iSECpartners
part of nccgroup

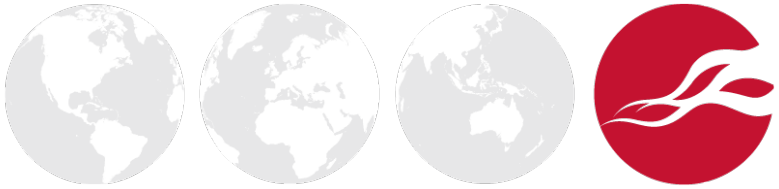
When Security Gets in the Way

PenTesting Mobile Apps That Use Certificate Pinning

Justine Osborne

Alban Diquet

black hat USA 2012



Outline

iSECpartners part of nccgroup

What is Certificate Pinning ?

- Definition and Background
- Consequences for Mobile Blackbox Testing

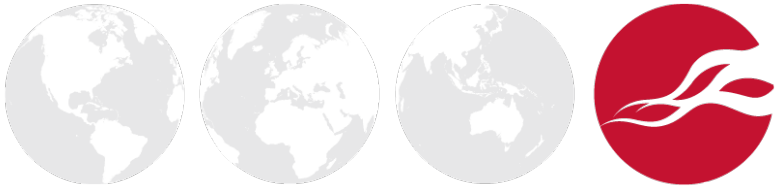
iOS

- Certificate Pinning Within an iOS App
- Intercepting the App's Traffic: MobileSubstrate Extension

Android

- Certificate Pinning Within an Android App
- Intercepting the App's Traffic: Custom JDWP Debugger

Conclusion



Outline

iSECpartners part of nccgroup

What is Certificate Pinning ?

- Definition and Background
- Consequences for Mobile Blackbox Testing

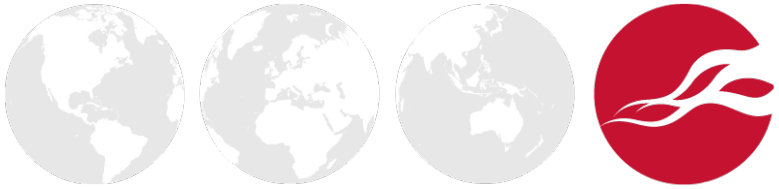
iOS

- Certificate Pinning Within an iOS App
- Intercepting the App's Traffic: MobileSubstrate Extension

Android

- Certificate Pinning Within an iOS App
- Intercepting the App's Traffic: Custom JDWP Debugger

Conclusion



Certificate Pinning

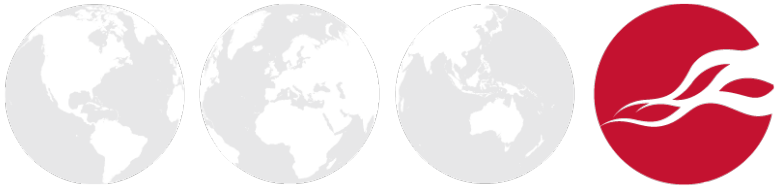
iSECpartners part of nccgroup

Hard-code in the client the certificate known to be used by the server

- Pin the server's certificate itself
  - Takes the CA system out of the equation
- Pin the CA certificate used to sign the server's certificate
  - Limit trust to certificates signed by one CA or a small set of CAs

Significantly reduces the threat of a rogue CA and of CA compromise

- Implemented in Chrome 13 for Google services



Certificate Pinning in Mobile Apps

iSECpartners part of nccgroup

Mobile is the ideal platform to implement certificate pinning

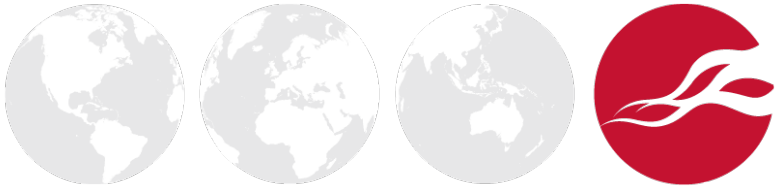
- A mobile App only needs to connect to a small set of servers
- The App's developers write the client-side code

A small list of trusted CA certificates can be included in the App itself

- The device's trust store is completely ignored

Certificate pinning is already being deployed

- Chrome for Android, Twitter, Cards.io…



Mobile Blackbox Testing

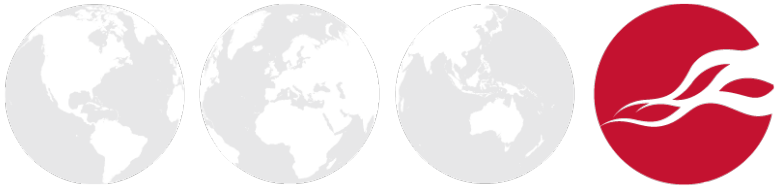
iSECpartners part of nccgroup

Some of the tester's tasks:

- Reversing the binary
- Analyzing the App's behavior at runtime (File I/O, IPC, etc…)
- **Intercepting the App's network traffic using a proxy**

The tester's proxy has to masquerade as the server

- Requires adding the proxy's CA certificate to the device trust store
- This **will not work** if the App does certificate pinning



What This Presentation is About

iSECpartners part of nccgroup

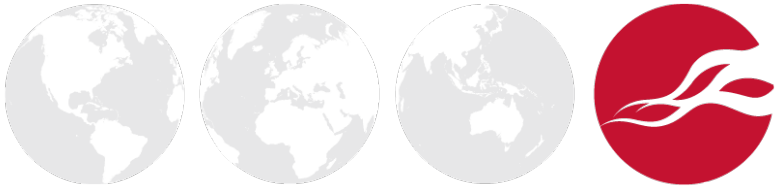
No simple solutions to defeat certificate pinning:

- Decompile the App's package/binary
- Change the certificate(s) ? Patch SSL validation methods ?
- Re-package and side-load the new binary

Blackbox assessments are usually short projects

Introducing new tools to make this easy:

- iOS SSL Kill Switch
- Android SSL Bypass



Outline

iSECpartners part of nccgroup

What is Certificate Pinning ?

- Definition and Background
- Consequences for Mobile Blackbox Testing

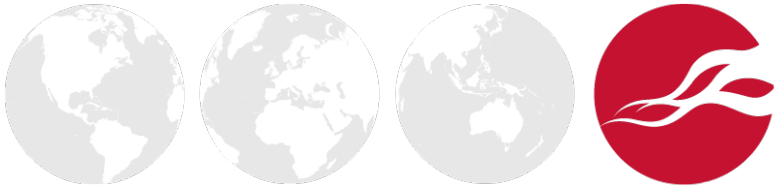
iOS

- Certificate Pinning Within an iOS App
- Intercepting the App's Traffic: MobileSubstrate Extension

Android

- Certificate Pinning Within an iOS App
- Intercepting the App's Traffic: Custom JDWP Debugger

Conclusion



Network Communication on iOS

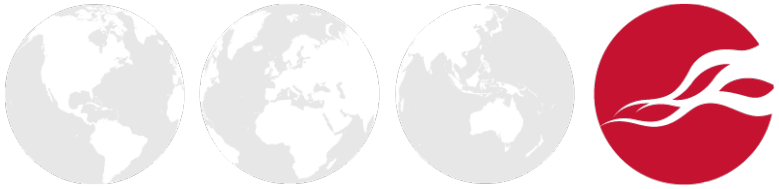
iSECpartners part of nccgroup

Several APIs to do network communication on iOS

- NSStream, CFStream, **NSURLConnection**

Most iOS Apps use NSURLConnection

- High level API to perform the loading of a URL request
- Verifies the server's certificate for *https:* URLs
- Developers can override certificate validation
  - To disable certificate validation (for testing only!)
  - To implement certificate pinning



NSURLConnection

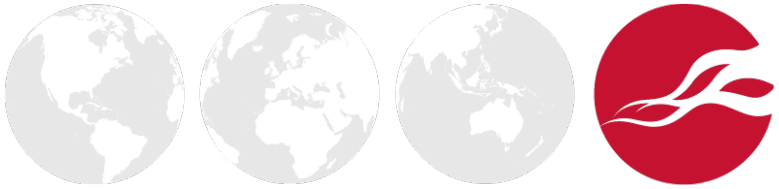
iSECpartners part of nccgroup

NSURLConnection has the following constructor:

- `-(id)initWithRequest:(NSURLRequest *)request delegate:(id <NSURLConnectionDelegate>)delegate`

The delegate has to implement specific methods

- Those methods get called as the connection is progressing
- They define what happens during specific events
  - Connection succeeded, connection failed, etc…
- Two documented ways to do custom certificate validation



NSURLConnectionDelegate

iSECpartners part of nccgroup

Connection Authentication

- `connection:willSendRequestForAuthenticationChallenge:`
- `connection:canAuthenticateAgainstProtectionSpace:`
- `connection:didCancelAuthenticationChallenge:`
- `connection:didReceiveAuthenticationChallenge:`
- `connectionShouldUseCredentialStorage:`

Connection Completion

- `connection:didFailWithError:`

MethodGroup

- `connection:willCacheResponse:` *required method*
- `connection:didReceiveResponse:` *required method*
- `connection:didReceiveData:` *required method*
- `connection:didSendBodyData:totalBytesWritten:totalBytesExpectedToWrite:` *required method*
- `connection:needNewBodyStream`
- `connection:willSendRequest:redirectResponse:` *required method*
- `connectionDidFinishLoading:` *required method*

Custom Certificate Validation

iSECpartners part of nccgroup

Connection Authentication

- `connection:willSendRequestForAuthenticationChallenge:` Strategy 1
- `connection:canAuthenticateAgainstProtectionSpace:`
- `connection:didCancelAuthenticationChallenge:`
- `connection:didReceiveAuthenticationChallenge:`
- `connectionShouldUseCredentialStorage:`

Connection Completion

- `connection:didFailWithError:`

MethodGroup

- `connection:willCacheResponse:` *required method*
- `connection:didReceiveResponse:` *required method*
- `connection:didReceiveData:` *required method*
- `connection:didSendBodyData:totalBytesWritten:totalBytesExpectedToWrite:` *required method*
- `connection:needNewBodyStream`
- `connection:willSendRequest:redirectResponse:` *required method*
- `connectionDidFinishLoading:` *required method*

Custom Certificate Validation

iSECpartners part of nccgroup

Connection Authentication

- `connection:willSendRequestForAuthenticationChallenge:` — Strategy 1
- `connection:canAuthenticateAgainstProtectionSpace:`
- `connection:didCancelAuthenticationChallenge:`
- `connection:didReceiveAuthenticationChallenge:` — Strategy 2 (deprecated)
- `connectionShouldUseCredentialStorage:`

Connection Completion

- `connection:didFailWithError:`

MethodGroup

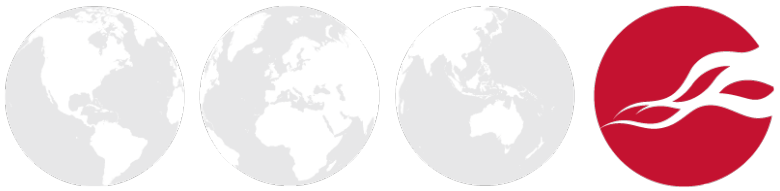
- `connection:willCacheResponse:` *required method*
- `connection:didReceiveResponse:` *required method*
- `connection:didReceiveData:` *required method*
- `connection:didSendBodyData:totalBytesWritten:totalBytesExpectedToWrite:` *required method*
- `connection:needNewBodyStream`
- `connection:willSendRequest:redirectResponse:` *required method*
- `connectionDidFinishLoading:` *required method*

Jailbroken iOS Development

iSECpartners part of nccgroup

MobileSubstrate

- Available on jailbroken devices
- "de facto framework that allows 3rd-party developers to provide run-time patches to system functions"
- MobileSubstrate patches are called "extensions" or "tweaks"

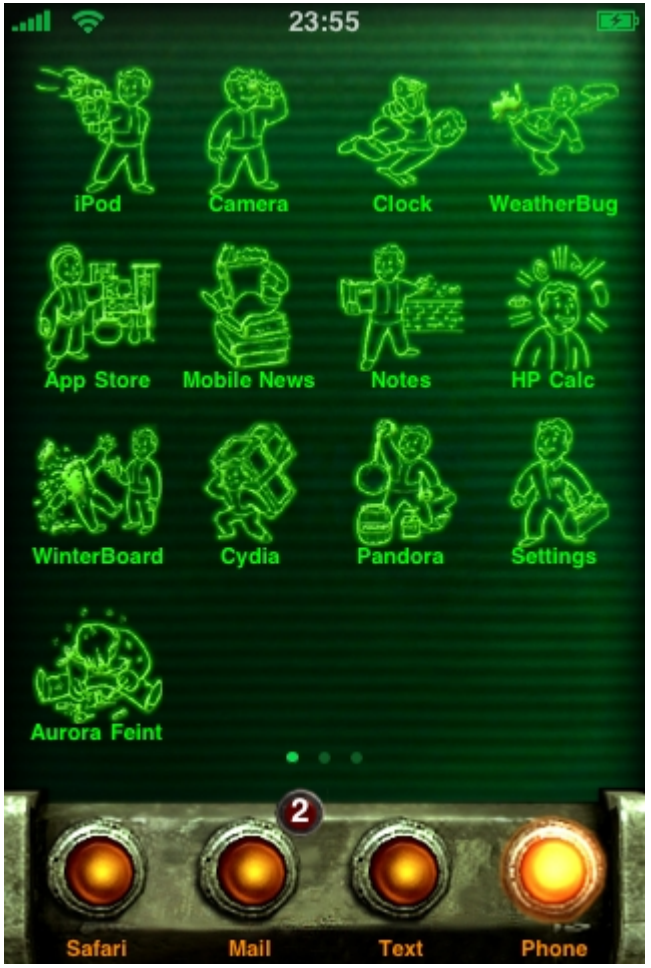


MobileSubstrate Extension

iSECpartners part of nccgroup

One example: WinterBoard

- Hooks into the SpringBoard APIs
- Allows users to customize their home screen



iOS SSL Kill Switch

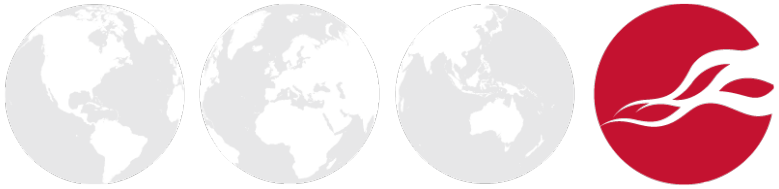
iSECpartners part of nccgroup

MobileSubstrate extension that patches NSURLConnection at runtime

- Automatically loaded with every App on the device

Hooks into the constructor for NSURLConnection:

- `-(id)initWithRequest:(NSURLRequest *)request delegate:(id <NSURLConnectionDelegate>)delegate`
  - Replaces the delegate with a "delegate proxy"
  - The "delegate proxy" forwards method calls to the original delegate
  - **Except** calls to any method that performs custom certificate validation



iOS SSL Kill Switch

iSECpartners part of nccgroup

Hooking NSURLConnection's constructor

```
#import "HookedNSURLConnectionDelegate.h"

%hook NSURLConnection

// Hook into NSURLConnection's constructor
- (id)initWithRequest:(NSURLRequest *)request delegate:(id <NSURLConnectionDelegate>)delegate
{
  // Create a delegate "proxy"
  HookedNSURLConnectionDelegate* delegateProxy;
  delegateProxy = [[HookedNSURLConnectionDelegate alloc] initWithOriginal: delegate];

  return %orig(request, delegateProxy); // Call the "original" constructor
}

%end
```

iOS SSL Kill Switch

iSECpartners part of nccgroup

Forwarding method calls to the original delegate

```
@implementation HookedNSURLConnectionDelegate : NSObject
```

...

```
- (void)connection:(NSURLConnection *)connection didReceiveResponse:(NSURLResponse *)response
{
  // Forward the call to the original delegate
  return [origiDelegate connection:connection didReceiveResponse:response];
}
```

iOS SSL Kill Switch

iSECpartners part of nccgroup

Intercepting calls to certificate validation methods

```
@implementation HookedNSURLConnectionDelegate : NSObject
```

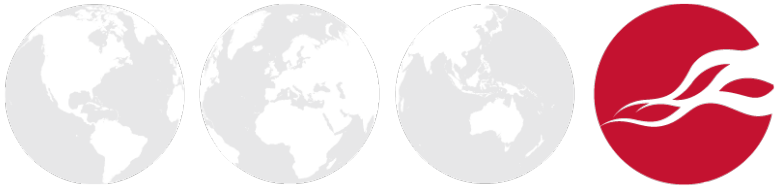
...

```
- (void)connection:(NSURLConnection *)connection
          willSendRequestForAuthenticationChallenge:(NSURLAuthenticationChallenge *)challenge
{
  // Do not forward... Accept all certificates instead
  if([challenge.protectionSpace.authenticationMethod
isEqualToString:NSURLAuthenticationMethodServerTrust])
  {
    NSURLCredential* serverCred;
    serverCred = [NSURLCredential credentialForTrust:challenge.protectionSpace.serverTrust];
    [challenge.sender useCredential:serverCred forAuthenticationChallenge:challenge];
  }
}
```

iOS SSL Kill Switch

iSECpartners
part of nccgroup

DEMO



Outline

iSECpartners part of nccgroup

What is Certificate Pinning ?

- Definition and Background
- Consequences for Mobile Blackbox Testing

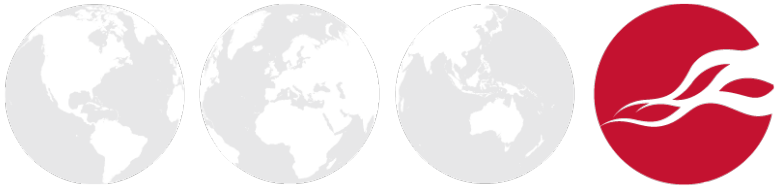
iOS

- Certificate Pinning Within an iOS App
- Intercepting the App's Traffic: MobileSubstrate Extension

Android

- Certificate Pinning Within an iOS App
- Intercepting the App's Traffic: Custom JDWP Debugger

Conclusion

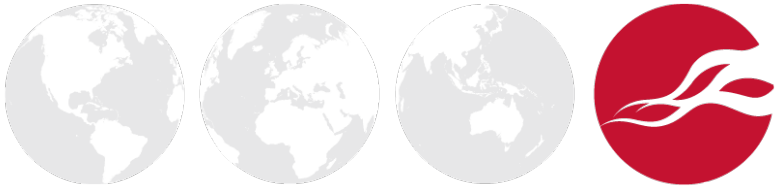


Certificate Validation on Android

iSECpartners part of nccgroup

Certificate Validation and Pinning on Android

- Device trust store cannot be modified by user until Android 4.0 (ICS)
- Certificate pinning can be implemented using an App specific trust store
- Common methods of certificate pinning outlined on Moxie's blog
  - http://blog.thoughtcrime.org/authenticity-is-broken-in-ssl-but-your-app-ha



Certificate Pinning on Android

iSECpartners part of nccgroup

Sample implementation

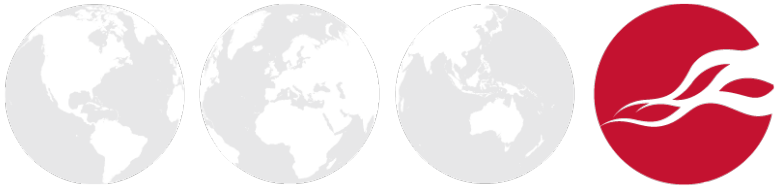
```
private InputStream makeRequest ( Context context, URL url ) {

       // Load custom Trust Store from a file
       AssetManager assetManager = context.getAssets();
      InputStream keyStoreInputStream = assetManager.open("yourapp.store");
      KeyStore trustStore = KeyStore.getInstance("BKS");
      trustStore.load(keyStoreInputStream, "somepass".toCharArray());

      TrustManagerFactory tmf = TrustManagerFactory.getInstance("X509");
      tmf.init(trustStore);

      // Init an sslContext with the custom Trust Store
      SSLContext sslContext = SSLContext.getInstance("TLS");
      sslContext.init(null, tmf.getTrustManagers(), null);

      // Use this sslContext for subsequent HTTPS connections
      HttpsURLConnection urlConnection = (HttpsURLConnection)url.openConnection();
      urlConnection.setSSLSocketFactory(sslContext.getSocketFactory());

      return urlConnection.getInputStream();
}
```

Bypassing Certificate Pinning

iSECpartners part of nccgroup

Many possible ways to implement a bypass

- Decompile/Patch/Recompile/Resign/Sideload
- Custom VM/ROM with hooks built in
- Native code hooking (Mulliner) or native code debugger (gdb, vtrace)
- JDWP debugger

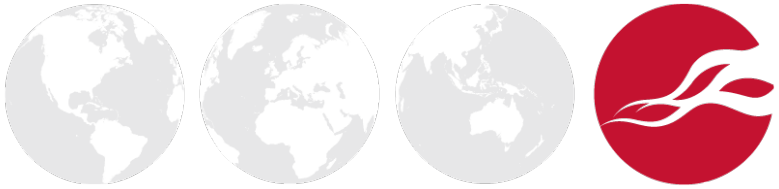


Bypassing Certificate Pinning

iSECpartners part of nccgroup

Many possible ways to implement a bypass

- Decompile/Patch/Recompile/Resign/Sideload
- Custom VM/ROM with hooks built in
- Native code hooking (Mulliner) or native code debugger (gdb, vtrace)
- **JDWP debugger**

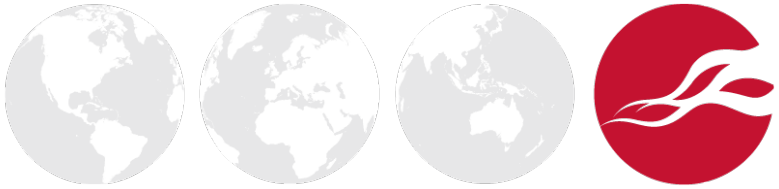


Java Debug Wire Protocol

iSECpartners part of nccgroup

What is the Java Debug Wire Protocol (JDWP) ?

- Standard Java debugging tools
- Programmatic debugging through Java APIs
  - Java Debug Interface (JDI)
- Python bindings available through AndBug

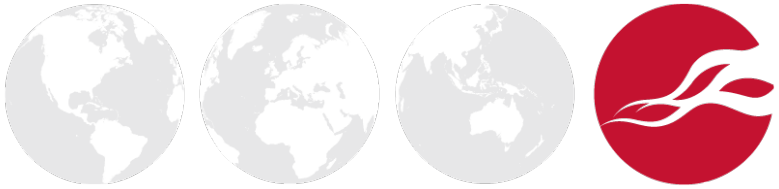


Java Debug Wire Protocol

iSECpartners part of nccgroup

What can we do with a JDWP debugger?

- Normal debugging tasks: set breakpoints, step, etc.
- Once suspended via a JDI event we can:
  - Get the current thread, frame, frame object, local variables and arguments references
  - Load arbitrary classes, instantiate Objects, invoke methods, get and set local variables and arguments values
  - And more…

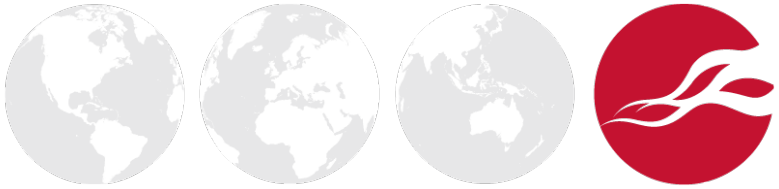


JDWP - Certificate Pinning Bypass

iSECpartners part of nccgroup

Many possible bypass implementations using JDWP debugger

- Three ideas explored:
  1. Custom ClassLoader which loads modified system classes
  2. Use DexMaker to generate proxy classes and replace instances with proxy class instances
  3. Breakpoint on set of SSLSocketFactory and replace it with another SSLSocketFactory configured with an all-trusting TrustManager

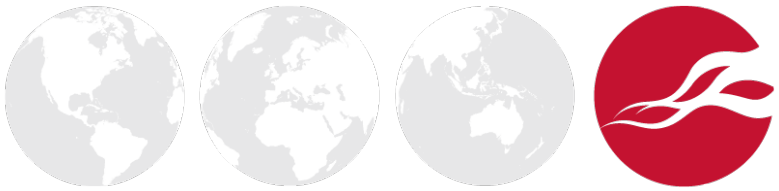


JDWP - Certificate Pinning Bypass I

iSECpartners part of nccgroup

Method 1: Custom ClassLoader to load hooked system classes

- Break on anything
- Set APK ClassLoader to a custom **ClassLoader**
- Custom **ClassLoader.loadClass()** modifies **HttpsURLConnection** upon loading
- Continue execution (may need to restart Activity)

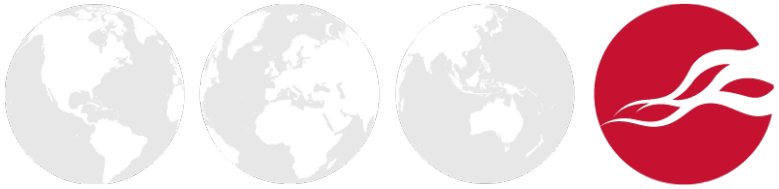


JDWP - Certificate Pinning Bypass I

iSECpartners part of nccgroup

Problems with method 1

- Might work in Java but not in Dalvik
- Class loading is different, **defineClass()** simply not supported
- User defined class loaders are **not** currently allowed to modify classes loaded from the $BOOTCLASSPATH
- Detailed in code comments in **/dalvik/vm/oo/Class.cpp**

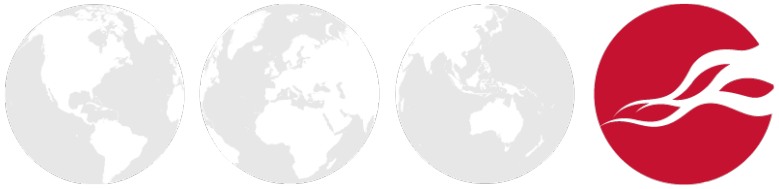


JDWP - Certificate Pinning Bypass II

iSECpartners part of nccgroup

Method 2: Object substitution with proxied copy

- Break after instantiation of **HttpsURLConnection**
- Use **DexMaker** to dynamically generate proxy class for **HttpsURLConnection**
- Replace **HttpsURLConnection** object with one created using the generated proxy class
- Continue execution

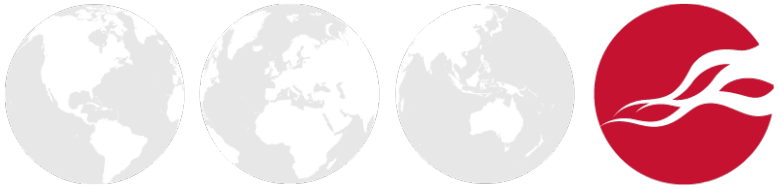


JDWP - Certificate Pinning Bypass III

iSECpartners part of nccgroup

Method 3: TrustManager substitution

- Break on **HttpsURLConnection.setSocketFactory()**
- Replace socket factory local variable with one created using an all-trusting **TrustManager**
- Continue execution

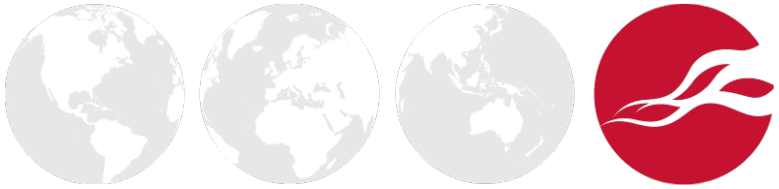


Android SSL Bypass

iSECpartners part of nccgroup

Simple implementation for first version

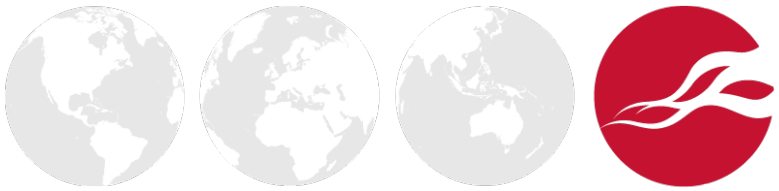
- Using Method 3 right now
- Future versions will explore method 2 for further extensibility



Android SSL Bypass

iSECpartners
part of nccgroup

DEMO



Outline

iSECpartners part of nccgroup

What is Certificate Pinning ?

- Definition and Background
- Consequences for Mobile Blackbox Testing

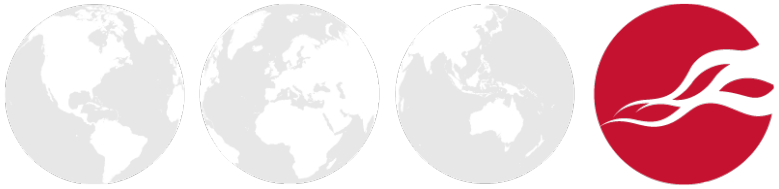
iOS

- Certificate Pinning Within an iOS App
- Intercepting the App's Traffic: MobileSubstrate Extension

Android

- Certificate Pinning Within an iOS App
- Intercepting the App's Traffic: Custom JDWP Debugger

Conclusion



Summary

iSECpartners part of nccgroup

iOS SSL Kill Switch

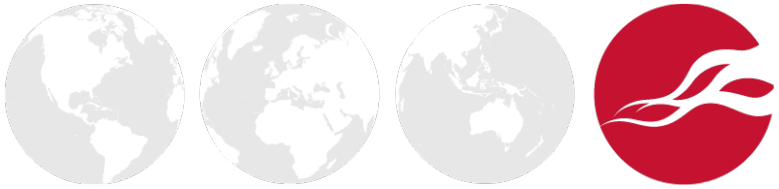
- Tested on iOS 4.3 and iOS 5.1
- https://github.com/iSECPartners/ios-ssl-kill-switch

Android SSL Bypass Tool

- Tested on Android 2.3.3 and 4.0.3
- https://github.com/iSECPartners/android-ssl-bypass

Comments / Ideas ?

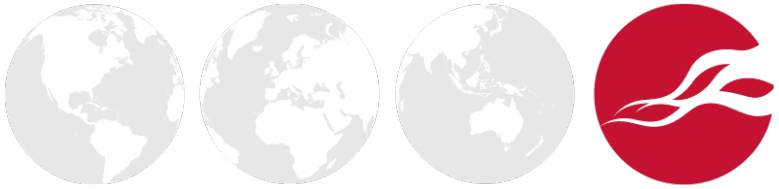
- justine@isecpartners.com
- alban@isecpartners.com



The End

iSECpartners
part of nccgroup

QUESTIONS ?



Reference Material

iSECpartners part of nccgroup

Certificate pinning on iOS

- http://blog.securemacprogramming.com/2011/12/on-ssl-pinning-for-cocoa-touch/

MobileSubstrate

- http://iphonedevwiki.net/index.php/MobileSubstrate

Certificate pinning on Android

- http://blog.thoughtcrime.org/authenticity-is-broken-in-ssl-but-your-app-ha

DEXMaker

- https://code.google.com/p/dexmaker

iSEC Partners on GitHub

- https://github.com/iSECPartners