セキュアコーディングガイド 第9版(Android 8.0対応)

セキュアコーディングWG リーダー

安藤　彰

<Akira.Ando@sony.com>

JAPAN SMARTPHONE SECURITY ASSOCIATION

アジェンダ

- セキュアコーディングガイドについて
- 〃　　第9版の改定内容
- ガイド HTML版のご紹介

JAPAN SMARTPHONE SECURITY ASSOCIATION

セキュアコーディング ガイドについて

２０１２年６月　初版公開

Androidアプリセキュリティのノウハウ集

通称：JSSECセキュアコーディングガイド
PDF文書とセキュアなサンプルコード一式（無償）
http://www.jssec.org/report/securecoding.html
「Android セキュアコーディング」と検索

デファクトスタンダードなガイド・基準

通信キャリアや多くのアプリベンダーでも活用。
受入基準にするアプリ発注会社もある。

ガイドの歴史

- 年1回から2回のペースで改訂

- 2014年
  - 英語版リリース

セキュアコーディングガイド

第９版

JAPAN SMARTPHONE SECURITY ASSOCIATION

第9版の特徴

2018年2月1日版　改定内容

主な改定内容はAndroid 8.0 (Oreo)への対応

- Autofillフレームワーク
- Permission
- Account Manager
- SSLv3非サポートについて
- Android ID
- WebView

Autofillフレームワーク

Autofillフレームワーク

- Autofillフレームワークとは？

  ・ユーザーが入力した情報を「保存」
  ・再度入力時に保存しておいた情報を「自動入力」

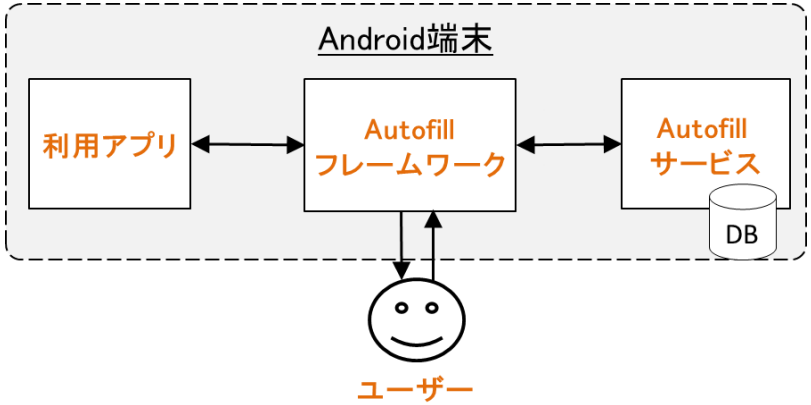
  を実現するための枠組みを提供する

- 扱うデータ

  ユーザー名、パスワード、住所、電話番号、
  クレジットカード情報 etc…

Autofillフレームワーク

- 登場人物
  - 利用アプリ：自動入力の対象アプリ
  - Autofillサービス：自動入力機能を実装するサービス
  - ユーザー： Autofillサービスの選択や保存の許可を行う
  - Autofillフレームワーク：3者間を繋ぐ役目



Autofillフレームワーク

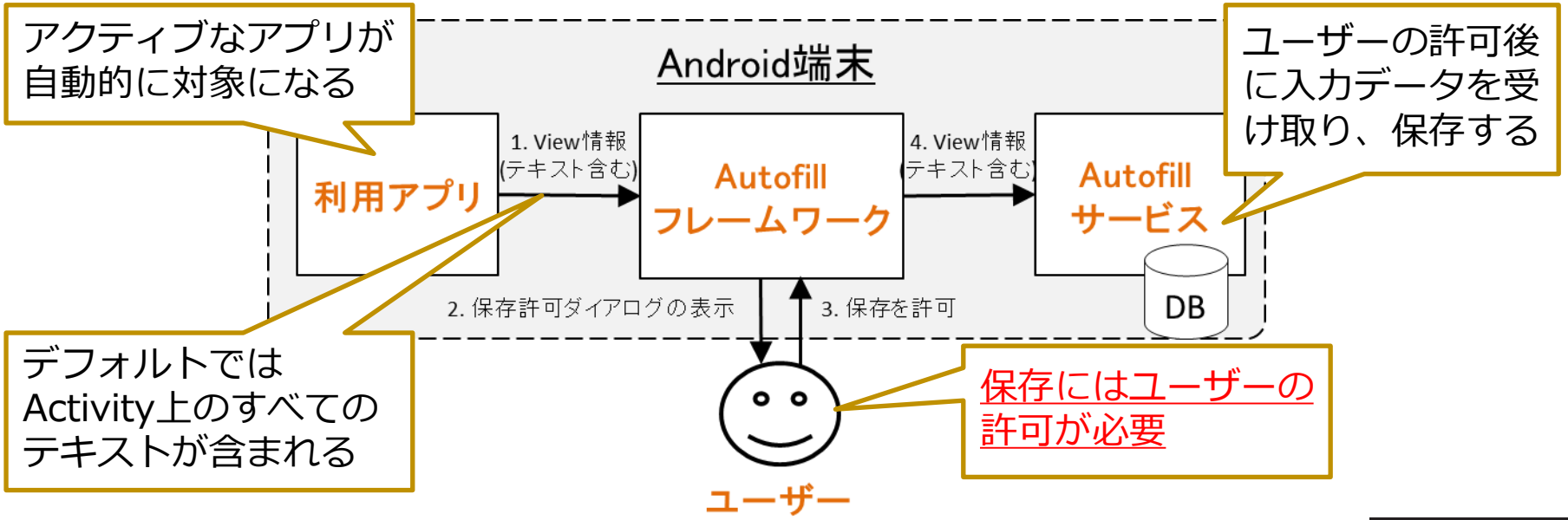
- ユーザーによるAutofillサービスの選択

「このアプリが信頼できることを確認してください」

とシステムが固定で表示する

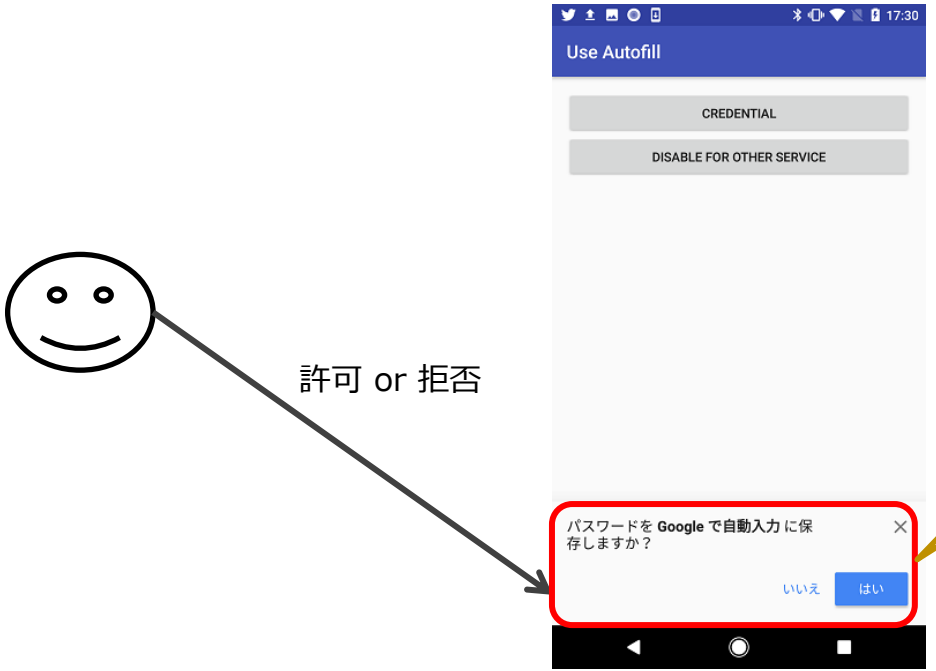
Autofillフレームワーク

- 概要(保存)



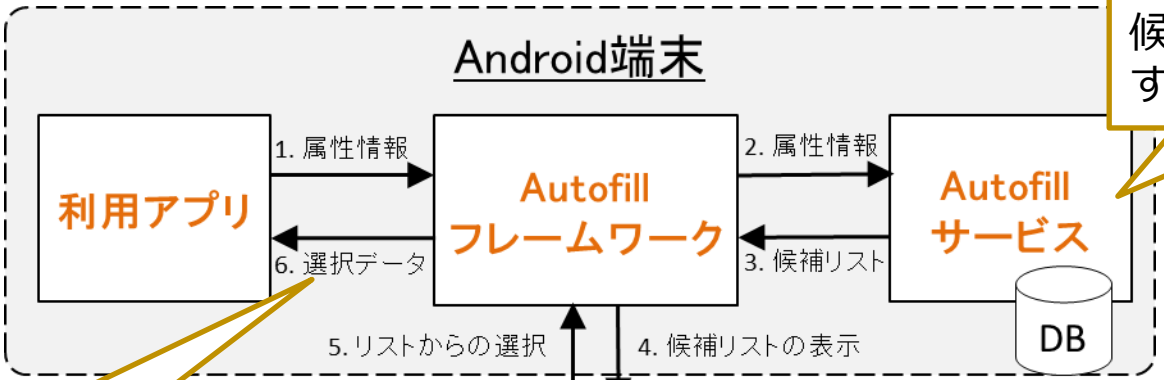
Autofillフレームワーク

- 概要(保存)



Autofillフレームワーク

- 概要(自動入力)



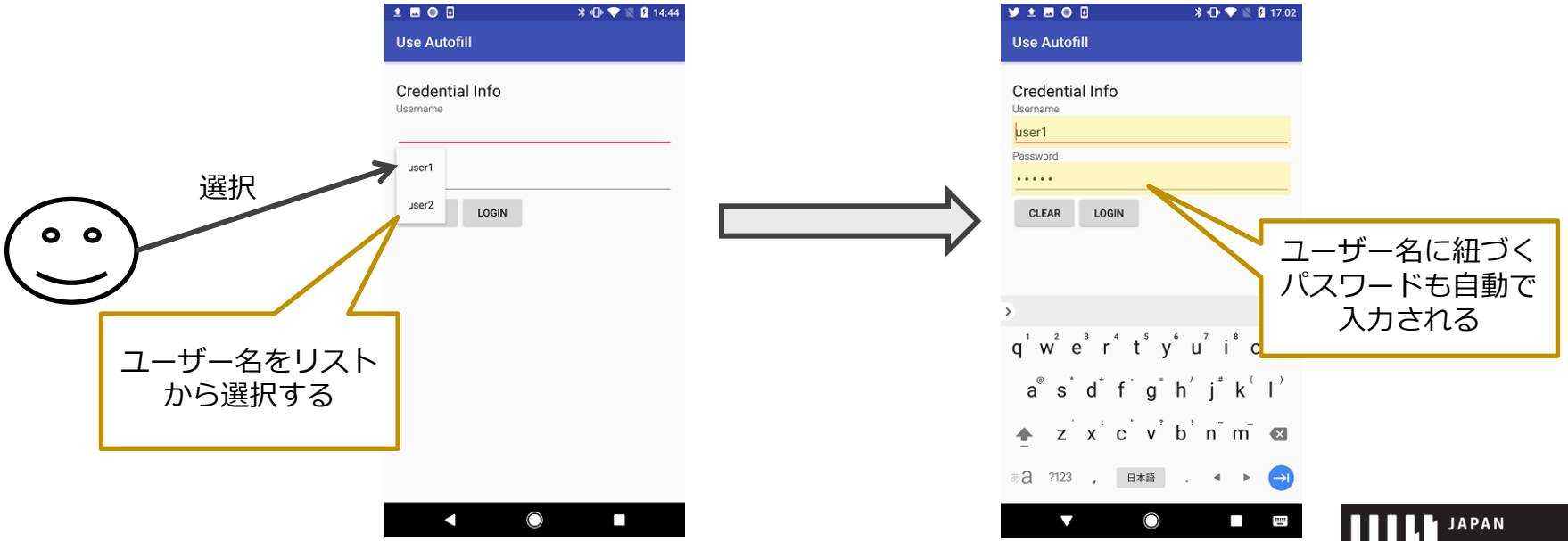
属性情報を元に、候補リストを作成する

ユーザーが選択したデータが自動入力される

入力するデータは、必ずユーザーが選択する

Autofillフレームワーク

- 概要(自動入力)



Autofillフレームワーク

- リスク
  - Autofillサービスがマルウェア・うっかりウェアだったら、、、

「重要情報の漏えい」や「偽情報の自動入力」につながる危険性がある

Autofillフレームワーク

- 対策
  - ユーザーが適切なAutofillサービスだけを選択(インストール)する
  - 保存時にユーザーが適切に許可を与える
  - 自動入力時にユーザーが正しく候補を選択する

Autofill機能のセキュリティは、ユーザーの選択や許可によるところが大きい！

- アプリにできることは、、、

Autofillフレームワーク

- アプリにできる対策

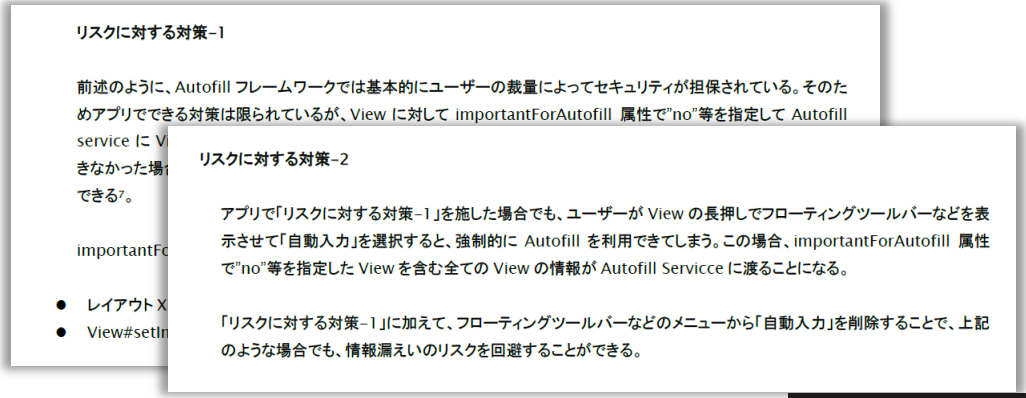
  唯一、

  View(Activity)毎にAutofill機能を無効にする

  ことだけ！

Autofillフレームワーク

- ガイドでは、以下の構成で解説
  - Autofillフレームワークの仕組み
  - セキュリティ上の懸案
  - リスクに対する対策



リスクに対する対策-1

前述のように、Autofill フレームワークでは基本的にユーザーの裁量によってセキュリティが担保されている。そのためアプリでできる対策は限られているが、View に対して importantForAutofill 属性で"no"等を指定して Autofill service に V
きなかった場
できる[7]。

importantFo

- レイアウト X
- View#setIn

リスクに対する対策-2

アプリで「リスクに対する対策-1」を施した場合でも、ユーザーが View の長押しでフローティングツールバーなどを表示させて「自動入力」を選択すると、強制的に Autofill を利用できてしまう。この場合、importantForAutofill 属性で"no"等を指定した View を含む全ての View の情報が Autofill Servicce に渡ることになる。

「リスクに対する対策-1」に加えて、フローティングツールバーなどのメニューから「自動入力」を削除することで、上記のような場合でも、情報漏えいのリスクを回避することができる。

Permission

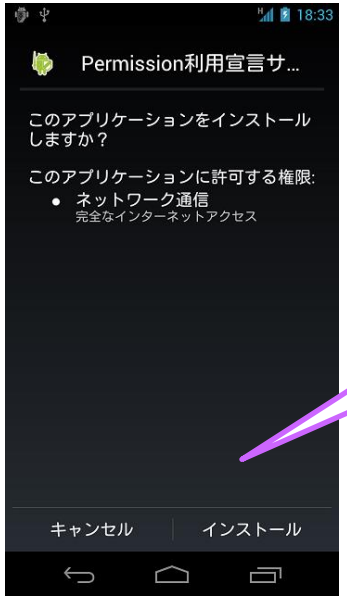
Permission

- 「5.2.3.6 Android 6.0以降のPermissionモデルの仕様変更について」について、Android 8.0における変更点を追記

端末のAndroid OSバージョン	アプリのtargetSDKVersion	アプリへの権限付与のタイミング	ユーザーによる権限制御
≧8.0	≧26	アプリ実行時(付与はPermission単位)	あり
	＜26	アプリ実行時(付与はGroup単位)	あり
	＜23	インストール時	あり(早急な対応が必要)
≧6.0	≧23	アプリ実行時(付与はGroup単位)	あり
	＜23	インストール時	あり(早急な対応が必要)
≦5.1	≧23	インストール時	なし
	＜23	インストール時	なし

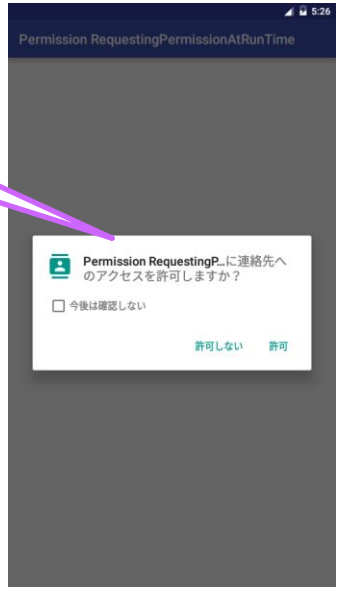
Android 6.0 (Marshmallow)の変更点

- Runtime Permissionの導入



インストール時に確認

実行時に確認



従来のPermission確認画面

新しいPermission確認画面

Android 6.0 (Marshmallow)の変更点

- 保護レベルがdangerousのpermissionのみ影響
  - ① 実行中の**必要に応じたユーザー許諾**
  - ② **Permission Group単位での権限管理**
    同じPermission Groupに属するpermissionは
    一度の要求で全てgrantされる
  - ③ **ユーザーによる許可の取り消し (revoke)**
  - – targetSdkがAndroid Mより前のバージョンでも、
    Android M 端末上では、ユーザーは権限をrevokeできる
  - – Android Mより前のOSの動作する端末上では
    新しいモデルは適用されない

Android 6.0 (Marshmallow)の問題点

- Permission Group単位での権限管理

例えば、*READ_CONTACTS*パーミッションを許可すると、

```
// AndroidManifes.xml内の宣言
<uses-permission android:name="android.permission.
READ_CONTACTS"/>
```

```
// READ_CONTACTSを要求する例
requestPermissions(
    new String[] {Manifest.permission. READ_CONTACTS},
    REQUEST_CODE);
```

同じGroupの*WRITE_CONTACTS*パーミッションが**不要な場合でも一緒に付与されてしまう**

Android 8.0 (Oreo)の改善点

- ユーザーから見たUIは同じ
  - Group単位での許可ダイアログの表示
- 実装において明示的な利用宣言・要求が必須になった

```
// AndroidManifes.xml内の宣言
<uses-permission android:name="android.permission.
WRITE_CONTACTS"/>
```

```
// WRITE_CONTACTSを要求する例
requestPermissions(
    new String[] {Manifest.permission. WRITE_CONTACTS},
    REQUEST_CODE);
```

明示的宣言が無い場合は許可されない

- 取消単位はGroupのまま同じ

Account Manager

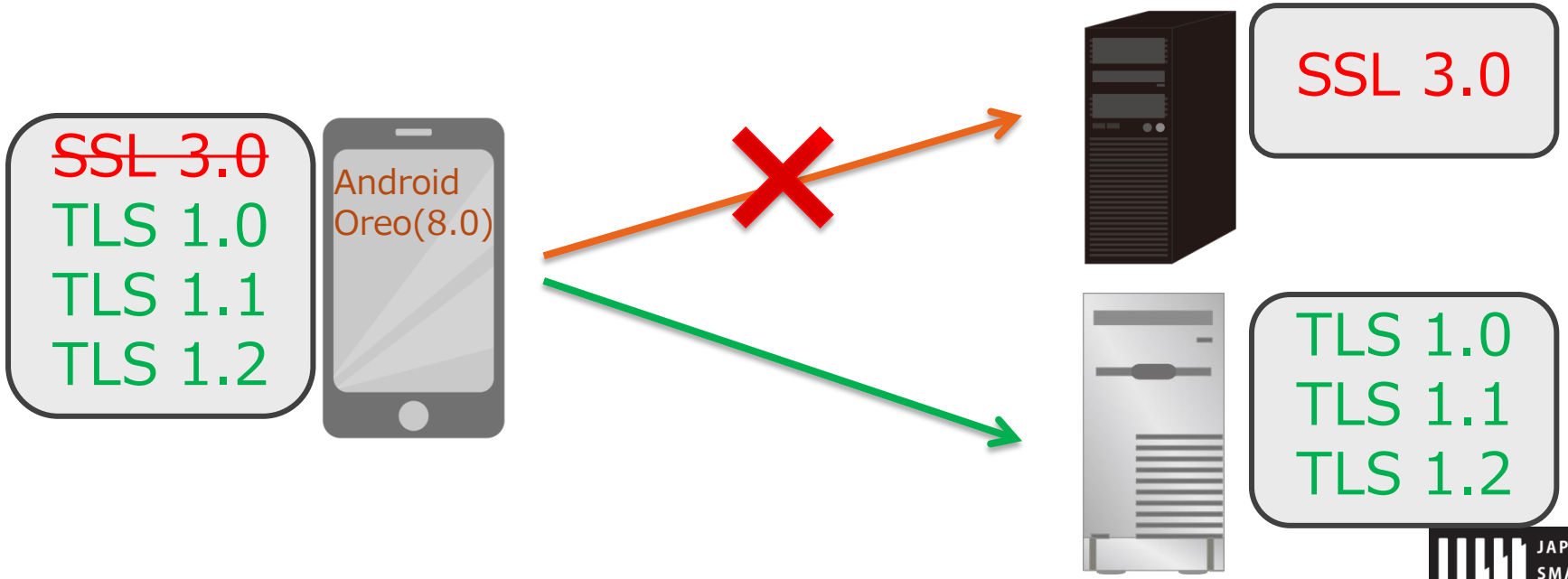
Account Manager

- 追加
  - 5.3.3.3 Android 8.0（API Level 26）以降で署名の一致しないAuthenticatorのアカウントを読めるケース
    - Android 8.0（API Level 26）以降で署名の一致しないAuthenticatorのアカウント情報を取得できるケースとその対策に関する記述を追記
- 変更・拡充
  - 5.3.2.6 Account Managerにパスワードを保存しない　　（推奨）
    - Android 7.0(API Level 24)以降のパスワード保存場所に関する記述を追記
  - 5.3.3.1 Account Managerの利用とPermission
    - Android 6.0(API Level 23)以降およびAndorid 8.0(API Level 26)以降のAccountManagerに関するPermissionとメソッドの対応を追記
  - 5.3.2.4 KEY_INTENTには、ログイン画面Activityのクラス名を指定した明示的Intentを与える　　（必須）
    - Android 4.4（API Level 19）前後でのKey Intentの挙動の違いについて追記

SSL3.0非サポート

SSLv3非サポートについて

- OSレベルでSSLv3のサポートを中止！



Android ID

変更点

- Android Nougat (7.1) 以前

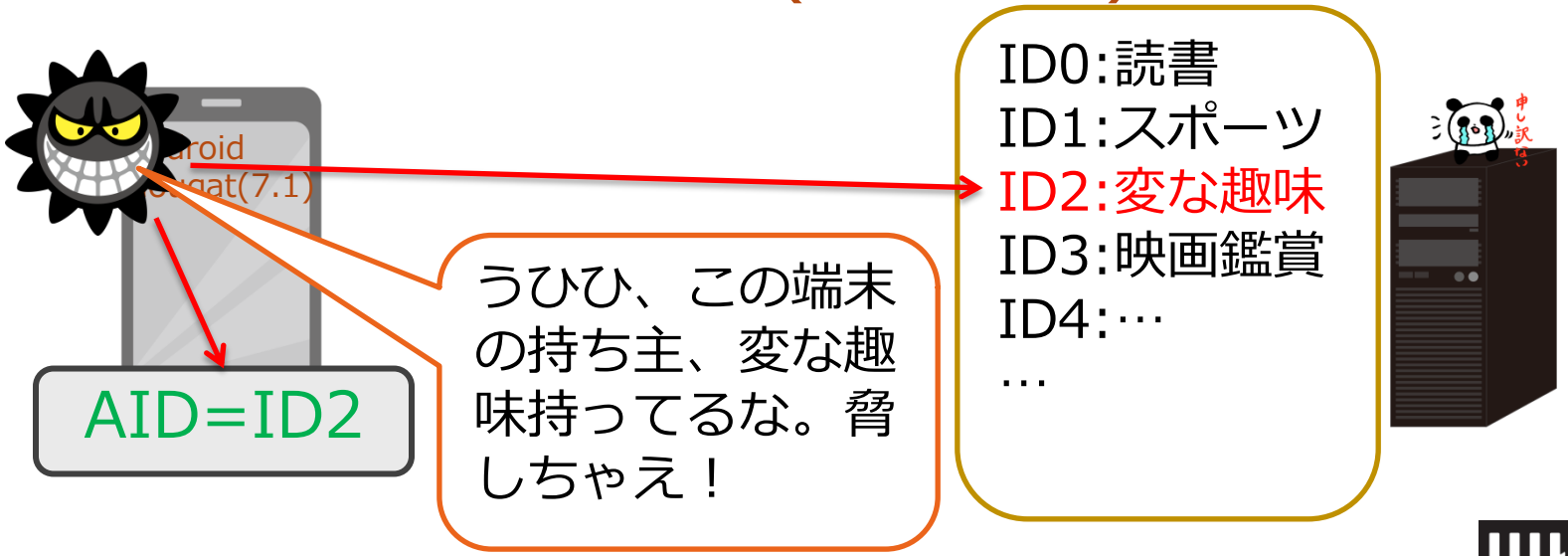
  端末固有値:
  端末毎にユニークな識別子

- Android Oreo (8.0) 以降

  アプリ固有値:
  端末 x アプリ毎にユニークな識別子

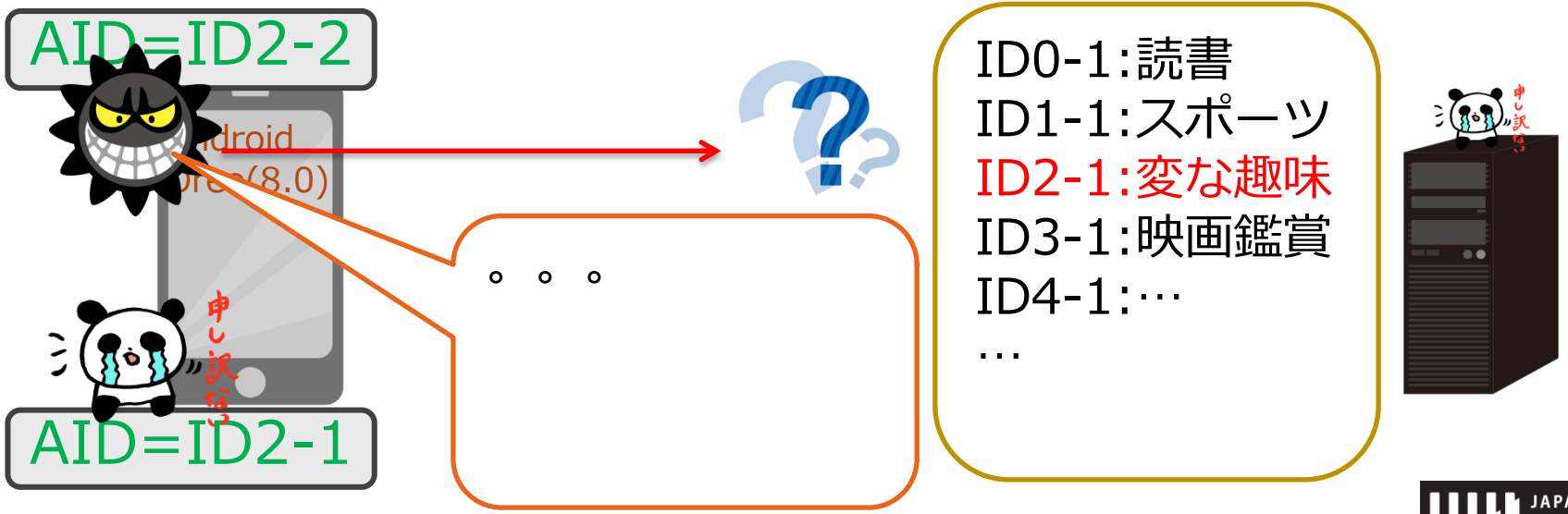
何が違うのか？

- 例えば、あるアプリがIDと趣味を多数の端末から集めてWebページに(うっかり)掲載



何が違うのか？

- Android 8.0の端末だと、各アプリ取得できるIDが違うので、



Android ID

- 特徴
  - 端末リセットするまで値は変わらない
  - 7.1以前：端末固有な値
  - 8.0以降：端末 X アプリ固有な値
    - 正確には、開発者(署名)単位なので再インストール後も値は不変
- 多少安全性が加味されたが、

「ユーザーによる取り換え困難なID」に変わりないので、プライバシー情報と紐づけた利用には注意が必要

WebView

WebView

- Network Security Configuration の平文通信禁止フラグが有効になった
  - Android 7.1まではWebViewに関しては効果が無かった

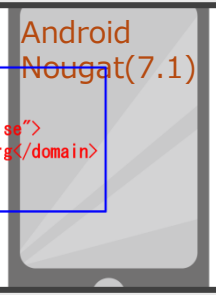
```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
    <domain-config cleartextTrafficPermitted="false">
        <domain includeSubdomains="true">jssec.org</domain>
    </domain-config>
</network-security-config>
```

Android 7.xでは

- 平文通信を禁止しても、、、

※(注意)理解優先のために実装は正確ではありません

URL uri = “**http**://example.com/api”;
uri.openConnect ();




```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
    <domain-config cleartextTrafficPermitted="false">
        <domain includeSubdomains="true">jssec.org</domain>
    </domain-config>
</network-security-config>
```

WebView wv = new WebView();
wv.loadUrl(“**http**://example.com/”);

example.com

WebView は平文通信できてしまう

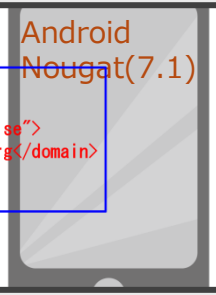
JAPAN SMARTPHONE SECURITY ASSOCIATION

Android 8.0以降は

- 平文通信を禁止すれば、、、

※(注意)理解優先のために実装は正確ではありません

URL url = “**http**://example.com/api”;
url.openConnection ();




```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
    <domain-config cleartextTrafficPermitted="false">
        <domain includeSubdomains="true">jssec.org</domain>
    </domain-config>
</network-security-config>
```

WebView wv = new WebView();
wv.loadUrl(“**http**://example.com/”);

example.com

接続方法によらず平文通信を防止できる

JAPAN SMARTPHONE SECURITY ASSOCIATION

(参考) Network Security Configurationとは

- Android 7.0 から追加された機能で、アプリ毎にHTTPS通信の制御ができる
  - プライベート証明書を使ったHTTPS通信
  - ピンニングによる証明書検証
  - 非暗号化（HTTP）通信の抑制
  - etc…

```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
    <domain-config cleartextTrafficPermitted="false">
        <domain includeSubdomains="true">jssec.org</domain>
    </domain-config>
</network-security-config>
```

ガイドHTML版のご紹介

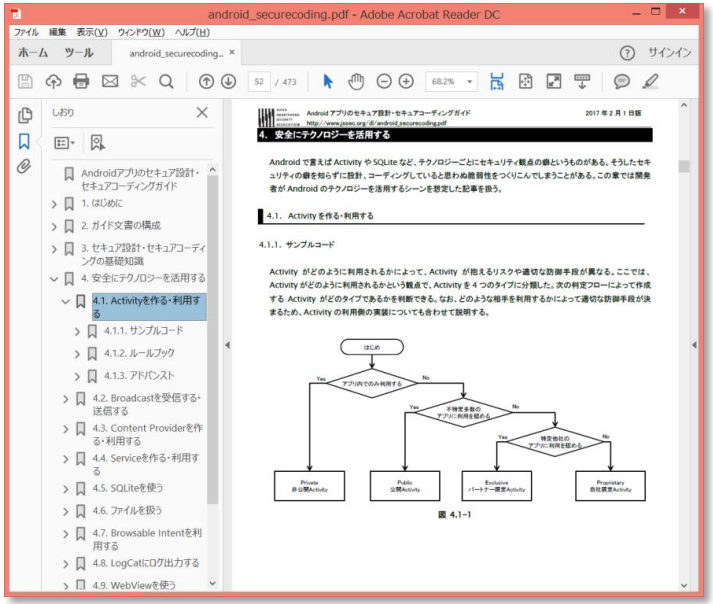
HTML版について

- URL
  - http://www.jssec.org/dl/android_securecoding/

- 特徴
  - 見出しの階層構造
  - 各見出しへの直接リンク
  - ソースコードの可読性向上

概要

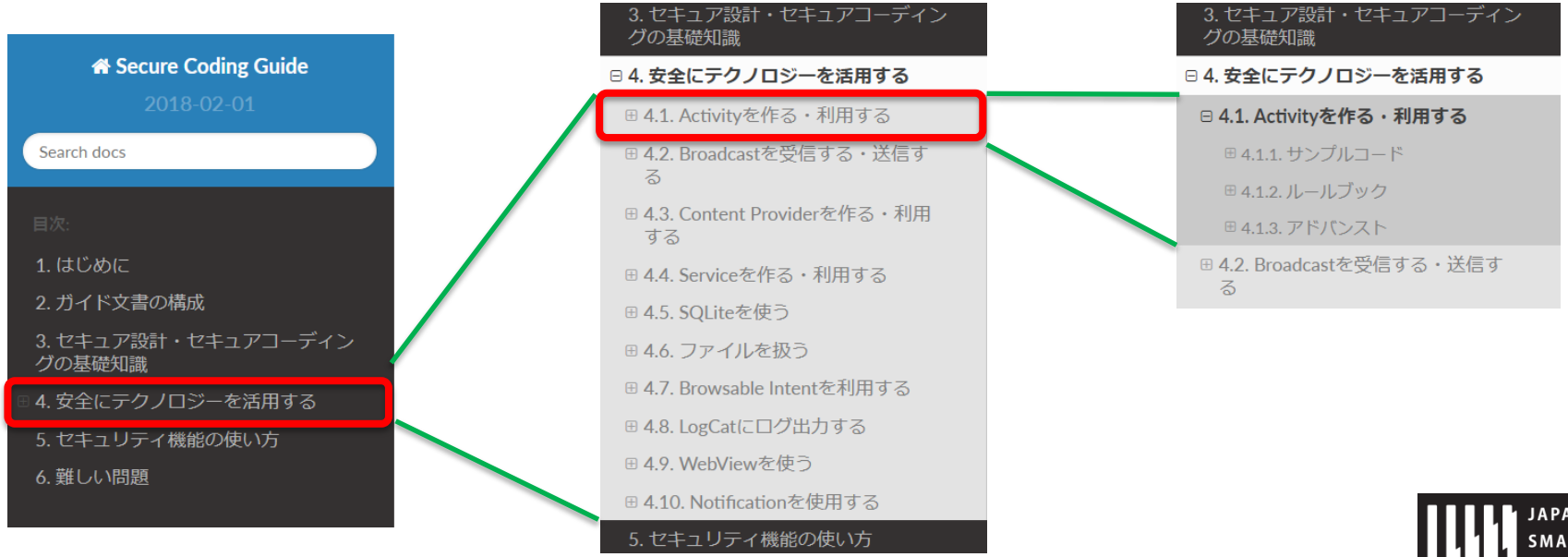
PDF版ガイド



HTML版

見出し

階層的な見出しで、目的の記事に素早くアクセス！



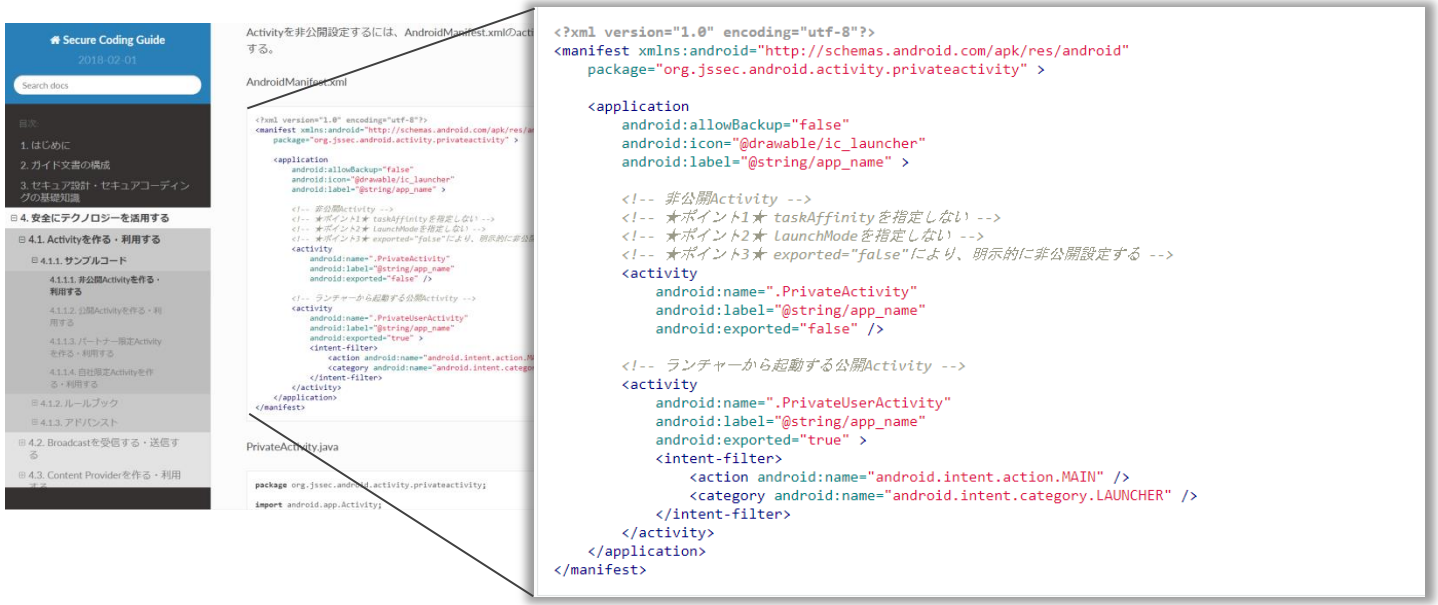
各見出しへの直接リンク

章・節にも直接リンクできて、参照が便利に！

「5.2.3.3. APKの改ざんを検出する」

コードの可読性向上

ページをまたがず、色も付いて、見やすく！




```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="org.jssec.android.activity.privateactivity" >

    <application
        android:allowBackup="false"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name" >

        <!-- 非公開Activity -->
        <!-- ★ポイント1★ taskAffinityを指定しない -->
        <!-- ★ポイント2★ launchModeを指定しない -->
        <!-- ★ポイント3★ exported="false"により、明示的に非公開設定する -->
        <activity
            android:name=".PrivateActivity"
            android:label="@string/app_name"
            android:exported="false" />

        <!-- ランチャーから起動する公開Activity -->
        <activity
            android:name=".PrivateUserActivity"
            android:label="@string/app_name"
            android:exported="true" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

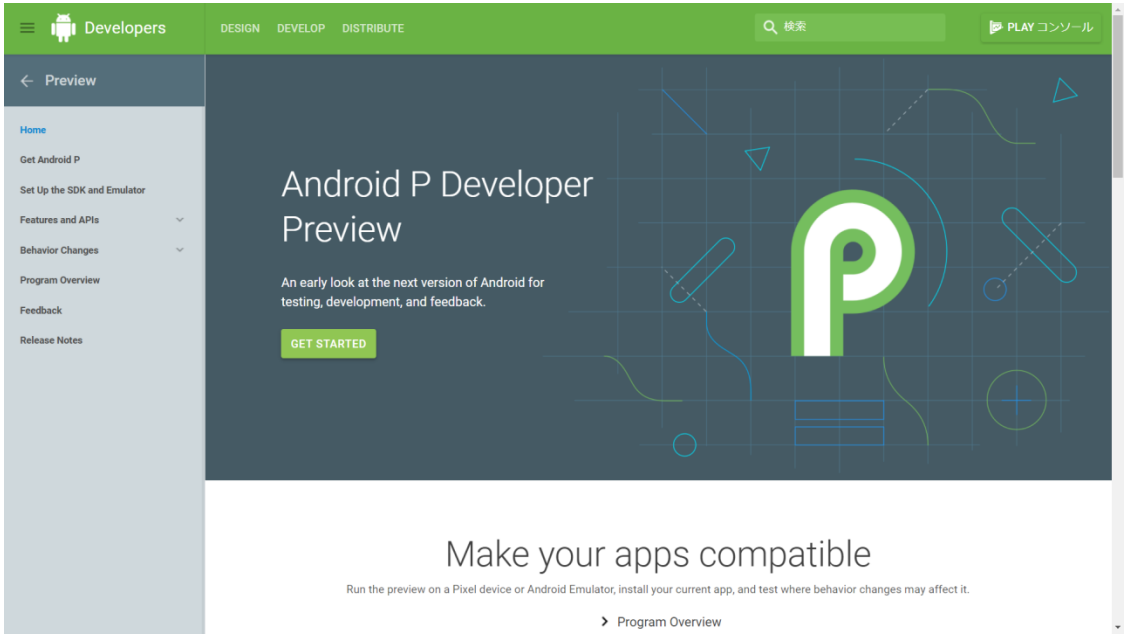

JAPAN SMARTPHONE SECURITY ASSOCIATION

セキュアコーディングガイド

第10版(告知)

Android P (9.0) 開発者プレビュー公開！

Oreo (8.0) との差分などの情報が開示！



第10版

- おそらく、Android P 対応が中心になります！

乞う、ご期待！

セキュアコーディングガイド

さいごに

ご協力いただける方は
下記の要領でご連絡ください

1. JSSECに会員としてご参加ください

- 会員かどうかは下記URLで確認
  - http://www.jssec.org/members/
- 会員として無理な方は、Androidセキュリティ部にご参加ください(下記URL)
  - https://groups.google.com/group/android-security-japan
  - 右の(4)では「Androidセキュリティ部」と記載してください

2. 次の書式でメール送信してください

```
To: Akira.Ando@sony.com
Subject: JSSECセキュアコーディングWG参加
本文:
(1) Google account: (メアドを記載)
(2) First name: (名前を記載)
(3) Last name: (名字を記載)
(4) Organization: (組織名を記載)
(5) Git access: (必要 or 不要)
```

- 各種アカウント発行後、メール返信にてご連絡いたします

  (1)はGmailメアドまたはご自身のメアドをGoogleアカウント化したもの

  (4)は下記URLページ内から選択
  - http://www.jssec.org/members/

ご協力お願い致します

JAPAN SMARTPHONE SECURITY ASSOCIATION