Androidアプリセキュリティ

スマートフォン・オンラインゲームのセキュリティを考える
【JOGA・JSSEC共催】
2011年11月17日

日本スマートフォンセキュリティフォーラム
アプリケーションWG セキュアコーディングTF
松並　勝 <Masaru.Matsunami@jp.sony.com>

Agenda

- はじめに
- Androidアプリセキュリティ
- セキュア設計・セキュアコーディング
- セキュア開発
- まとめ

はじめに

かんたんな自己紹介です

ソニーGp.内のソフト専門会社

Sony Digital Network Applications, Inc.

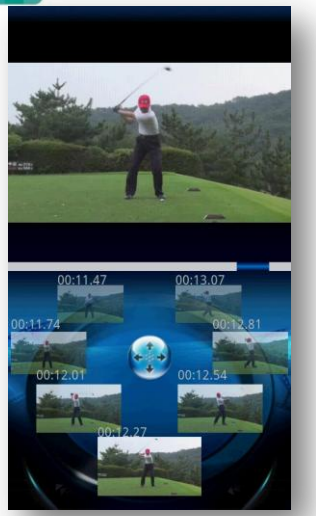
- 多種多様なソニー製品のソフトを開発

http://www.sony.jp/

スマフォアプリも自社開発

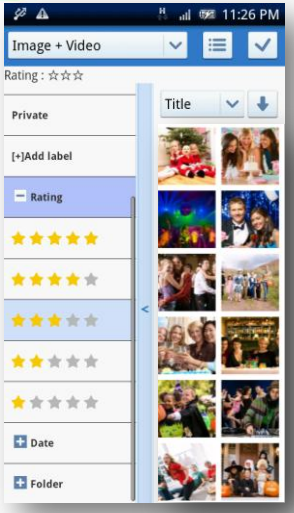
Sony Digital Network Applications, Inc.

frame grabber



video trimmer

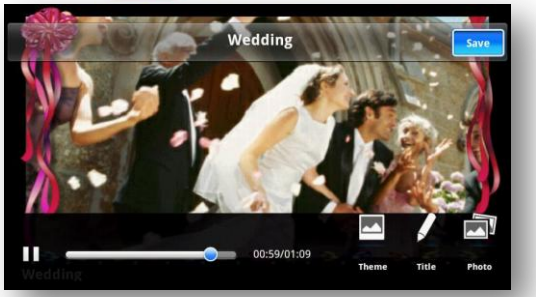
picture manager



video connector

photo editor

photo movie creator



ソニー製品のセキュリティ確保

Sony Digital Network Applications, Inc.

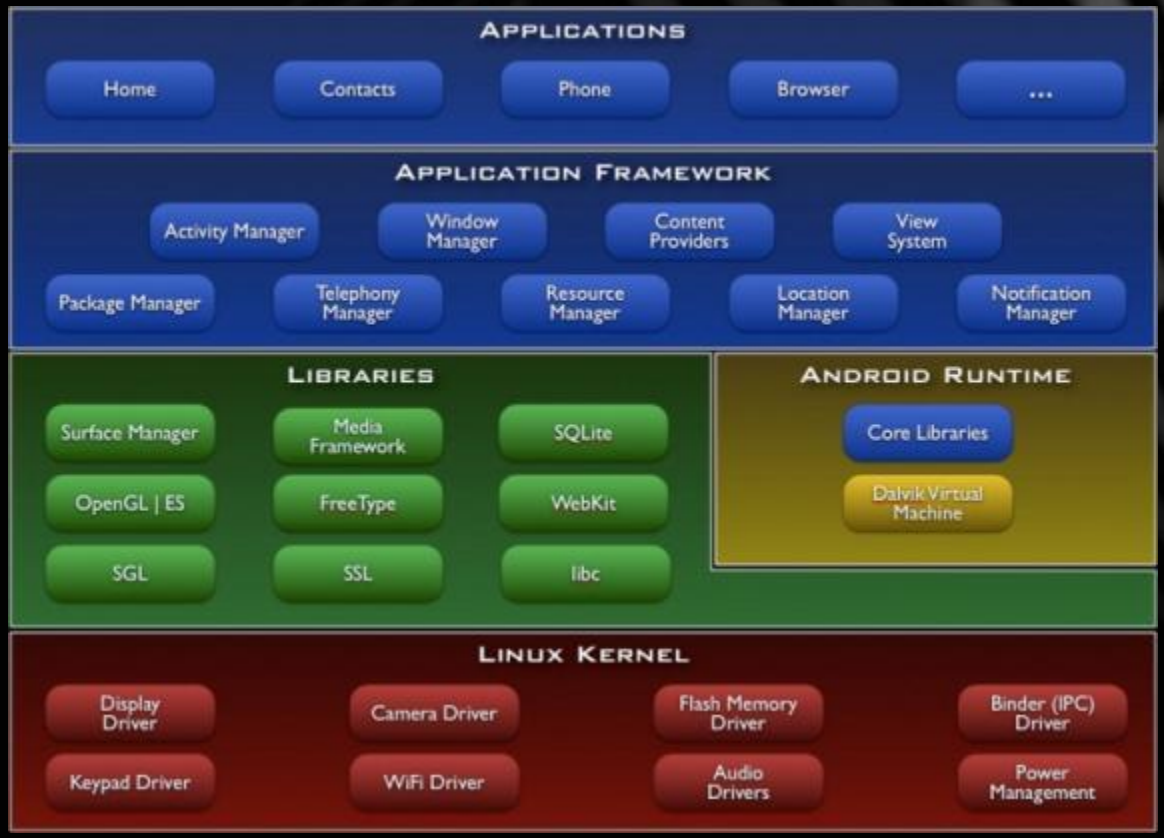
- セキュリティ専門組織がある
  - SSAG: Software Security Assurance Group
  - 現在、9名のセキュリティ技術者（コンサルタント）で構成
  - わたしはこの組織のマネージャー
- ソニー製品のセキュリティ確保をリード
  - 2002年～　PCアプリ、情報家電のセキュリティ
  - 2009年～　Android製品まるごとのセキュリティ
- 地味ですが大切な活動、2002年から継続
  - 気が付くとインターネット時代に欠かせない技術

Androidアプリセキュリティ

それでは本題にはいります

今日のスコープ

- 普段はぜんぶやっていますが、今日はアプリレイヤーのセキュリティに話を絞ります



アプリのセキュリティ

端末のセキュリティ

Androidスマフォに入っている情報

情報	備考
電話番号	
電話帳	
通話履歴	受発信の日時や番号
IMEI	携帯電話の端末ID
IMSI	回線契約者ID
各種センサー情報	GPS、地磁気、加速度…
Settings	各種設定情報…
アカウント	Googleアカウント…
アプリ一覧	
メディアデータ	写真、動画、音楽、録音…

情報	備考
Eメールアドレス	
Eメールのメールボックス	送受信メール本文、添付…
SMSのメールボックス	送受信SMSメッセージ本文…
Webブックマーク	
Web閲覧履歴	
カレンダー	予定、ToDo、イベント…
Facebook	ID、コンテンツキャッシュ…
Twitter	ID、コンテンツキャッシュ…
mixi	ID、コンテンツキャッシュ…
…	

- 端末情報、ユーザーデータ、クラウドサービスに接続するためのアカウント情報（認証情報）が含まれる
- インストールするアプリが増えると、どんどんユーザーデータがスマフォに持ち込まれる

電話帳（Contacts）の1件のエントリに含まれる情報

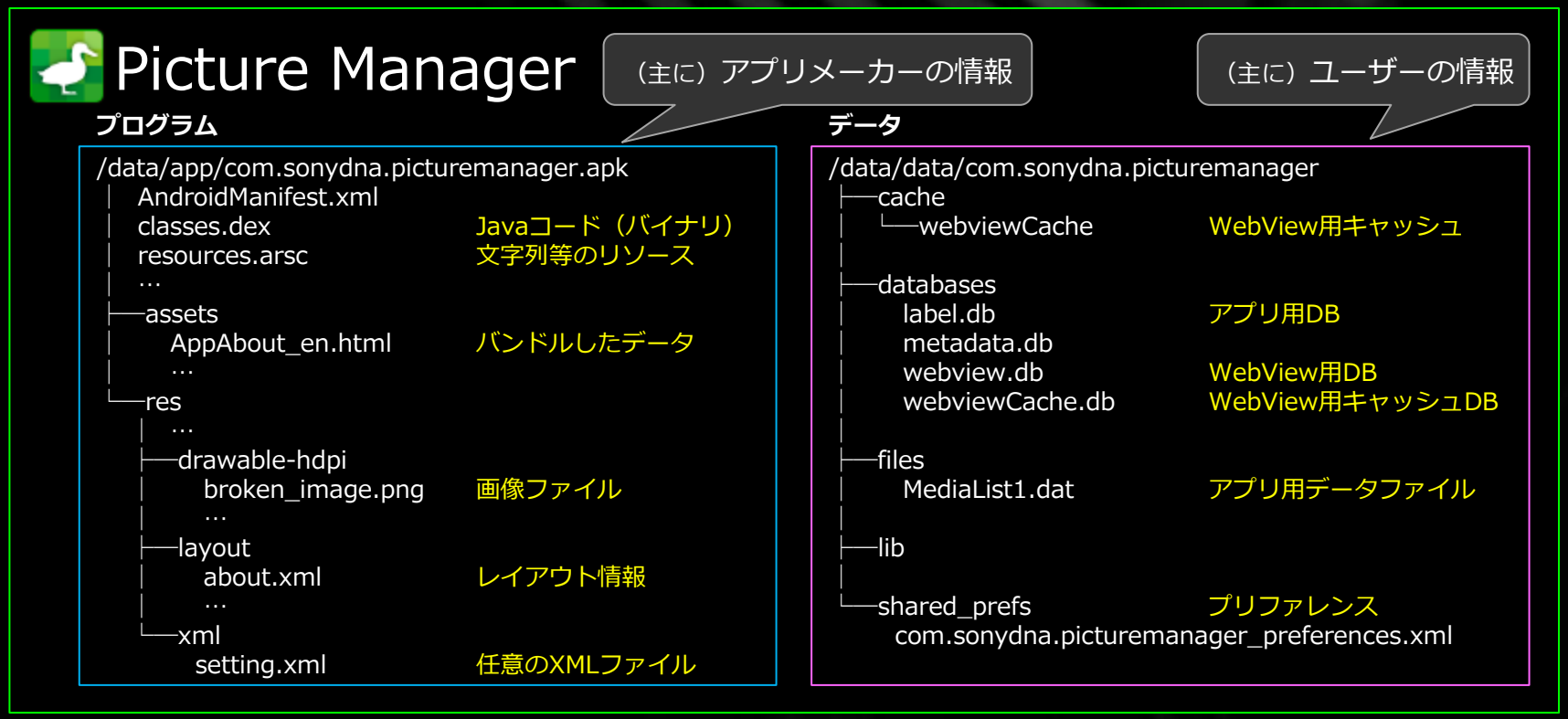
情報	内容
名前	表示名、名前、苗字、ミドルネーム…
電話番号	自宅、携帯電話、仕事、FAX、MMS…
Eメールアドレス	自宅、仕事、携帯電話…
プロフィール画像	サムネール画像、大きな画像…
インスタントメッセンジャー	AIM、MSN、Yahoo、Skype、QQ、Google Talk、ICQ、Jabber、Netmeeting…
ニックネーム	略称、イニシャル、旧姓、別名…
住所	国、郵便番号、地域、地方、町、通り…
グループ	お気に入り、家族、友達、同僚…
ウェブサイト	ブログ、プロフィールサイト、ホームページ、FTPサーバー、自宅、会社…
イベント	誕生日、記念日、その他…
関係する人物	配偶者、子供、父、母、マネージャー、助手、同棲関係、パートナー…
SIPアドレス	自宅、仕事、その他…
…	

- スマフォのユーザーに関する情報だけではなく、そのユーザーの1ホップ先の人々の情報もスマフォには含まれている

ユーザーの個人生活・社会生活がスマフォに集約される

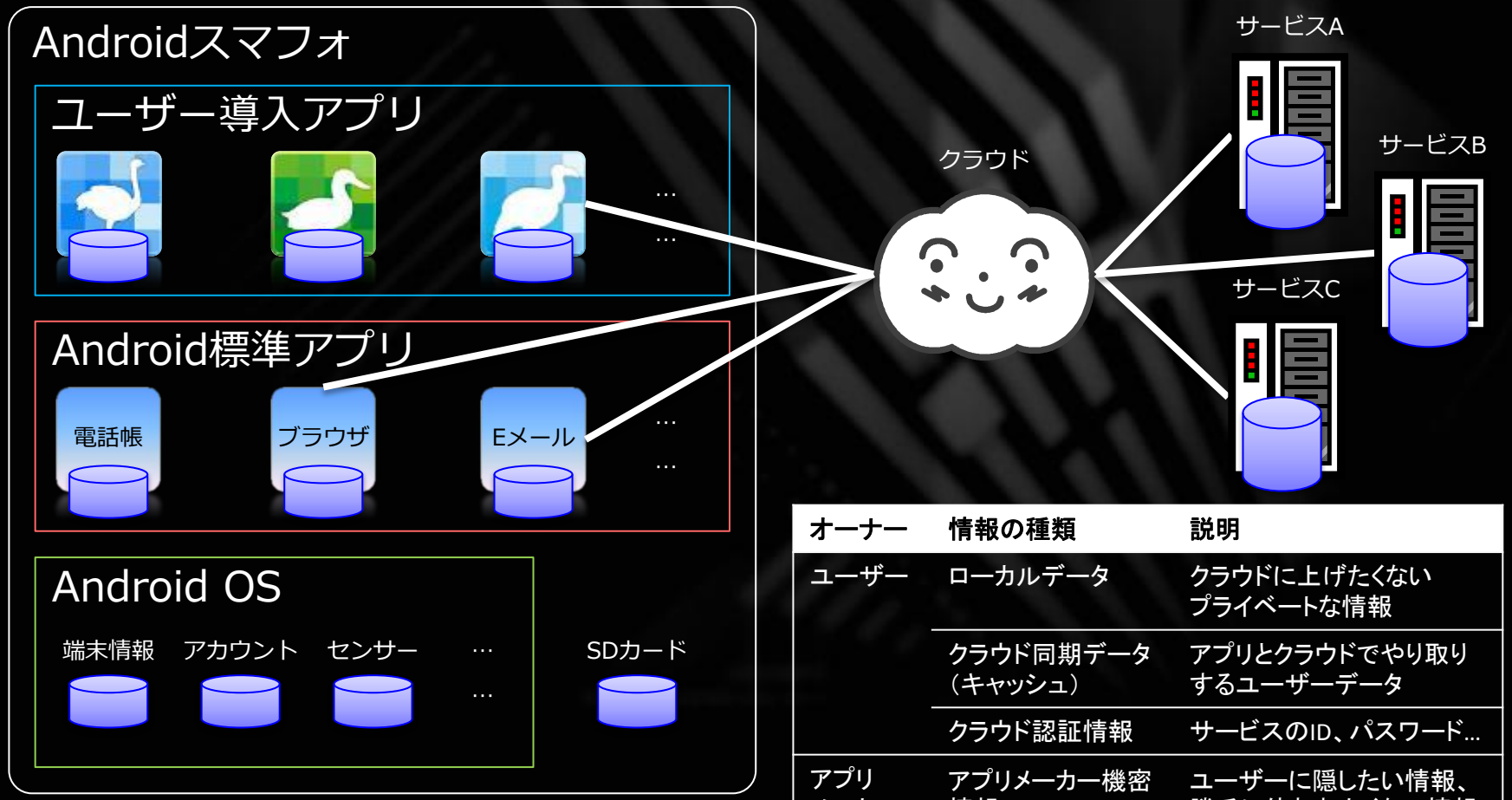
- 導入するアプリが増えれば増えるほど、スマフォはユーザーの生活の中で使われる機会が増える
- 導入されるアプリを通じて、ユーザーの個人生活・社会生活にかかわる情報もスマフォに溜め込まれていく
- 電話帳やSNSアプリを通じて、そのスマフォのユーザーに限らず、他の人々にかかわる情報もスマフォに溜め込まれていく

アプリが抱えている情報



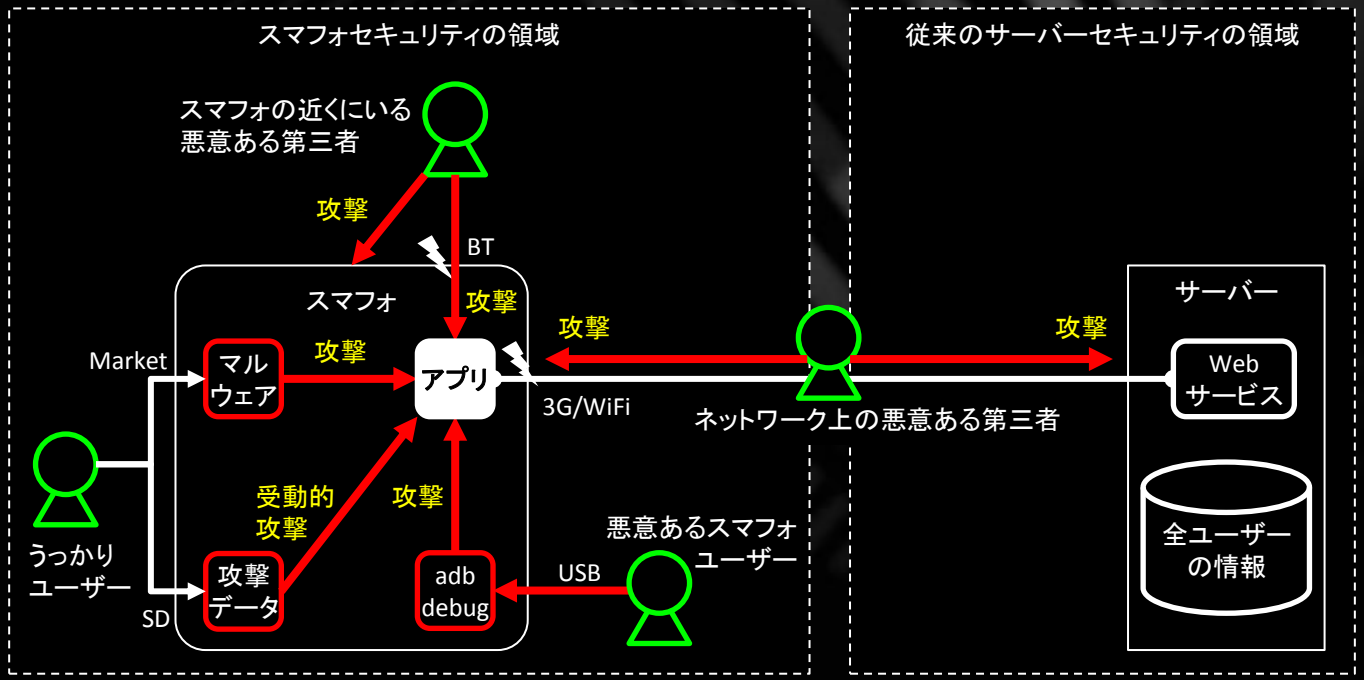
- アプリメーカーの情報の中には、ユーザーには見せたくない情報、勝手に利用されたくない情報もある

Android端末内にある情報の俯瞰図



オーナー	情報の種類	説明
ユーザー	ローカルデータ	クラウドに上げたくないプライベートな情報
	クラウド同期データ（キャッシュ）	アプリとクラウドでやり取りするユーザーデータ
	クラウド認証情報	サービスのID、パスワード…
アプリメーカー	アプリメーカー機密情報	ユーザーに隠したい情報、勝手に使われたくない情報

アプリはあちこちから狙われる



サンドボックス

- アプリは、抱えている情報（や機能）を悪意ある他のアプリから守る義務がある
- サンドボックスとは、アプリ内の情報や機能が、他のアプリから読み書きさせない防御の仕組み

※SDカード（external storage）上の情報はほとんど守られていない

アプリ間連携

- アプリ同士が機能や情報を融通しあうことで、ユーザー体験を向上できる
  - Androidにはリッチなアプリ間連携の仕組みが用意されている
  - Activity, Service, Content Provider, Broadcast Receiver…
- サンドボックスに穴をあける行為である
  - アプリは各々が抱える情報（や機能）を、他のアプリから勝手に利用されないように守る責務がある
  - 意図した範囲で、想定外の利用ができないように、サンドボックスに穴をあける絶妙な設計・コーディングが求められる

セキュア設計・セキュアコーディング

インターネット時代のソフトウェア技術者のたしなみです

アプリ開発者の責務

- 自分のアプリは自分で守る
  - アプリが抱えているユーザーデータ、メーカー機密情報が他のアプリから勝手に利用されないように、セキュア設計・セキュアコーディングする
  - 特にユーザーデータはお客様からお預かりしている大切な資産であることを肝に銘じる
- 他のアプリに迷惑をかけない
  - 自分のアプリに踏み台として悪用できる脆弱性があると、自分のアプリを信用してアプリ間連携してくれている他のアプリに迷惑をかけることになる
  - 脆弱性を自分のアプリに作り込んでしまわないように、セキュア設計・セキュアコーディングする

JSSECセキュアコーディングTF

- Androidアプリのセキュア設計・セキュアコーディングTipsを集めて文書化し公開
  - みんなでよってたかってTipsを集めて、みんなで共有
  - Android開発初心者にも役立つ文書にしたい
  - 「Androidって安全だよね」とみんなが言っている未来をつくりたい
- JOGAのみなさんもぜひご参加ください！
  - JOGAはJSSECの特別会員です

Androidアプリの
セキュア設計・セキュアコーディングガイド

- 防御のテクニック
  - PermissionとProtection Level
  - 暗号化、電子署名
  - 暗号化通信
  - Shared UID
  - …
- 穴を作らないテクニック
  - Activity
  - Content Provider
  - Broadcast Receiver
  - Service
  - SQLite
  - WebKit
  - HTML5
  - …

- こんな粒度でTipsの文書化を進めています

例：Activity (1/3)

- アプリ開発者は、Activityを実装する場合、次の対策を検討しなければならない。
  1. 内部使用のActivityを非公開とする
  2. 呼び出し元アプリを制限する
  3. 受信Intentを処理する前に、Intentの内容の安全性を確認する
  4. 呼び出し側に結果情報を返す場合、機密に気を付ける
  5. uses-permissionにより得た機能資産・情報資産の保護を検討する

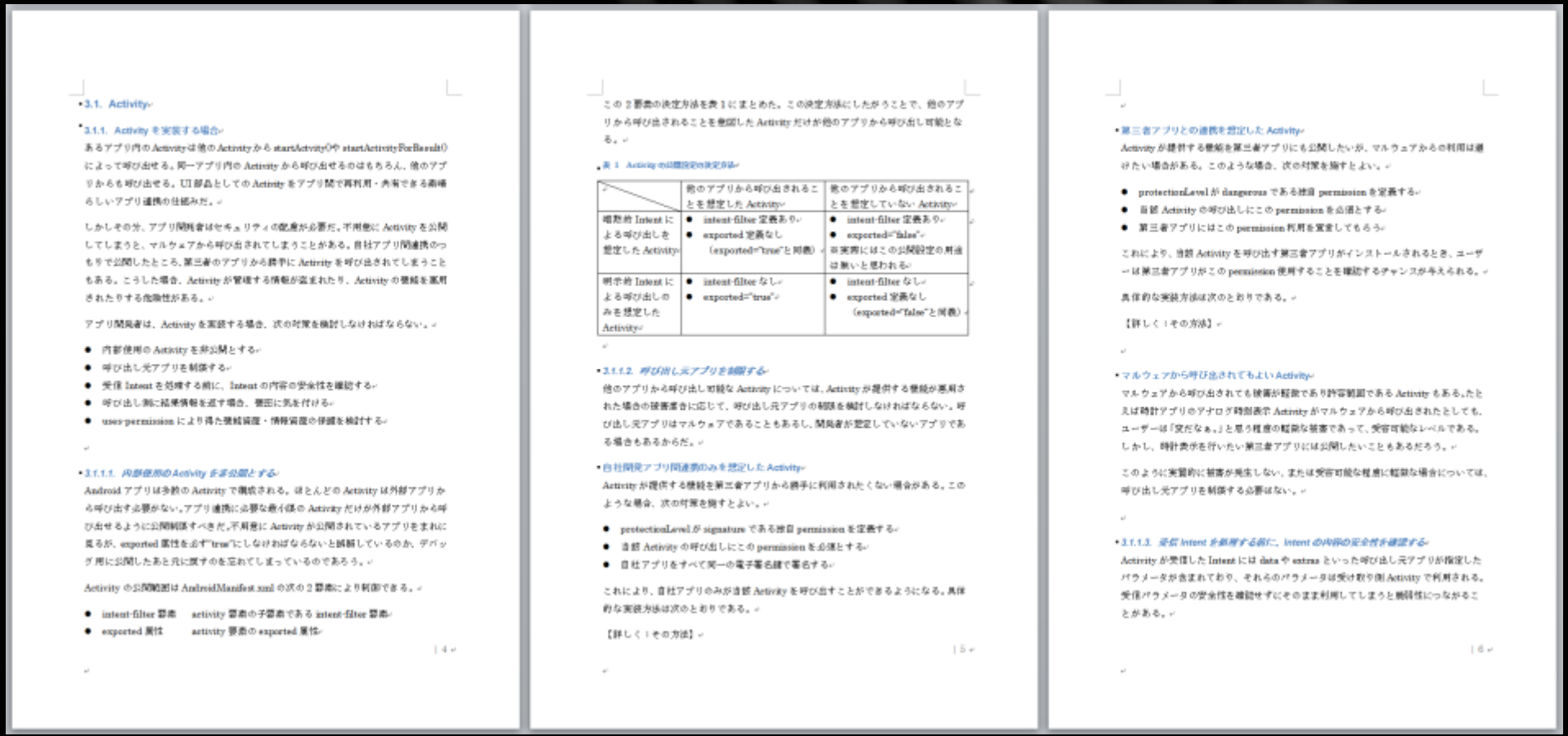
例：Activity (2/3)

1. 内部使用のActivityを非公開とする

- Activityの公開範囲はAndroidManifest.xmlの次の2要素により制御できる。
  - intent-filter要素 activity要素の子要素であるintent-filter要素
  - exported属性　activity要素のexported属性
- この2要素の決定方法を表 1にまとめた。この決定方法にしたがうことで、他のアプリから呼び出されることを意図したActivityだけが他のアプリから呼び出し可能となる。

表1	他のアプリから呼び出されることを想定したActivity	他のアプリから呼び出されることを想定していないActivity
暗黙的Intentによる呼び出しを想定したActivity	● intent-filter定義あり ● exported定義なし（exported=”true”と同義）	● intent-filter定義あり ● exported=”false” ※実際にはこの公開設定の用途は無いと思われる
明示的Intentによる呼び出しのみを想定したActivity	● intent-filterなし ● exported=”true”	● intent-filterなし ● exported定義なし（exported=”false”と同義）

例：Activity (3/3)



3.1. Activity

3.1.1. Activity を実装する場合

あるアプリ内の Activity は他の Activity から startActivity() や startActivityForResult() によって呼び出せる。同一アプリ内の Activity から呼び出せるのはもちろん、他のアプリからも呼び出せる。UI 部品としての Activity をアプリ間で再利用・共有できる素晴らしいアプリ連携の仕組みだ。

しかしその分、アプリ開発者はセキュリティの配慮が必要だ。不用意に Activity を公開してしまうと、マルウェアから呼び出されてしまうことがある。自社アプリ間連携のつもりで公開したところ、第三者のアプリから誤って Activity を呼び出されてしまうこともある。こうした場合、Activity が管理する情報が盗まれたり、Activity の機能を悪用されたりする危険性がある。

アプリ開発者は、Activity を実装する場合、次の対策を検討しなければならない。

- 内部使用の Activity を非公開とする
- 呼び出し元アプリを制限する
- 受信 Intent を処理する前に、Intent の内容の安全性を確認する
- 呼び出し側に結果情報を返す場合、宛先に気を付ける
- uses-permission により得た機能資産・情報資産の保護を検討する

3.1.1.1. 内部使用の Activity を非公開とする

Android アプリは多数の Activity で構成される。ほとんどの Activity は外部アプリから呼び出す必要がない。アプリ連携に必要な最小限の Activity だけが外部アプリから呼び出せるように公開制限すべきだ。不用意に Activity が公開されているアプリをまれに見るが、exported 属性を必ず"true"にしなければならないと誤解しているのか、デバッグ用に公開したあと元に戻すのを忘れてしまっているのであろう。

Activity の公開範囲は AndroidManifest.xml の次の 2 箇所により制御できる。

- intent-filter 要素　　activity 要素の子要素である intent-filter 要素
- exported 属性　　activity 要素の exported 属性

この 2 箇所の決定方法を表 1 にまとめた。この決定方法にしたがうことで、他のアプリから呼び出されることを意図した Activity だけが他のアプリから呼び出し可能となる。

表 1　Activity の公開設定の決定方法

	他のアプリから呼び出されることを想定した Activity	他のアプリから呼び出されることを想定していない Activity
暗黙的 Intent による呼び出しを想定した Activity	● intent-filter 定義あり ● exported 定義なし（"true"と同義）	● intent-filter 定義あり ● exported="false" ※実際にはこの公開設定の用途は無いと思われる
明示的 Intent による呼び出しのみを想定した Activity	● intent-filter なし ● exported="true"	● intent-filter なし ● exported 定義なし（"false"と同義）

3.1.1.2. 呼び出し元アプリを制限する

他のアプリから呼び出し可能な Activity については、Activity が提供する機能が悪用された場合の被害度合に応じて、呼び出し元アプリの制限を検討しなければならない。呼び出し元アプリはマルウェアであることもあるし、開発者が想定していないアプリである場合もあるからだ。

自社開発アプリ間連携のみを想定した Activity

Activity が提供する機能を第三者アプリから誤手に利用されたくない場合がある。このような場合、次の対策を施すとよい。

- protectionLevel が signature である独自 permission を定義する
- 当該 Activity の呼び出しにこの permission を必須とする
- 自社アプリをすべて同一の電子署名鍵で署名する

これにより、自社アプリのみが当該 Activity を呼び出すことができるようになる。具体的な実装方法は次のとおりである。

【詳しく：その方法】

第三者アプリとの連携を想定した Activity

Activity が提供する機能を第三者アプリにも公開したいが、マルウェアからの利用は避けたい場合がある。このような場合、次の対策を施すとよい。

- protectionLevel が dangerous である独自 permission を定義する
- 当該 Activity の呼び出しにこの permission を必須とする
- 第三者アプリにはこの permission 利用を宣言してもらう

これにより、当該 Activity を呼び出す第三者アプリがインストールされるとき、ユーザーは第三者アプリがこの permission 使用することを確認するチャンスが与えられる。

具体的な実装方法は次のとおりである。

【詳しく：その方法】

マルウェアから呼び出されてもよい Activity

マルウェアから呼び出されても被害が軽微であり許容範囲である Activity もある。たとえば時計アプリのアナログ時刻表示 Activity がマルウェアから呼び出されたとしても、ユーザーは「変だなぁ」と思う程度の軽微な被害であって、受容可能なレベルである。しかし、時計表示を行いたい第三者アプリには公開したいこともあるだろう。

このように実質的に被害が発生しない、または受容可能な程度に軽微な場合については、呼び出し元アプリを制限する必要はない。

3.1.1.3. 受信 Intent を処理する前に、Intent の内容の安全性を確認する

Activity が受信した Intent には data や extras といった呼び出し元アプリが指定したパラメータが含まれており、それらのパラメータは受け取り側 Activity で利用される。受信パラメータの安全性を確認せずにそのまま利用してしまうと脆弱性につながることがある。

ガイドの進捗状況

ちょうどメンバー募集
をはじめたところです

今日はこのためにきました！

TFメンバー募集！

Tips募集！

連絡先： JSSEC事務局 <sec@jssec.org>
Subject: セキュアコーディングTF 参加申込み

セキュア開発

ここから先はセキュリティを考慮した開発スタイルについて説明します。Androidアプリに限定しない、より一般的な話です

セキュリティ技術者のジレンマ

- とある不幸なできごと
  1. 前職セキュリティベンダーから弊社に転職してきた2002年、とある製品の出荷前に脆弱性検査を任されました。
  2. もてる力を最大限発揮してようやく脆弱性を見つけました。
  3. その瞬間から険悪なムード。
     出荷に間に合わせるべく、開発者の徹夜作業が始まりました。

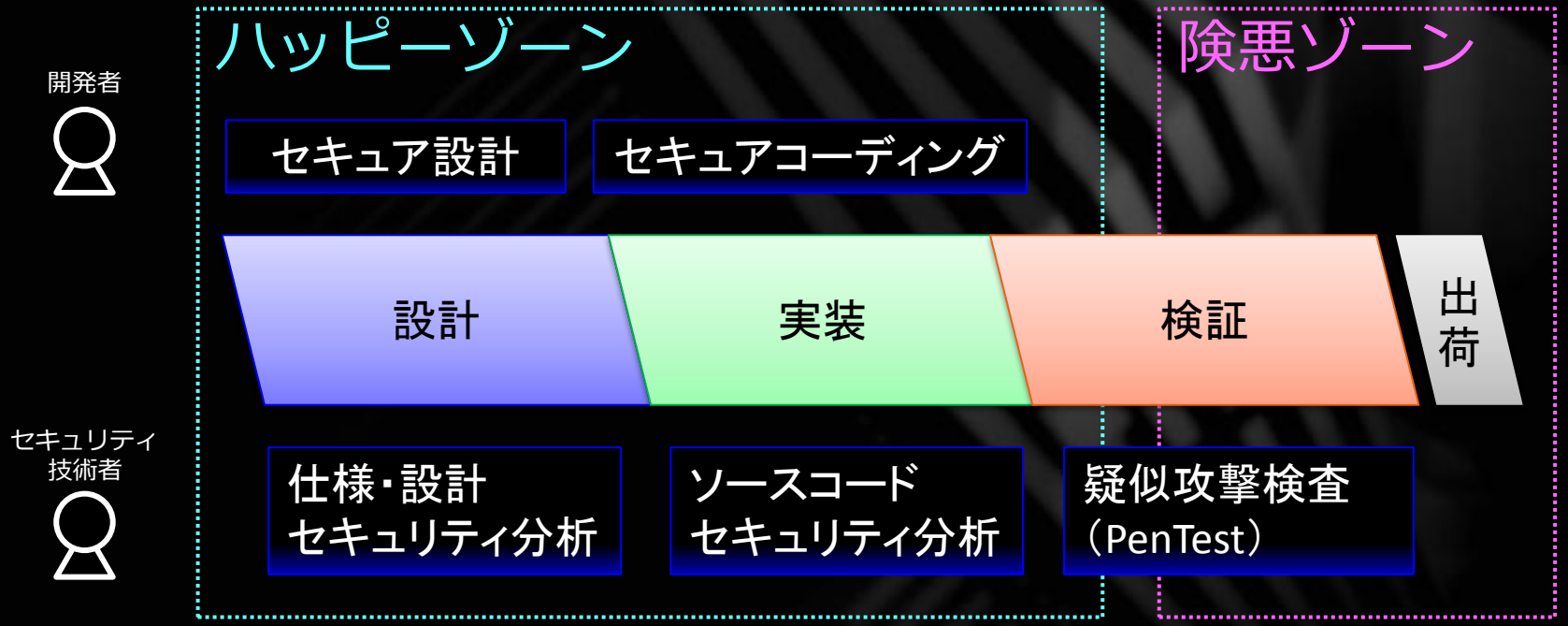
- セキュリティ技術者のジレンマ
  - セキュリティ技術を極めると、開発者からは嫌われる
  - 手抜き検査をすると、開発者から喜ばれる
    （一応専門家に診てもらったとお墨付きがもらえるので）

ごもっともでございます。

もっと早くに
言えよ！

ムﾑ(●`Д')∩キ ――――――

上流でやればハッピー



- いまでは開発者の人たちと仲良くやっています
  - 険悪ゾーンは外部のセキュリティベンダーに委託

開発スピードを落とさないために

- セキュリティ専門組織（SSAG）をつくる
  - セキュア設計・セキュアコーディングを熟知したメンバーで構成
  - 1つのSSAGが複数の開発チームをセキュリティ面でサポート
  - 開発者は Creative な仕事に専念できる

開発チーム

- **製品そのものの開発**
- **セキュアコーディング研修受講**
- **SSAGへの情報提供**
  - 設計情報・ソースコード提供
- **セキュリティ対策の実施**
  - ルールにしたがった開発
  - SSAGの対策提案を検討・実施

連携

SSAG

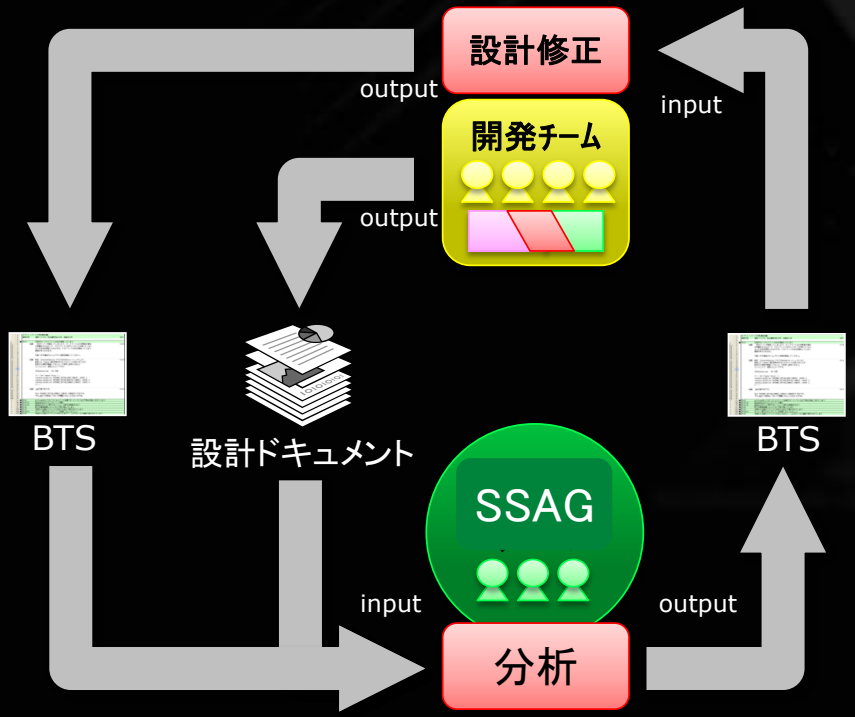
- **ルールの策定と周知**
  - セキュア設計・セキュアコーディングガイド
- **セキュリティ研修の開催**
- **セキュリティサービス提供**
  - 仕様・設計セキュリティ分析
  - ソースコードセキュリティ分析
  - 開発チームに対策提案

セキュアコーディング研修

- ソースコードを書く開発者は受講必須
- 大量に生成されるソースコードのセキュリティ品質を高めるための重要な教育
- 開発者がSSAGの指摘を理解できるためにも重要

仕様・設計セキュリティ分析

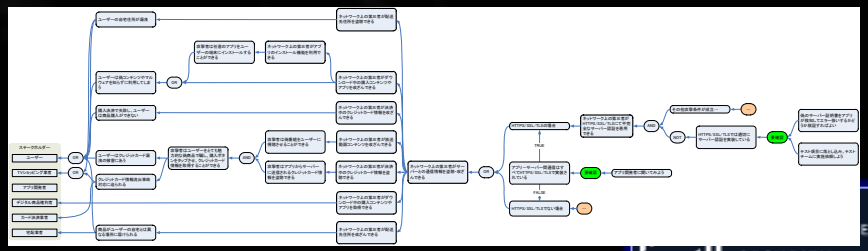
- 設計文書をSSAGが分析し、リスクを見える化
- リスクおよび対策提案をBTS経由で開発者に提示し、設計変更依頼
- 設計変更方法等の問い合わせにSSAGが対応、開発者を支援
- 一部の設計をSSAGが実施することもある



セキュリティ分析表

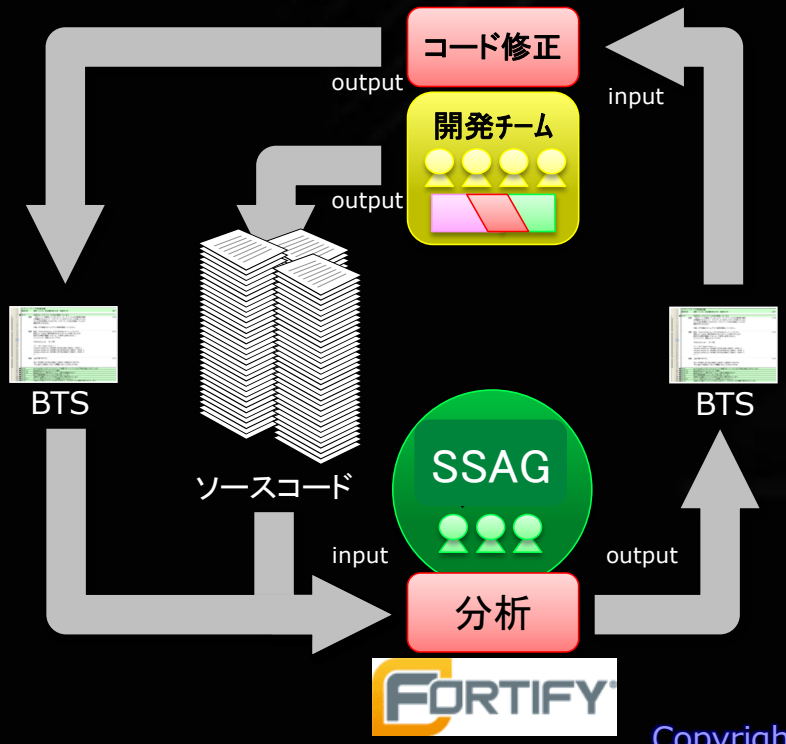
分析対象において、セキュリティ視点で気になった箇所の記録		リスク関連情報の記録		
分析対象	発見箇所	分析対象からコピペ（あとで検索できるように）	関連情報タイプ	関連情報の内容	ID
アプリ要件定義書	p.1	インターネット版TVショッピングアプリ	情報資産	サーバーとの通信情報　→ 情報資産の内容そのもの、情報	AS-1
				通信するから情報資産の送受があるだろう。実際にはこの後に出てくる情報資産が送受信されることになる	
			脆弱性	HTTPS/SSL/TLSにて不完全なサーバー認証	AS-2
				サーバーとの通信はHTTPS/SSL/TLSを使っているかどうかわからないが、使っていた場合にはサーバー認証不備の脆弱性について注意が必要	
				過去にCopyleでサーバー証明書の検証が不適切でMITMされても警告やエラーが表示されないという脆弱性があった	
			要確認	通信経路の暗号化やサーバー認証はどうなっているのか？	
			ステークホルダー	ユーザー	
			ステークホルダー	TVショッピング業者	
			ステークホルダー	アプリ開発者	
			攻撃者	ネットワーク上の第三者	
			脅威	盗聴	
			脅威	サーバーになりすまし	
		TVショッピング番組放送をリアルタイム視聴	機能資産	TVショッピング番組の動画視聴機能	AS-3
			情報資産	放送動画コンテンツ	AS-4
				動画が動画共有サイトで放送されたりした場合、TVショッピング業者は困るの？むしろ宣伝になって好都合？	
			攻撃者	悪意あるユーザー	
			ステークホルダー	TVショッピング業者	
		商品をワンタップで購入	機能資産	商品購入の電子決済機能	AS-5
				確認画面を挟まないでよいのか？「振って購入しちゃった。買うつもりはなかった。」とユーザーが主張した場合、返品対応が必要になるのではないか？	
			脅威	ユーザーから「購入ボタンを振って触ってしまって購入扱いになった。買うつもりはなかった。キャンセルして欲しい。」という問い合わせが発生する	RD-1
			要確認	ワンタップ購入では、確認画面を挟まなくて大丈夫？	
		事前に登録したクレジットカード情報	情報資産	クレジットカード情報	AS-6
			要確認	クレジットカード情報はどこに保存？端末？サーバー？	

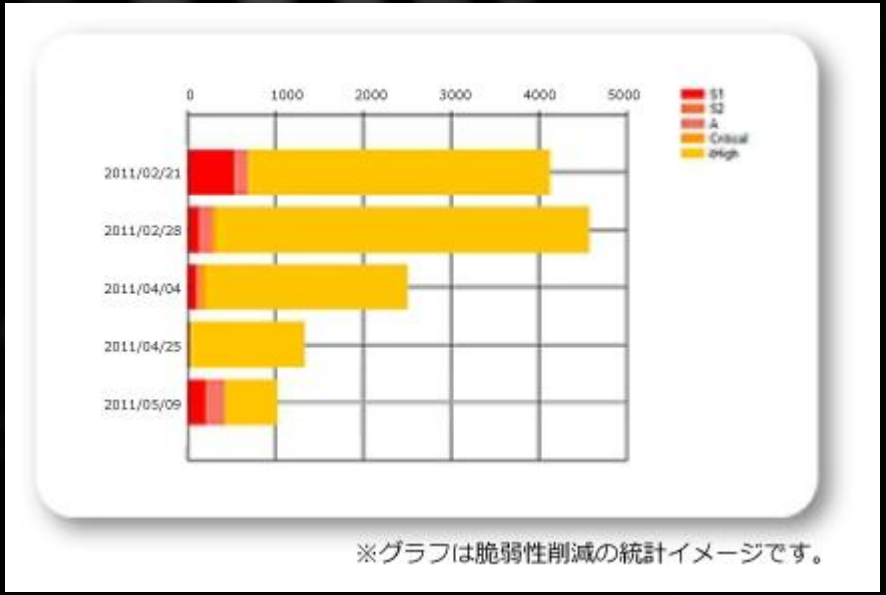
セキュリティ分析図



ソースコードセキュリティ分析

- 脆弱性分析ツールFortifyでソースコードを毎週スキャンし、スキャン結果をWebで時系列に統計表示
- スキャン結果をSSAGが精査し、BTSで開発者に修正依頼
- 修正方法等の問い合わせにSSAGが対応、開発者を支援





宣伝

- Androidセキュリティでお困りであれば、是非お声掛けください
  - 弊社は前述のセキュア開発サービスだけでなく、コンサルティングサービスも提供しています
  - お試し1時間無料コンサルティングもやっています
  - Androidに限らず、ソフトウェアセキュリティ全般もカバーしています
- 問い合わせ先

ソニーデジタルネットワークアプリケーションズ株式会社
セキュリティビジネスグループ
松並　勝　<Masaru.Matsunami@jp.sony.com>
http://www.sonydna.com/

まとめ

TFメンバー募集！

Tips募集！

連絡先： JSSEC事務局 <sec@jssec.org>
Subject: セキュアコーディングTF 参加申込み