2016.03.23 JSSECセキュアコーディングデイ

iOSアプリの
セキュアコーディング入門

福本 郁哉

じこしょうかい

福本 郁哉 @株式会社SRA

- ECサイト、業務系システム開発（2009〜）
- iOSアプリ開発（2011 頃〜）
  - 企業内向け業務アプリ
- セキュリティ（2012 頃〜）
  - Webアプリ脆弱性診断
  - モバイルアプリ(Android, iOS)の脆弱性診断
    - AppleにCore Foundationの脆弱性報告: CVE-2015-1092
  - JSSEC『Androidアプリのセキュア設計・セキュアコーディングガイド』の執筆に参加

iOSの安全神話

「iOSは安全」

というイメージ

「iOSは安全」というイメージ形成

- Androidが悪目立ちしていた（している）？
- 特にモバイル流行の初期、「Androidは危ない」というニュースが目を引いていた
- Androidのシェアはダントツに多い: 83%

  2015 Q2, by IDC http://www.idc.com/prodserv/smartphone-os-market-share.jsp

  ➢ 攻撃のターゲットになることも多い

「Appleが審査してるから安全なんでしょ？」

App Storeの審査は脆弱性診断ではない

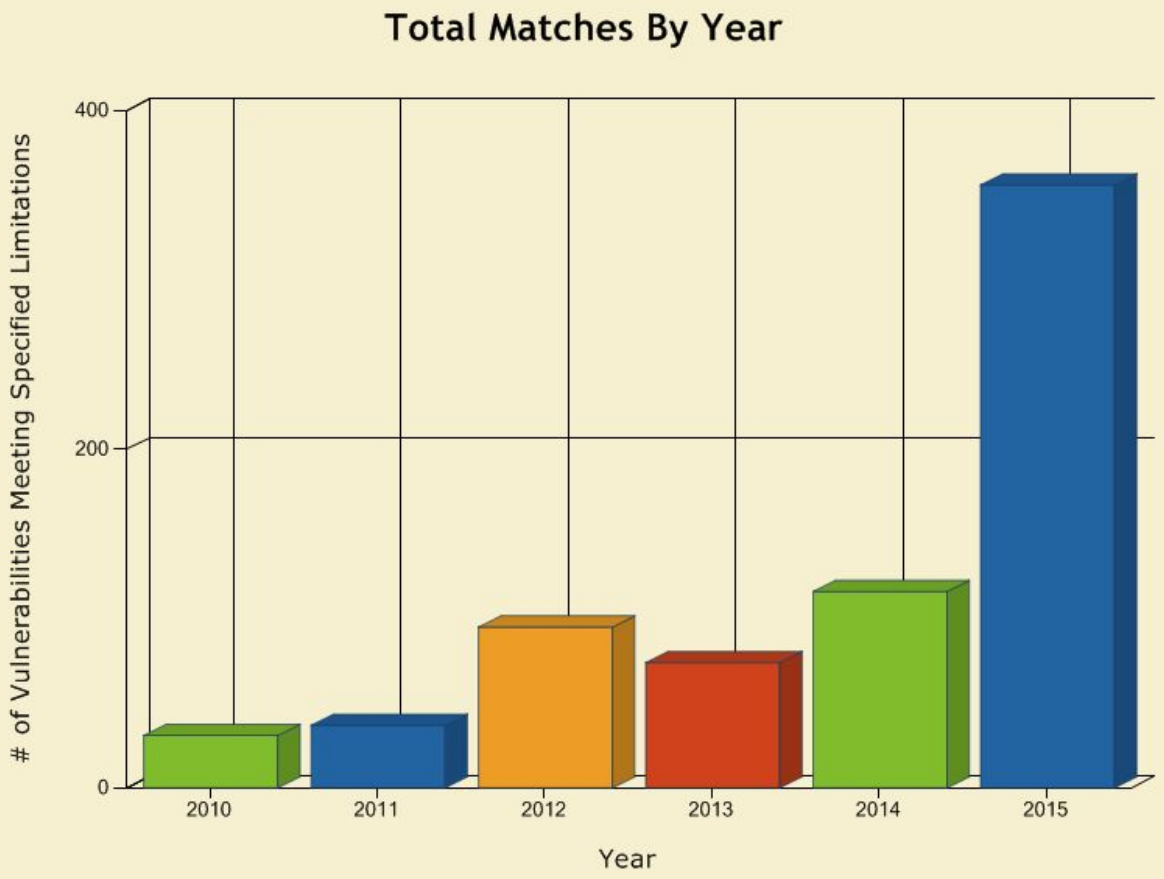
App Storeの審査はアプリの脆弱性診断ではない

- 迷惑アプリ・ウイルスは積極的に排除
  - e.g. ユーザーの個人情報を過剰に要求
  - 既知のウイルス
- App Store Review Guidelines https://developer.apple.com/app-store/review/guidelines/
  - 主なリジェクト理由
    - クラッシュ、バグ
    - ユーザビリティ
  - 脆弱性についての記載はない

iOSの脆弱性報告数は増加傾向

Search Parameters:

- **Contains Software Flaws (CVE)**
- **Keyword (text search):** apple ios
- **Publication Start Date:** January 2010
- **Publication End Date:** December 2015
- **CVSS Version:** 3



NVD - Statistics Results

iOSの脆弱性報告数は増加傾向

Top 50 Products By Total Number Of "Distinct" Vulnerabilities in 2015

Go to year: 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 All Time Leaders

	Product Name	Vendor Name	Product Type	Number of Vulnerabilities
1	Mac Os X	Apple	OS	384
2	Iphone Os	Apple	OS	375
3	Flash Player	Adobe	Application	313
4	Air Sdk & Compiler	Adobe	Application	
5	Air Sdk	Adobe	Application	
6	AIR	Adobe	Application	
7	Internet Explorer	Microsoft	Application	
8	Chrome	Google	Application	
9	Firefox	Mozilla	Application	
10	Windows Server 2012	Microsoft	OS	
11	Ubuntu Linux	Canonical	OS	
12	Windows 8.1	Microsoft	OS	
13	Windows Server 2008	Microsoft	OS	
14	Windows 7	Microsoft	OS	
15	Windows 8	Microsoft	OS	
16	Windows Rt 8.1	Microsoft	OS	
17	Windows Rt	Microsoft	OS	
18	Safari	Apple	Application	
19	Windows Vista	Microsoft	OS	
20	Android	Google	OS	130
21	Acrobat	Adobe	Application	130

https://www.cvedetails.com/top-50-products.php?year=2015

VB NEWS EVENTS RESEARCH

Software with the most vulnerabilities in 2015: Mac OS X, iOS, and Flash

EMIL PROTALINSKI DECEMBER 31, 2015 8:23 AM

TAGS: ADOBE, ANDROID, APPLE, CVE, FLASH, OS X, VULNERABILITIES, WINDOWS

http://venturebeat.com/2015/12/31/software-with-the-most-vulnerabilities-in-2015-mac-os-x-ios-and-flash/

App Storeの審査はアプリの脆弱性診断ではない

- 過去の騒ぎ
  - Charlie Miller 事件（2011）
    - わざと脆弱性持つアプリを作って提出したら審査通っちゃった
    - ブログで公開→アプリ削除
    - 開発者アカウント停止
  - XcodeGhost（2015）
    - 魔改造Xcode
    - フレームワークの一部が悪意あるコードに置き換えられていた
    - これで作ったアプリにはバックドアが仕込まれることになった

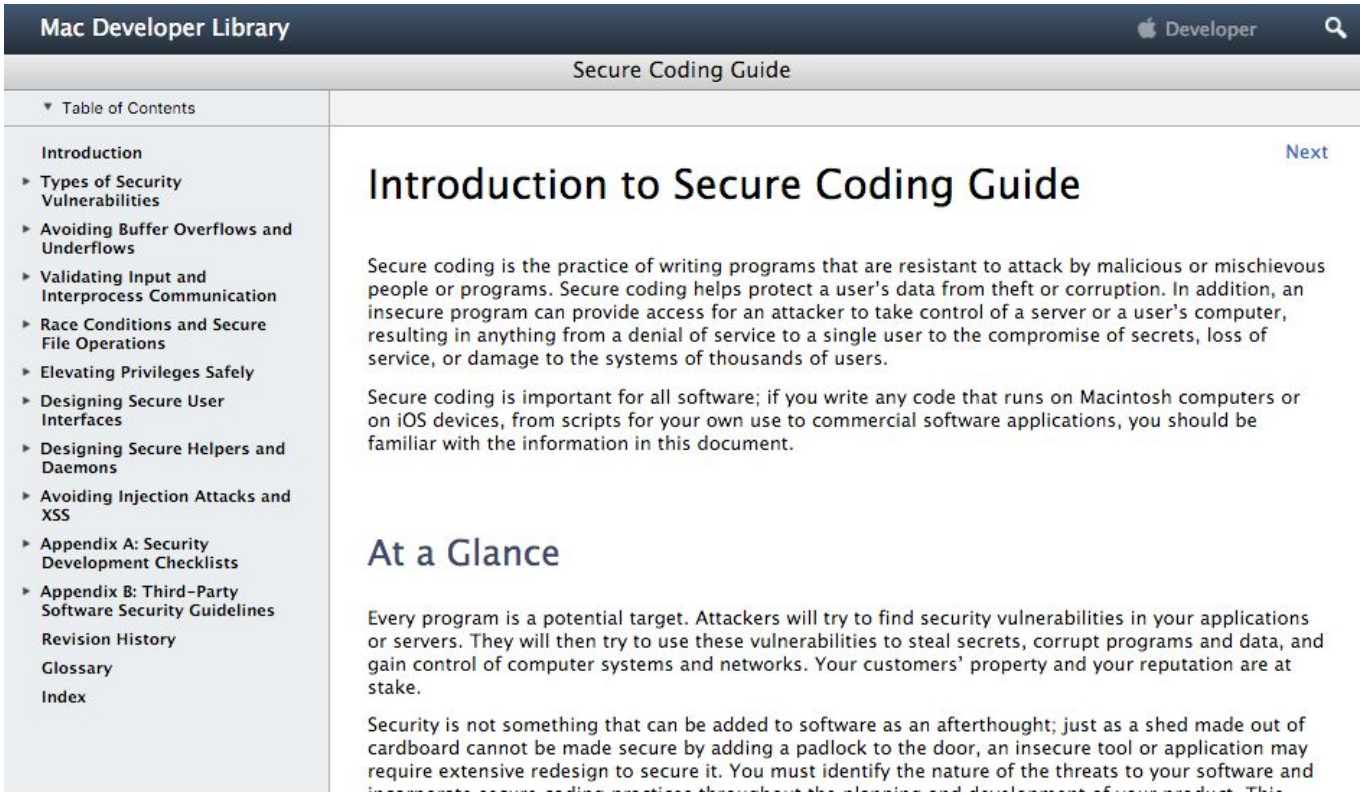
「iOSは安全」
というイメージは捨てたほうがいい

アプリの安全性は
開発者の責任

安全性を意識した開発
が必要

Appleのセキュアコーディングガイド

- AppleのSecure Coding Guide https://developer.apple.com/library/mac/documentation/Security/Conceptual/SecureCodingGuide/Introduction.html



iOSのAPIには落とし穴がたくさんある

- iOSアプリの脆弱性の多くは、
  - **APIの適切でない選択**
  - **APIの危険な使い方**

  に原因がある
- App Storeの審査は、オフィシャルなAPIを使った結果としての脆弱性にはわりと無頓着
- **APIの特性を理解**したうえで、安全を意識した開発（設計・実装）が不可欠

iOSアプリの
セキュア
コーディング

本日のトピック

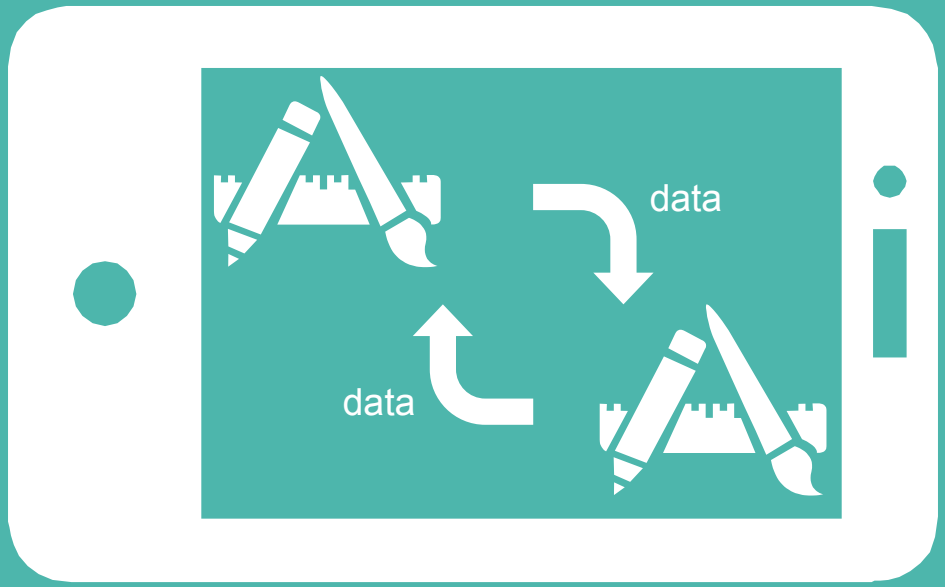
iOSアプリ開発でよく使われる機能から以下をピックアップして、
API使用時の注意点やセキュア実装の具体例をご紹介します。

- アプリ間連携
  - Custom URL Scheme
  - Pasteboard
- HTTPS通信時のサーバ証明書検証
  - NSURLConnection
  - NSURLSession
  - UIWebView / WKWebView

お断り:
Swiftのコードは載せてません

フレームワークのAPIは共通ですから...

アプリ間連携



iOSでのアプリ間連携

- **Custom URL Scheme**
- **Pasteboard**
- App Extensions
- App Group
- iTunes File Sharing

Custom URL Scheme

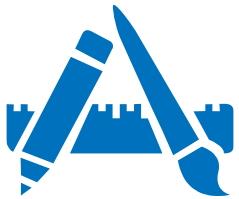
Custom URL Scheme

com.partner.app

```
NSURL *myAppURL = [NSURL URLWithString:@" myapp://foo?data1=hoge&data2=fuga"];
[[UIApplication sharedApplication]  openURL:myAppURL]
```

OSがアプリを起動

com.my.app

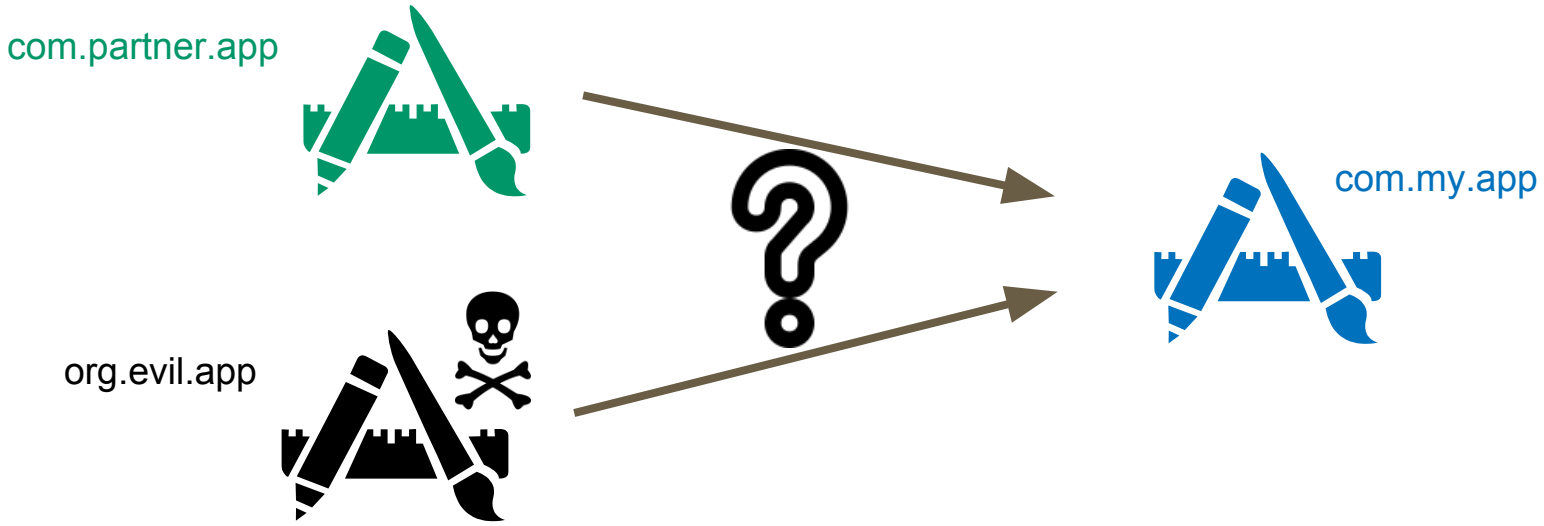


myapp://
に反応するよう、インストール時に OSに登録

```
- (BOOL)application:(UIApplication *)application  openURL:(NSURL *)url
sourceApplication:(NSString *)sourceApplication  annotation:(id)annotation {
    NSString *theQuery = [url query];
    // クエリパラメータ（data1=hoge&data2=fuga）を受け取った
}
```

Custom URL Scheme
注意点その1

- どんなアプリから起動されるか分からない
  - **自分を呼び出したアプリのチェック**
  - **入力値（URLクエリ）のチェック、無害化**



Custom URL Scheme
呼び出し元アプリのチェック

```
- (BOOL)application:(UIApplication *)application openURL:(nonnull NSURL *)url
sourceApplication:(nullable NSString *)sourceApplication annotation:(nonnull id)
annotation {
    // 呼び出し元アプリをチェックする
    if ([sourceApplication isEqualToString:@"com.partner.app"]) {
        NSString *queryStr = [[url query]
stringByReplacingPercentEscapesUsingEncoding:NSUTF8StringEncoding];
        // クエリを取り出して何かする
        ---snip---
        return YES;
    }
    return NO;
}

// application:openURL:sourceApplication:annotation: はiOS 9.0で非推奨になった
// iOS 9.0 以降ではapplication:openURL:options:を使う

- (BOOL)application:(UIApplication *)application openURL:(nonnull NSURL *)url
options:(nonnull NSDictionary<NSString *,id> *)options {
    // 呼び出し元アプリをチェックする
    if ([options[UIApplicationOpenURLOptionsSourceApplicationKey] isEqualToString:
@"com.partner.app"]) {
        ---snip---
    }
}
```

Custom URL Scheme
呼び出し元アプリのチェック

- application:handleOpenURL: は使わない
  - 呼び出し元アプリを知ることができない

```
- (BOOL)application:(UIApplication *)application handleOpenURL:(NSURL *)url {
    // 呼び出し元アプリをチェックすることができない...

}
```

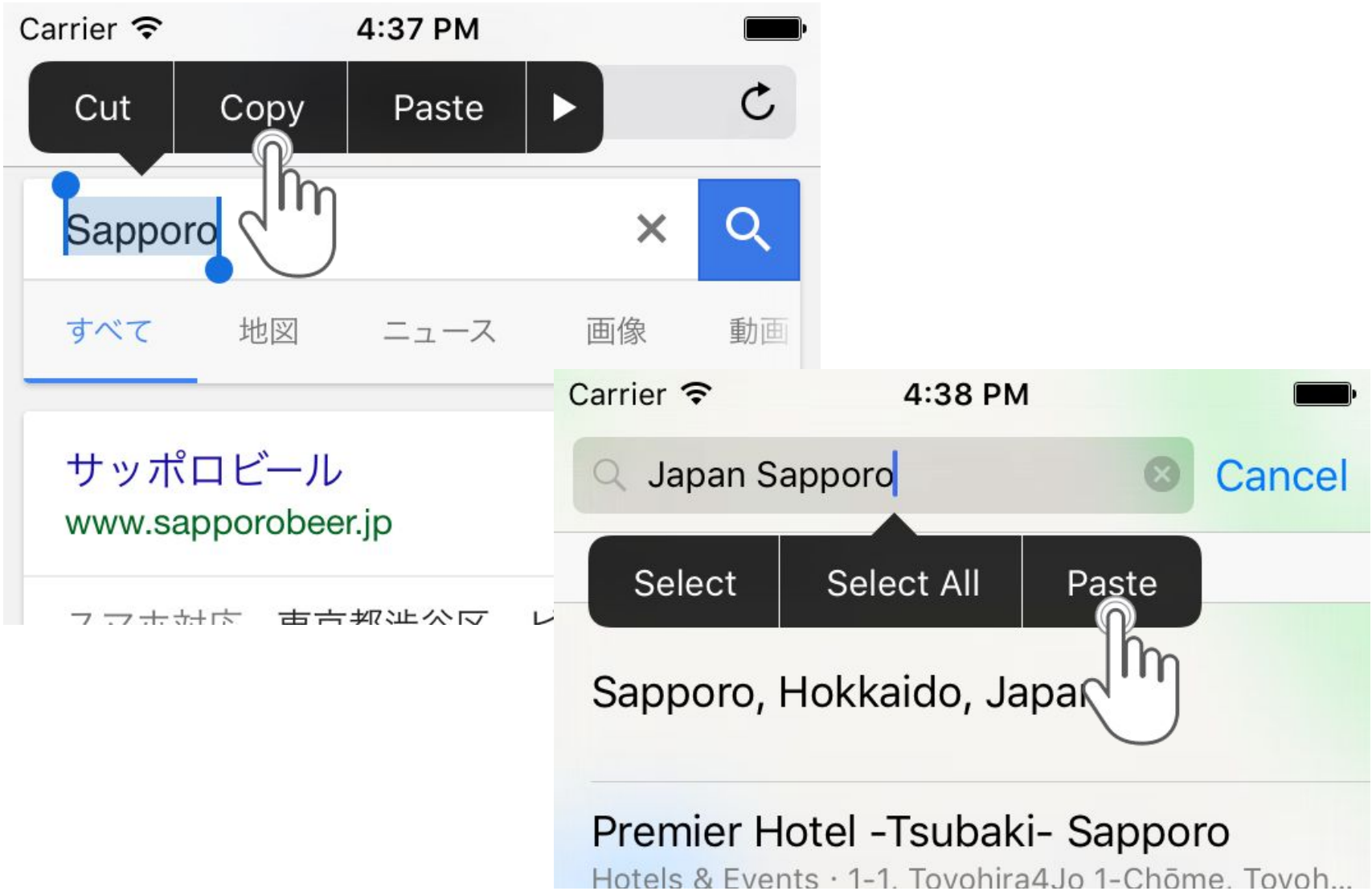
Custom URL Scheme
注意点その2

- URL Scheme Hijackingの可能性
  - 複数の同一名スキームがOSに登録された場合の挙動が定義されていない
    →どのアプリが起動されるのか分からない
  - 自分が受け取るべきデータが他のアプリに横取りされるかもしれない
  - 想定外のアプリにデータを渡してしまうかもしれない

センシティブなデータはURL Schemeで送受信しない

Pasteboard

Pasteboard


Carrier
4:37 PM
Cut
Copy
Paste
Sapporo
すべて
地図
ニュース
画像
動画
サッポロビール
www.sapporobeer.jp
Carrier
4:38 PM
Japan Sapporo
Cancel
Select
Select All
Paste
Sapporo, Hokkaido, Japan
Premier Hotel -Tsubaki- Sapporo
Hotels & Events · 1-1, Toyohira4Jo 1-Chōme, Toyoh...

Pasteboard
System Pastebord と App Pasteboard

- Pasteboardには2種類ある
  - **System Pasteboard**
    - 端末内の**全てのアプリで共有**される
    - シングルトン
    - テキストフィールドの「コピー」「切り取り」「貼り付け」で使われるやつ
  - **App Pasteboard** (Custom-Named Pasteboard)
    - **同じ開発者により署名されたアプリ間でのみ共有**される

- **自作のアプリ間でデータ共有するときにはApp Pasteboardを使う**
  - System Pasteboardは使用しない
  - iOS 7.0 未満では、Pasteboardの名前さえ知ればどのアプリからでもアクセスできてしまうことに注意
  - 自作ではない特定のアプリとデータ共有する場合は、Pasteboard自体使わない

Pasteboard
「コピー」「切り取り」の抑制

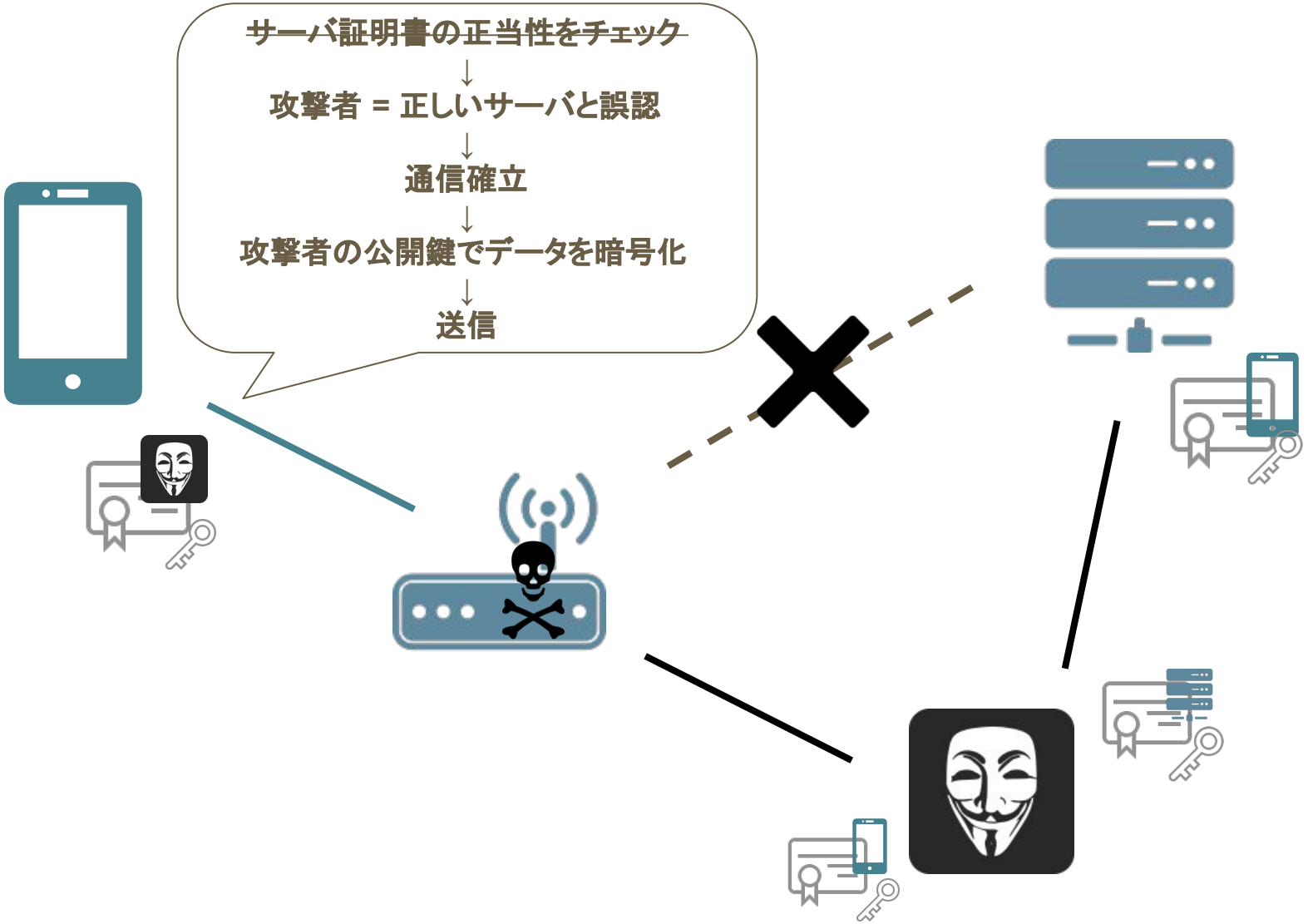
- TextFieldやWebViewの長押しで[コピー/切り取り]されたデータは system pasteboardに保存され、全アプリに共有されてしまう
- ユーザーに[コピー/切り取り]されたくない箇所では、コンテキストメニューから項目を消してしまおう

```
#import "RestrictedTextField.h"
// RestrictedTextFieldはUITextFieldのサブクラス

@implementation RestrictedTextField

- (BOOL)canPerformAction:(SEL)action withSender:(id)sender {
    if (action == @selector(copy:) || action == @selector(cut:)) {
        return NO;
    } else {
        return YES;
    }
}

@end
```

HTTPS通信時のサーバ証明書検証

中間者攻撃 (Man-in-the-middle attack)

中間者攻撃 (Man-in-the-middle attack)



iOS SDKで用意されている代表的な通信系API

- NSURLConnection
- NSURLSession
- WebView (WKWebView, UIWebView)

それぞれに、
サーバ証明書検証をバイパスさせるAPI
が用意されている

HTTPS通信時のサーバ証明書検証をバイパスさせるコード NSURLConnection

```
- (void)connection:(NSURLConnection *)connection
willSendRequestForAuthenticationChallenge:(NSURLAuthenticationChallenge *)challenge
{
    if ([challenge.protectionSpace.authenticationMethod isEqualToString:
NSURLAuthenticationMethodServerTrust]) {
        [challenge.sender useCredential:[NSURLCredential credentialForTrust:
challenge.protectionSpace.serverTrust] forAuthenticationChallenge:challenge];
    }
    [challenge.sender continueWithoutCredentialForAuthenticationChallenge:
challenge];
}
```

HTTPS通信時のサーバ証明書検証をバイパスさせるコード
NSURLConnection

```
// NSURLRequestのプライベートメソッド

+ (BOOL)allowsAnyHTTPSCertificateForHost:(NSString *)host {
    return YES;
}
```

※ 非公開APIなので、App Storeの審査でリジェクトされる可能性大

HTTPS通信時のサーバ証明書検証をバイパスさせるコード NSURLSession

```
- (void)URLSession:(NSURLSession *)session
    didReceiveChallenge:(NSURLAuthenticationChallenge *)challenge
      completionHandler:(void (^)(NSURLSessionAuthChallengeDisposition,
NSURLCredential *))completionHandler {
    if ([challenge.protectionSpace.authenticationMethod isEqualToString:
NSURLAuthenticationMethodServerTrust]) {
        SecTrustRef secTrustRef = challenge.protectionSpace.serverTrust;
        NSURLCredential *credential = [NSURLCredential credentialForTrust:
secTrustRef];
        // NSURLCredentialのオブジェクトを返すと許可扱いになる
        completionHandler(NSURLSessionAuthChallengeUseCredential, credential);
    }
}
```

HTTPS通信時のサーバ証明書検証をバイパスさせるコード
UIWebView, WKWebView

- NSURLConnection / NSURLSessionのサーバー証明書検証バイパスコードはUIWebViewの通信にも影響する
- WKWebViewでは、WKNavigationDelegateでNSURLSessionと同様のバイパスコードを書くことができる

```
- (void)webView:(WKWebView *)webView didReceiveAuthenticationChallenge:
(nonnull NSURLAuthenticationChallenge *)challengecompletionHandler:(nonnull
void (^)(NSURLSessionAuthChallengeDisposition, NSURLCredential * _Nullable))
completionHandler {
    SecTrustRef trust = challenge.protectionSpace.serverTrust;
    NSURLCredential *credential = [NSURLCredential credentialForTrust:trust];
    // NSURLCredentialのオブジェクトを返すと許可扱いになる
    completionHandler(NSURLSessionAuthChallengeUseCredential, credential);
}
```

なぜバイパスコードを書いてしまうか

- 開発中のデバッグ、テストのために自己署名証明書を使うことが多い
  - 一時的な回避のつもりで実装
  - そのままリリースされる

→ **検証用のコードがリリース版に入らないように**細心の注意を払う
  - #ifdef DEBUG

→ 最近は安い証明書もたくさんあるので**買ってもいいのでは**、という気もする
  - 無料の証明書もけっこうある

しかし、こんなケースも…
サードパーティ製広告モジュールの脆弱性

公開日：2015/10/14　最終更新日：2015/10/22

JVN#48211537
iOS 版 Party Track SDK におけるサーバ証明書の検証不備の脆弱性

概要
株式会社アドウェイズが提供する iOS 版 Party Track SDK には、サーバ証明書の検証不備の脆弱性が存在します。

影響を受けるシステム
- iOS 版 Party Track SDK 1.6.6 より前のバージョンを組み込んだアプリケーション

詳細情報
株式会社アドウェイズが提供する iOS 版 Party Track SDK が組み込まれた iOS アプリには、HTTPS を利用した暗号通信において、サーバ証明書を適切に検証しない脆弱性が存在します。開発者によると、SDK が行う通信以外にも、NSURLConnection を使って行うアプリ内の通信全般がサーバ証明書を検証しない状態になるとのことです。

想定される影響
通信経路上の第三者に暗号通信の内容を盗聴される、あるいは通信内容を改ざんされるなど、中間者攻撃 (man-in-the-middle attack) の被害を受ける可能性があります。

対策方法
SDK をアップデートし、アプリケーションをリビルドする
iOS 版 Party Track SDK を組み込んだアプリケーションの開発者は、株式会社アドウェイズが提供する情報をもとに対策済みの Party Track SDK にアップデートし、アプリケーションをリビルドしてください。

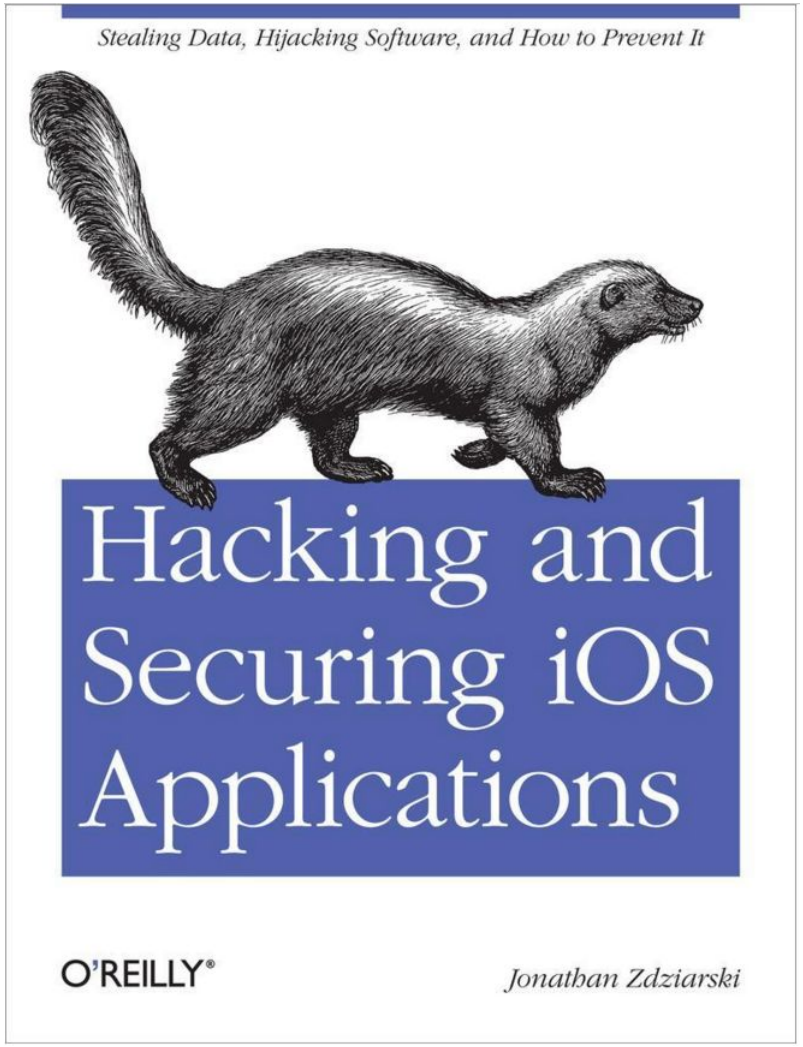
JVN
HOME
JVNとは
脆弱性レポートの読み方
脆弱性レポート一覧
VN-JP
VN-JP (連絡不能)
VN-VU
TA
TRnotes
JVN iPedia 脆弱性対策情報データベース
MyJVN
JVNJS/RSS
ベンダ情報一覧
連絡不能開発者一覧
脆弱性情報の届出
お問合せ先

https://jvn.jp/jp/JVN48211537/

- 広告用のモジュールに証明書検証バイパス系のコードが入っていた
- バイパス系コードの影響は広範囲に及ぶ
- アプリ自身が実装していなくても、組み込んだサードパーティ製のモジュールに脆弱性があれば影響を受けてしまう
- いかんともしがたい
- **アプリの最終確認はリリースビルドで！**

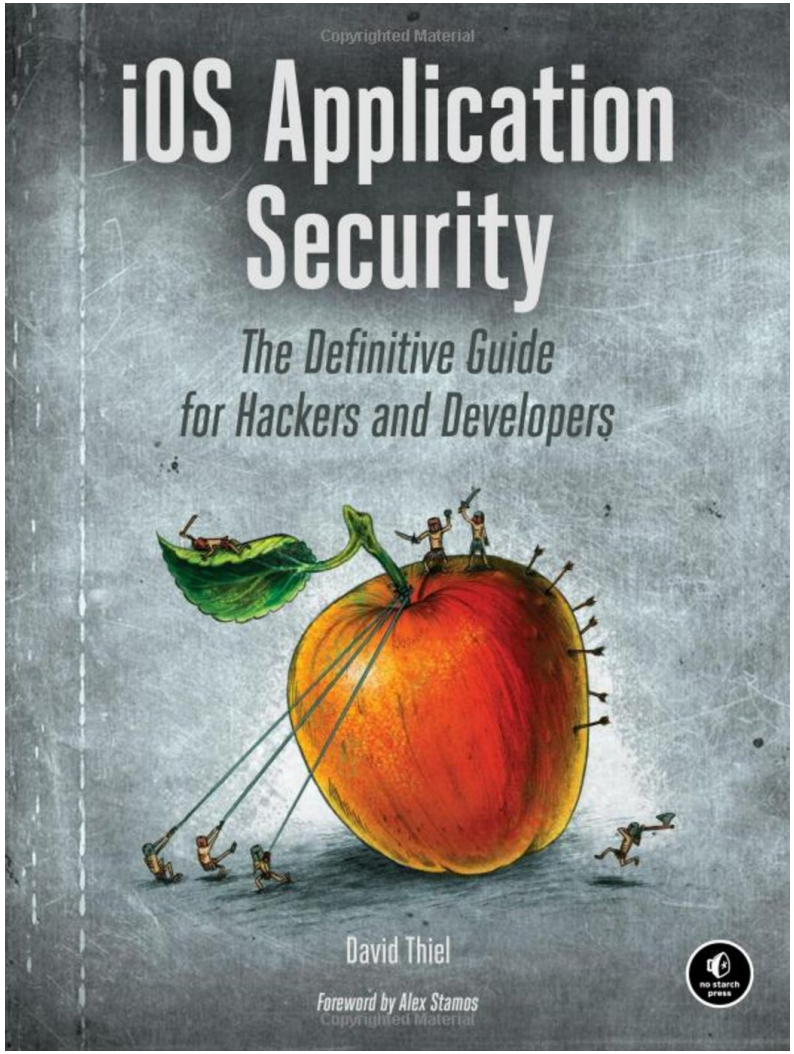
iOSセキュリティ関連書籍の紹介

iOSセキュリティ関連書籍 - その1



- OSとアプリの構造が詳説
  - iOSのセキュリティモデル
  - アプリのIPAアーカイブ
  - アプリのバイナリ構造
- 攻撃者視点と防御視点の双方でセキュア開発の方法を解説
  - ただし、一般アプリの開発者としては、ちょっとマニアック過ぎな感ある
  - 低レイヤーの攻撃・防御がメイン
    - カウンター・フォレンジック
    - Jailbreak検知
    - ランタイムの保護
  - ペネトレーションテストには有用
- やや古い
  - 2012年
  - iOS 5.x までに対応

iOSセキュリティ関連書籍 - その2



- **最新**のiOSセキュリティ本（たぶん）
  - 2016年2月
  - iOS 9 対応
- 開発者視点で書かれている
  - APIの落とし穴と安全な使い方を詳説
  - 危険な例だけじゃなく、安全な**実装例**が豊富
- 読みやすい
  - 平易な英文で簡潔に書かれている
  - 適度な分量(259ページ)

おわり