest) args[0];
        HttpContext context = null;
        if (args[1] != null) {
            context = (HttpContext) args[1];
        }

        /* Log pertinent information */
        Log.d("HttpInterceptor", "[" + request.getMethod() + "] -> " + r

        /* Call the original method */
        return invoke(_this, request, context);
    }
});
```

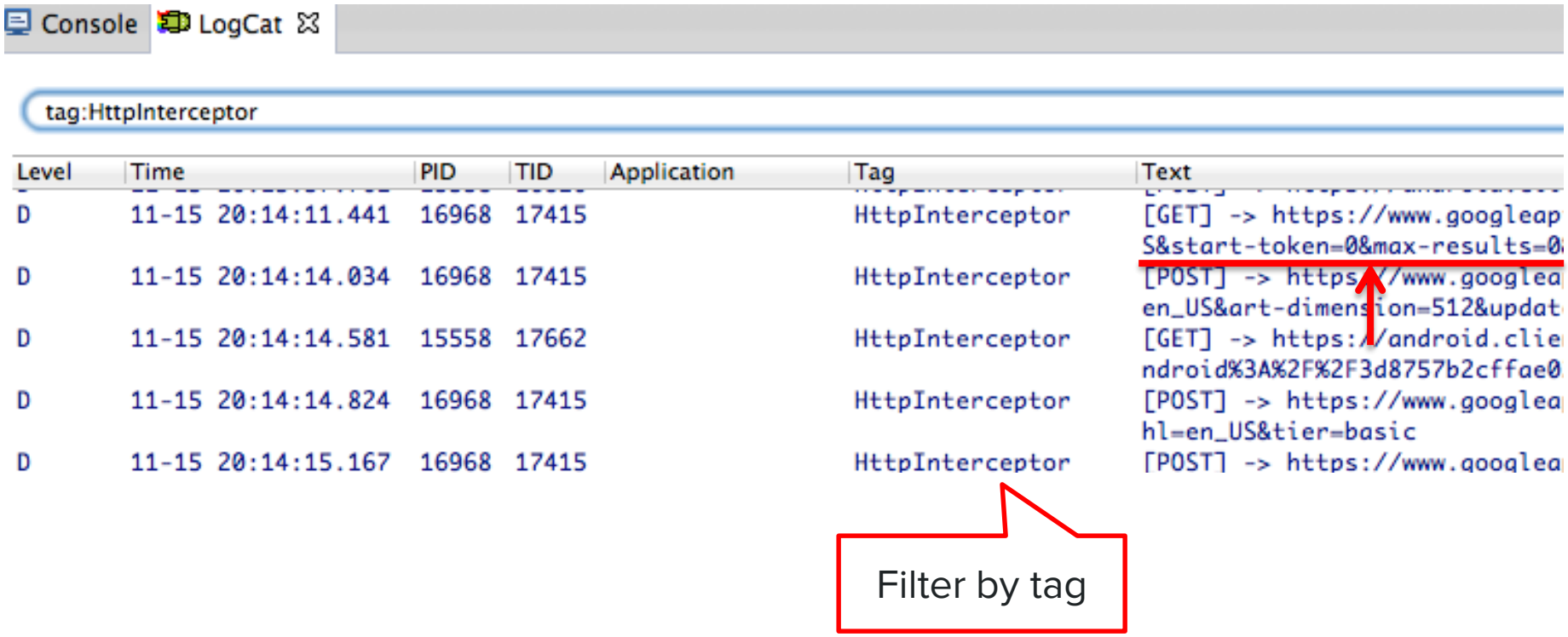
# Cydia Substration

ANDROID FUNCTION HOOKING

```
/* Our method alteration code */
MS.hookMethod(_class, execMethod, new MS.MethodAlteration() {
    public Object invoked(Object _this, Object... args) throws Throwable
    {
        /* Cast arguments into useful types */
        HttpUriRequest request = (HttpUriRequest) args[0];
        HttpContext context = null;
        if (args[1] != null) {
            context = (HttpContext) args[1];
        }

        /* Log pertinent information */
        Log.d("HttpInterceptor", "[" + request.getMethod() + "] -> " + r

        /* Call the original method */
        return invoke(_this, request, context);
    }
});
```

# Android LogCat

RESULTS



# COUNTER-MEASURES

PROTECTING ANDROID APPLICATIONS

# Android Application Security

## DREAMING OF ELECTRIC SHEEP

- Security subroutines
  - C / C++ (NDK)
  - Assembly
  - Inline functions
- Avoid kernel calls if possible
- Java bytecode obfuscation
- Certificate pinning

# CONCLUSIONS

MOBILE APPLICATION SECURITY

# Management Not Security

BYOD VERTICAL CLOUD INTEGRATION WITH APT SYNERGY

- MDM is Mobile Device *Management*
- Client-side enforcement
- Devices lie

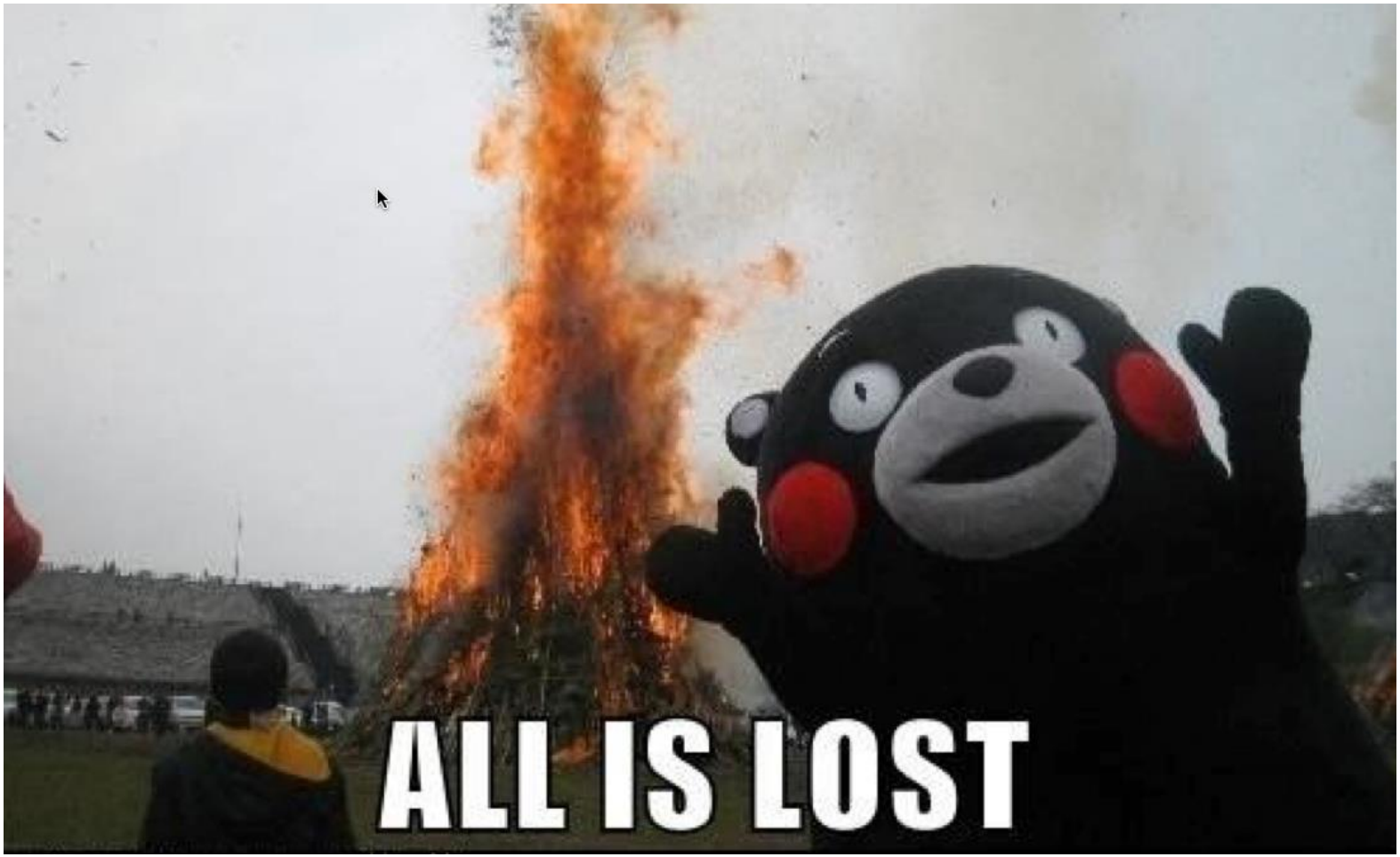
# Mobile Security

IN A NUT SHELL …

- Client-side enforcement
  - Your architecture is probably broken, fix that instead
- If the business model dictates that you must…
  - Perhaps the revenue model depends on it
  - Perhaps you have to integrate with legacy code
  - Perhaps there’s some other crazy reason for on-device security

# Client-side Enforcement

YOU CAN TRY, BUT IT IS NOT GOING TO WORK …



# Good Hunting!


MIS TRAINING INSTITUTE

BISHOP FOX®

# Contact Us

**bishopfox.com**

**contact@bishopfox.com**

**@bishopfox on the Twitters**

**github.com/BishopFox**

***We're hiring!***

BISHOP FOX®

# Image Citations

- Internet Image: LobStoR/Wikimedia
- iPhone Image: Zach Vega/Wikimedia
- Hacker Image: chanpipat/FreeDigitalPhotos.net
- Lock Image: Stuart Miles/FreeDigitalPhotos.net
- Binary Image: noegranado/Flickr Creative Commons