明日、敗訴しないための
のセキュアコーディング

@ken5scal

About Me

- 名前
  - 鈴木研吾
  - @kengoscal(Twitter), ken5scal(Github)
- 職歴
  - セキュリティ系(2011年11月〜)
  - Money Forward所属（2014年11月〜）

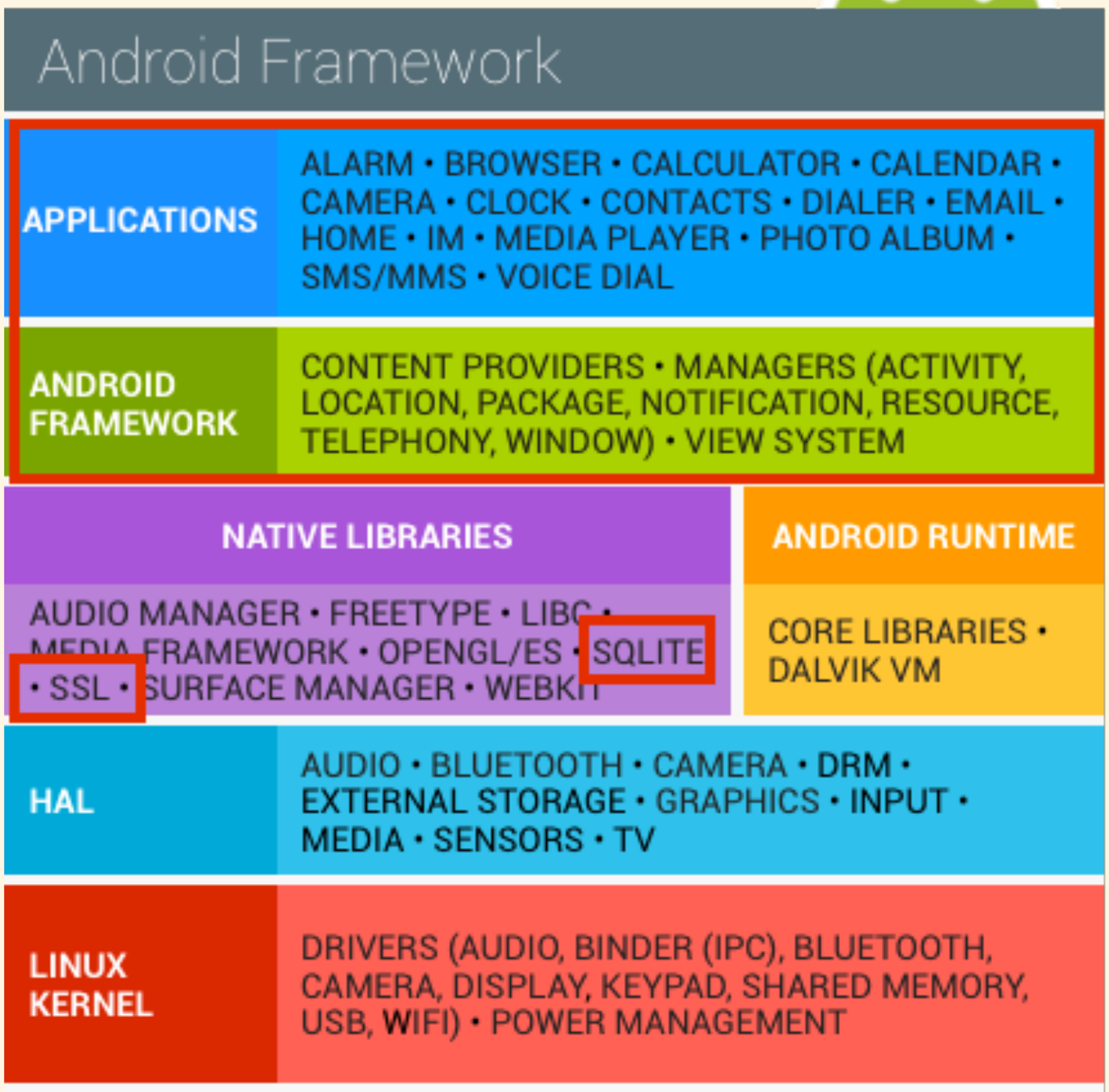
私とAndroid

- 2015年: 1月からAndroid開発に参加
- Marshmallowから
- 先輩開発者が経験した辛さを伝説としてのみ知る

今回話すこと

- Androidセキュアコーディング(以下、SC)について
- 開発の各段階でどんなSCが必要になるのか
- Application, Android Framework層をメインに話そうと思います

今回話すこと


Android Framework
APPLICATIONS
ALARM • BROWSER • CALCULATOR • CALENDAR • CAMERA • CLOCK • CONTACTS • DIALER • EMAIL • HOME • IM • MEDIA PLAYER • PHOTO ALBUM • SMS/MMS • VOICE DIAL
ANDROID FRAMEWORK
CONTENT PROVIDERS • MANAGERS (ACTIVITY, LOCATION, PACKAGE, NOTIFICATION, RESOURCE, TELEPHONY, WINDOW) • VIEW SYSTEM
NATIVE LIBRARIES
ANDROID RUNTIME
AUDIO MANAGER • FREETYPE • LIBC • MEDIA FRAMEWORK • OPENGL/ES • SQLITE • SSL • SURFACE MANAGER • WEBKIT
CORE LIBRARIES • DALVIK VM
HAL
AUDIO • BLUETOOTH • CAMERA • DRM • EXTERNAL STORAGE • GRAPHICS • INPUT • MEDIA • SENSORS • TV
LINUX KERNEL
DRIVERS (AUDIO, BINDER (IPC), BLUETOOTH, CAMERA, DISPLAY, KEYPAD, SHARED MEMORY, USB, WIFI) • POWER MANAGEMENT

アウトライン

- **背景**
- Androidにおけるセキュアコーディングは必要？
- どこから始める？
- 設計フェーズ、実装フェーズ、運用フェーズ
- まとめ

背景

- 登場人物
  - A社 -> 発注側。Eコマースの受注システムを設計〜保守込みで契約
  - B社 -> 開発側（受注側)
- 受注システムを利用したユーザーのクレカ情報が流出
  - 原因はコードレベルでのセキュリティ対策不足
- A社はB社を”**債務不履行**"と損害賠償で民事訴訟
  - ただし契約には"**本件ウェブサイトのセキュリティ対策を講じる義務を負うことは規定されていなかった**"

[引用]http://www.softic.or.jp/semi/2014/5_141113/op.pdf

結果

＿人人人人人＿

＞　B社敗訴　＜

￣Y^Y^Y^Y^Y￣

判決

- "被告は，原告に対し，２２６２万３６９７円及びこれに対する平成２３年１０月１５日から支払済みまで年６分の割合による金員を支払え。"

- IPA/経産省が**特定の攻撃に対する対策**を推奨していため、”**被告には重過失が認められるというべき**”とされた。

[引用]http://www.softic.or.jp/semi/2014/5_141113/op.pdf

要は

- 代表的な機関が何らかの警告を出したら、その対策をすることが当然と認められる前例ができた。

ではAndroidでは？

アウトライン

- 背景
- **Androidにおけるセキュアコーディングは必要？**
- どこから始める？
- 設計フェーズ、実装フェーズ、運用フェーズ
- まとめ

Androidの場合は？

- Android脆弱性の学習・点検ツールが提供されてる

ちなみに・・・

動作対象OS	Microsoft Windows Vista (32bit/64bit版) Microsoft Windows 7 (32bit/64bit版) Microsoft Windows 8 (32bit/64bit版) Microsoft Windows 8.1 (32bit/64bit版)
ハードウェアスペック	「OSの動作に支障がなく、Eclipseを使用したAndroidアプリのビルドが　支障なく行えること」 「メモリ 4GB以上を推奨」
統合開発環境 Eclipse	「Eclipse Foundationサイト配布版 Juno Packages v 4.2.0」 以降 または 「Android Developersサイト配布版 ADT Bundle for Windows 」
Java Development Kit	Java SE Development Kit 7 以上 がインストールされていること
Android Development Tools	ADT 22.3 以上がインストールされていること
Android Software Development Kit	Android API 8 、Android API 10 SDK がインストールされていること

ちなみに・・・

動作対象OS	Microsoft Windows Vista (32bit/64bit版) Microsoft Windows 7 (32bit/64bit版) Microsoft Windows 8 (32bit/64bit版) Microsoft Windows 8.1 (32bit/64bit版)
ハードウェアスペック	「OSの動作に支障がなく、Eclipseを使用したAndroidアプリのビルドが　支障なく行えること」 「メモリ 4GB以上を推奨」
統合開発環境 Eclipse	「Eclipse Foundationサイト配布版 Juno Packages v 4.2.0」 以降 または 「Android Developersサイト配布版 ADT Bundle for Windows 」
Java Development Kit	Java SE Development Kit 7 以上 がインストールされていること
Android Development Tools	ADT 22.3 以上がインストールされていること
Android Software Development Kit	Android API 8 、Android API 10 SDK がインストールされていること

ちなみに・・・

動作対象OS	Microsoft Windows Vista (32bit/64bit版) Microsoft Windows 7 (32bit/64bit版) Microsoft Windows 8 (32bit/64bit版) Microsoft Windows 8.1 (32bit/64bit版)
ハードウェアスペック	「OSの動作に支障がなく、Eclipseを使用し く行えること」 「メモリ 4GB以上を推奨」
統合開発環境 Eclipse	「Eclipse Foundationサイト配布版 Juno P または 「Android Developersサイト配布版 ADT E
Java Development Kit	Java SE Development Kit 7 以上 がインス

ちなみに・・・

動作対象OS	Microsoft Windows Vista (32bit/64bit版) Microsoft Windows 7 (32bit/64bit版) Microsoft Windows 8 (32bit/64bit版) Microsoft Windows 8.1 (32bit/64bit版)
ハードウェアスペック	
統合開発環境 Eclipse	
Java Development Kit	

微妙なライン

- これを対策の推奨と取られてしまうか？？
- のであれば、訴訟リスクのためセキュリティ対策をAndroidでもしなければいけないかも

訴訟リスクだけではない

セキュリティ対策は誰のため？

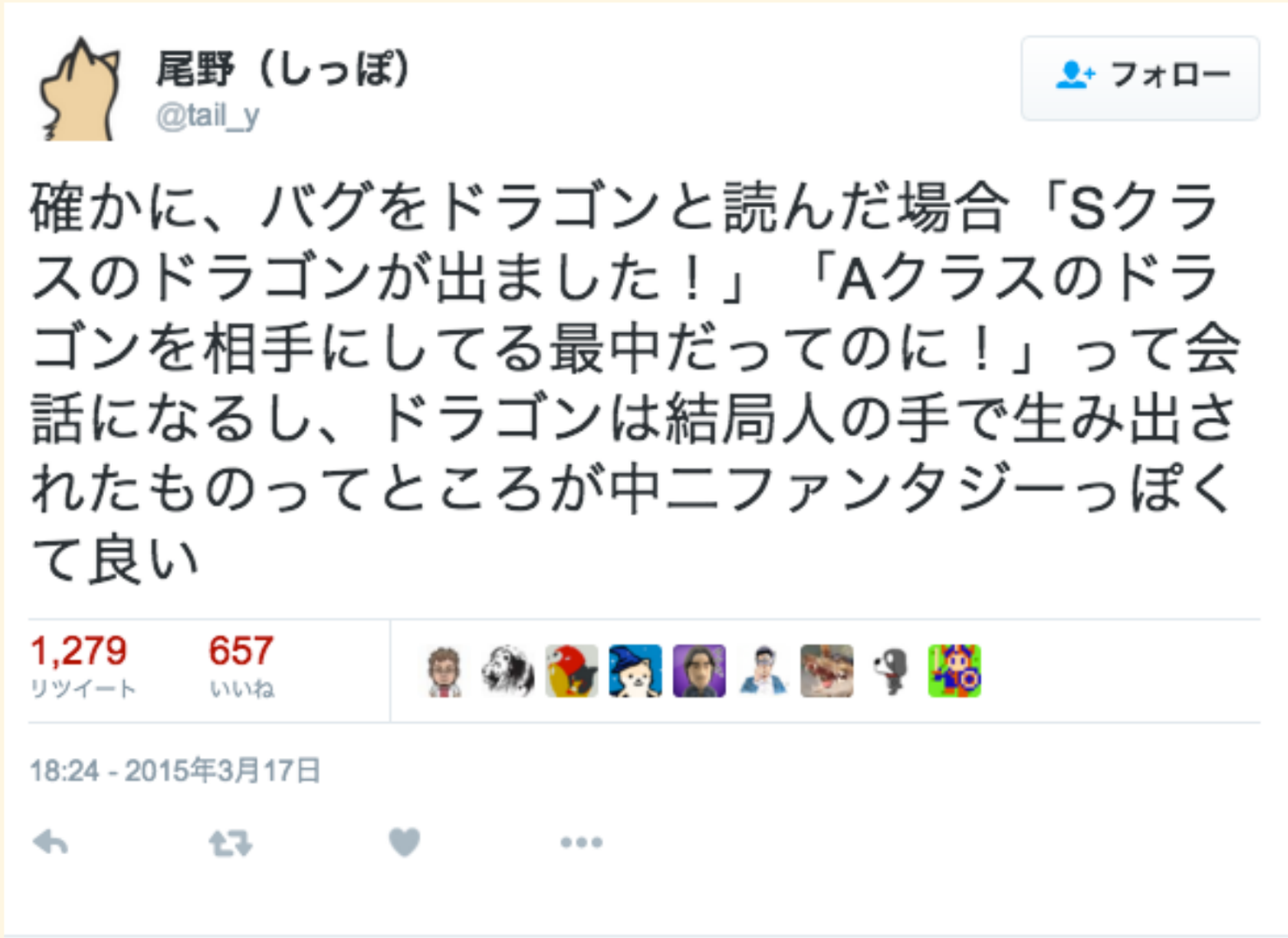
サービス利用者のためでもある

- そもそも脆弱性(攻撃対象のウィークポイント)は**バグ**である
- => バグ対応・対策は品質向上である
- => 品質向上はユーザー体験を向上・保証する
- セキュリティ対策をしたところで、ユーザー体験に直結しないかもしれない
- だが、セキュリティ未対策が体験を著しく損ねることは多いにありえる
- やるやらは体験を損ねた時のリスクと影響具合から判断すべき

なので

- SCは必要
- ただ必要だからやるのではない
- ユーザー体験のためにSCすべき

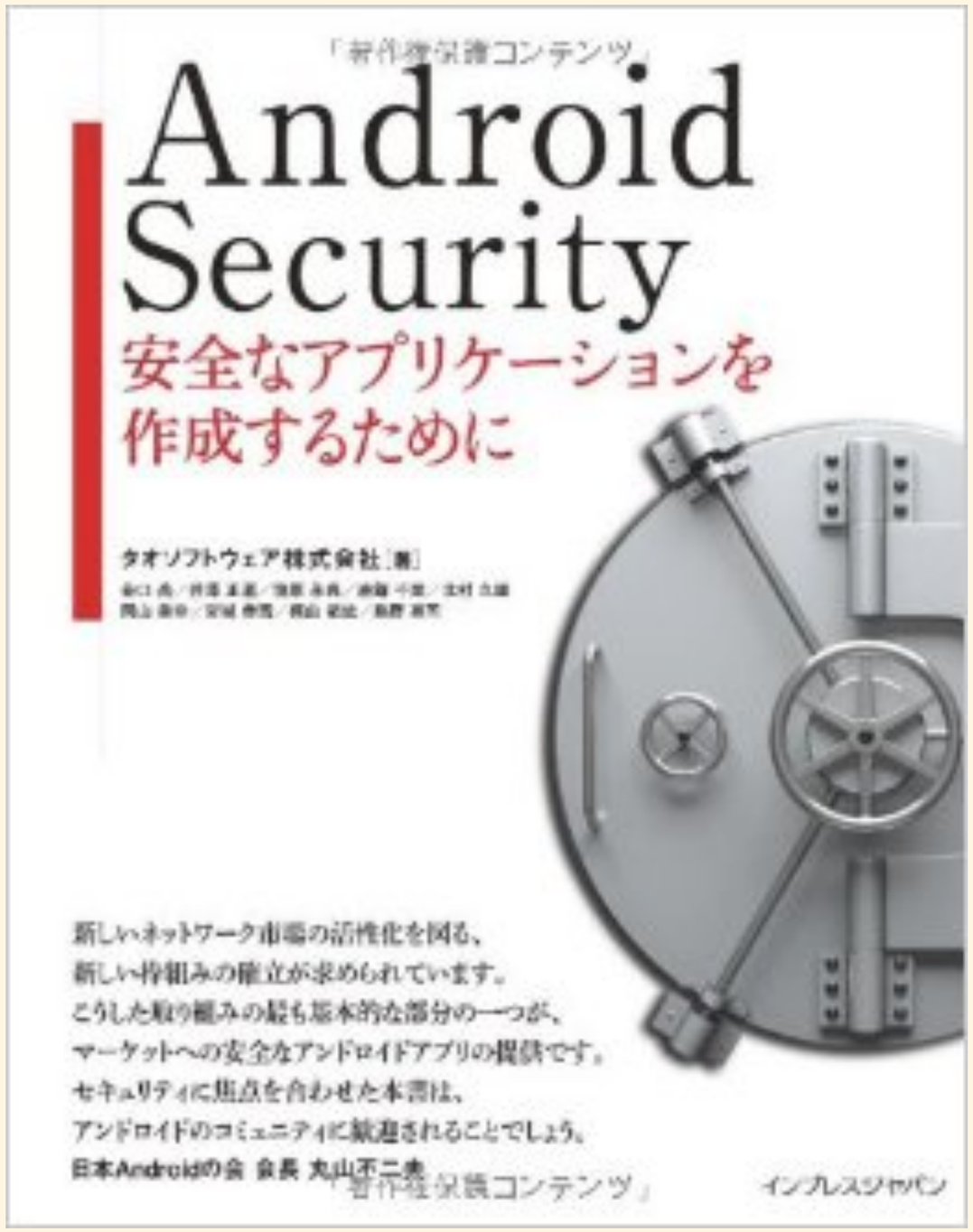
バグを潰してレベルを上げる



アウトライン

- 背景
- Androidにおけるセキュアコーディングは必要？
- **どこから始める？**
- 設計フェーズ、実装フェーズ、運用フェーズ
- まとめ

良書 - その1



- 包括的な知識を身につけられる
- 殺られた系の事例が紹介されてる

良書 - その2

- 最強
- Marshmallowにも対応
- サンプルコードも抱負
- こいつに沿えばほぼ間違いない

完

完

・もうちょっとだけ続く

課題

- **とにかく範囲が広すぎる！**
- これを全て理解し、適切な機能に、正しく実装する事は難しい
- Android SCの範囲
  - ファイルアクセス権限、APKファイル保護、パーミッション、Activity, Broadcast, Service, Content Provider、インテント、暗号化、SQLite、Logcat, WebView, Account Manger, https, プライバシー情報、パスワード入力画面, アプリケーション保護, 証明書, 新しいPermission, Clipboard, etc, etc

なので

- 開発しようとしている機能・サービスにおいて
- 開発における各段階に応じて
- どのSCが必要か・不要か
- QualityとSpeedのバランスを見極めて
- 対応することが大事

Build Security Inの紹介

- 最近セキュリティ業界で盛り上がってる。
- https://buildsecurityin.us-cert.gov

Build Security Inの紹介

- 最近セキュリティ業界で盛り上がってる

- “Build Security In is a collaborative effort that provides practices, tools, guidelines, rules, principles, and other resources that software developers, architects, and security practitioners can use to build security into software in every phase of its development.”

・ by US-CERT(米国国土安全保障省（DHS）配下の情報セキュリティ対策組織)

[引用]http://www.softic.or.jp/semi/2014/5_141113/op.pdf

Build Security Inの紹介

- 最近セキュリティ業界で盛り上がってる

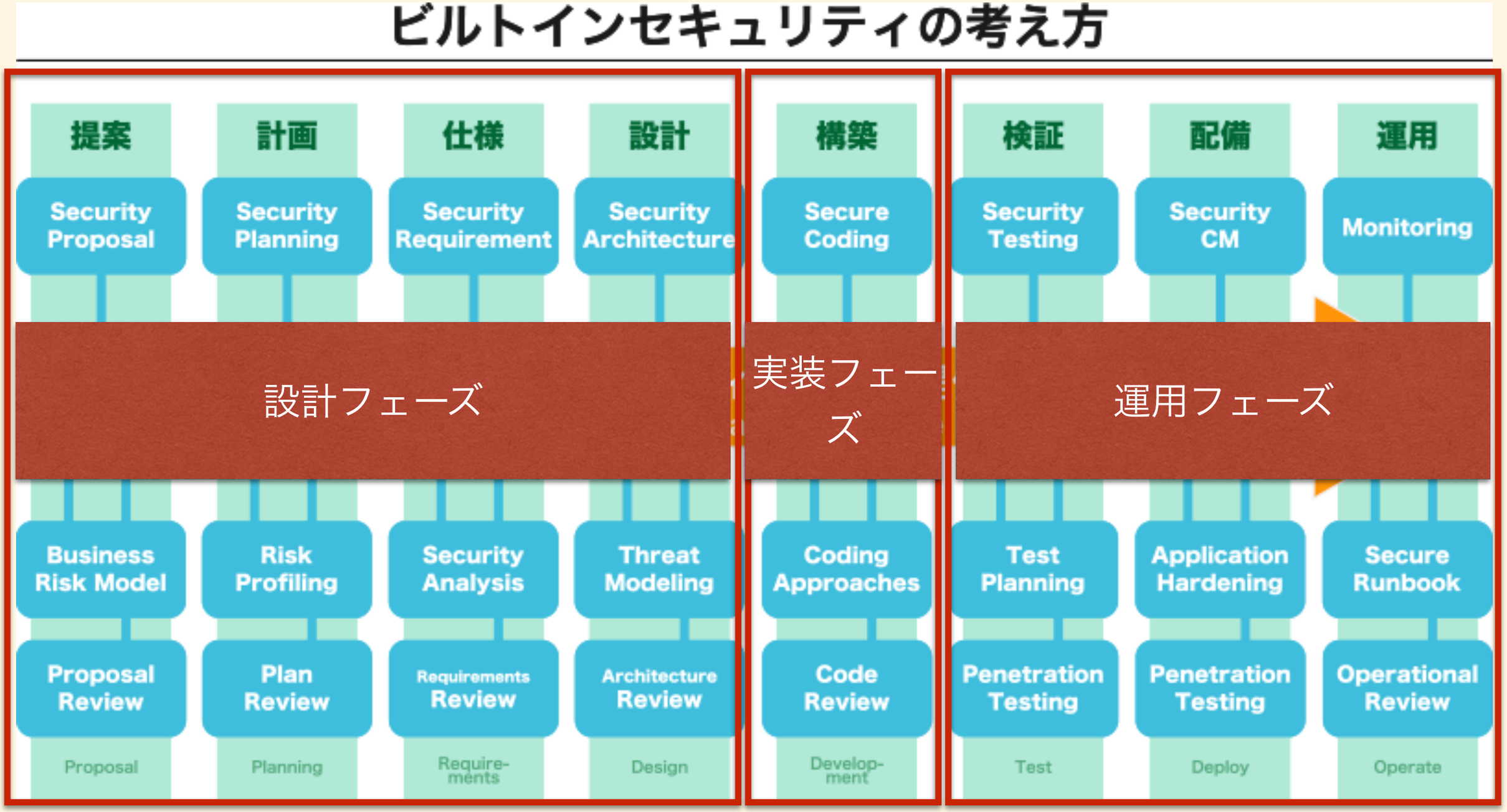
- “ソフトウェア開発の全工程にセキュリティを組み込めるようなベスト・プラクティス、ツール、ガイドライン等のリソースを開発者などに提供する協同的な取り組み”

[引用]http://www.softic.or.jp/semi/2014/5_141113/op.pdf

どういうことか、と言うと

[引用: 株式会社セタ・インターナショナル] http://www.seta-international.co.jp/vietnam_offshore/securityoption.html

今回はこんな感じで区切る



[引用: 株式会社セタ・インターナショナル] http://www.seta-international.co.jp/vietnam_offshore/securityoption.html

アウトライン

- 背景
- Androidにおけるセキュアコーディングは必要？
- どこから始める？
- **設計フェーズ**、実装フェーズ、運用フェーズ
- まとめ

設計フェーズのSCでやること

- **SCそのものではない**
- 機能・サービスリスクに対する脅威と脆弱性の把握
- リスクの顕在化likelihoodや影響レベルから、やるやら判断
- 必要なSCの洗い出し
- 開発計画への盛り込み

- 一般的な脆弱性の発生原因となるバグが設計・開発工程で発生する確率は**80%**

  - Kevin Soo Hoo, “Tangible ROI through Secure Software Engineering.”Security Business Quarterly,. Vol.1, No.2, Fourth Quarter, 2001.参考

Androidでいうと

- ココらへんについて特に注意を払いたい
  - Permission
  - 3rd Party製の広告モジュール
  - データの保管方法
  - WebViewで何処で使って、何をしたいか
  - 連携アプリの有無（コンポーネント間の連携）
- 手戻り工数が大きそう・設計レベルでやり直さないといけないもの等

Case: Money Forward

- 

のB2B版を開発中
- サービス紹介: 金融機関に資産情報をスクレイピングにいって、それを一元化するアプリ

例えばこういう仕様

- 連携可能な1000行以上の金融機関のログイン情報等をもらう必要あり。

- 自社の他アプリとの連携あり

- デフォルトでパスコードロックをオンに。できれば指紋認証も。

こういう事を考える

- Permission
  - ネットワーク通信、ユーザーアカウント、指紋認証ぐらいしか使わない

```
<uses-permission
android:name="android.permission.INTERNET" />
<uses-permission
android:name="android.permission.USE_FINGERPRINT
```

こういう事を考える

- データの保管方法
  - 金融機関のログイン情報はどこにも保存しない
  - 連携可能な金融機関の一覧は大きいけど、秘密にしたい情報ではないので、SQLiteOpenHelperでDBに保存
- パスコードは暗号化しておきたいのでFileに書き出す
  - AES/CBC/pkcs5padding

こういう事を考える

- WebViewで何処で使って、何をしたいか
  - FAQと利用規約を表示させるくらい
  - WebViewのJavaScriptInterfaceを明示的に無効化

```
// You don't need JavaScript at all for FAQ
webView.getSettings().setJavaScriptEnabled(false);
```

- もしくはChrome Custom Tabを使う
  - Chrom Custom Tabはそもそも
    addJavaScriptInterface的なことができないので

こういう事を考える

- 連携アプリの有無（コンポーネント間の連携）
  - 自社のアプリと連携するための独自Permissionを作る
- 長いので「Androidアプリのセキュア設計・セキュアコーディングガイド」を参照されたし
  - 4.1.1.4. 自社限定 Activity を作る・利用する
  - 5.2.1.2. 独自定義の Signature Permission で自社アプリ連携する

Anti Case: Skype

- ユーザーの個人情報・チャット履歴が流出した
  - 独自生成したディレクトリ・ファイル下に保存した -> AndroidOSによる保護機能がきかなかった
  - DBファイルも平文で保存されていた
- Build in Securityの「設計フェーズ」において「データの保管方法」について設計から考えてれば、防げた（かも）

アウトライン

- 背景
- Androidにおけるセキュアコーディングは必要？
- どこから始める？
- 設計フェーズ、**実装フェーズ**、運用フェーズ
- まとめ

実装フェーズのSCでやること

- あるある系のチェックが主
  - https通信における証明書検証漏れ
  - logcat漏れ
  - WebViewのaddJavascriptInterface
  - ファイル暗号化時のお作法漏れ
- ここは開発者として抑えておいた方がいいとおもう

https通信における証明書検証漏れ

- 2013年に比較して、暗号通信が盗聴・解読されるリスクのあるアプリの割合が悪化

- 「Androidアプリ脆弱性調査レポート2015」 by 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

- 基本的に独自実装しようとしたものが多い

- 独自TrustManager

- 独自HostNameVerifier

- の時のみ、検証しないようにする...など注意を払う

- 今時SSL証明書は安いのだから買っちゃえばと言う気がしなくもない

https通信における証明書検証漏れ

- JSSECの「Android アプリのセキュア設計・セキュアコーディングガイド」

  - 5.4. HTTPSで通信する

- http://www.slideshare.net/jpcert_securecoding/androidssltls

logcat漏れ

- デバッグ時に出してたlogcatの消し忘れ
- Proguard
- 副作用が嫌ならせめてctrl + shift + f

logcat漏れ

- JSSECの「Android アプリのセキュア設計・セキュアコーディングガイド」
  - 4.8. LogCatにログを出力する

WebView

- addJavascriptInterfaceで頑張りすぎない
- WebViewクラスで読み込むURLを制限してない
- インターネットからのリソース取得はhttp(s)プロトコルのみ & 自社ドメインに縛る
- fileプロトコルでリソース取得が必要な場合は対象をattr, resだけに絞る

WebView

- JSSECの「Android アプリのセキュア設計・セキュアコーディングガイド」
  - 4.9. WebView を使う
- http://2012.k-of.jp/sites/all/files/slides/android-securecoding.pdf
- https://ierae.co.jp/uploads/webview.pdf

ファイル暗号化時のお作法

- ファイル暗号化時のAES鍵が平文でソースコードに書いてあったのを見た。たまげた

- 脆弱で非推奨な暗号・アルゴリズムを使わない。DESとか。

ファイル暗号化時のお作法

- JSSECの「Android アプリのセキュア設計・セキュアコーディングガイド」
  - 5.6. 暗号化技術を使う

アウトライン

- 背景
- Androidにおけるセキュアコーディングは必要？
- どこから始める？
- 設計フェーズ、実装フェーズ、**運用フェーズ**
- まとめ

運用フェーズのSCでやること

- keyStoreと秘密鍵の保存を気にする段階
- 盗まれると乗っ取られる
  - <- 同じ見た目のアプリを作って、金融機関ログイン情報の飛ばし先を悪意あるサーバに向けるとか。
- バックアップしておかないと、同じアプリとしてリリースできなくなるリスクもある

SCか?

ぼくのかんがえたさいきょうの運用環境

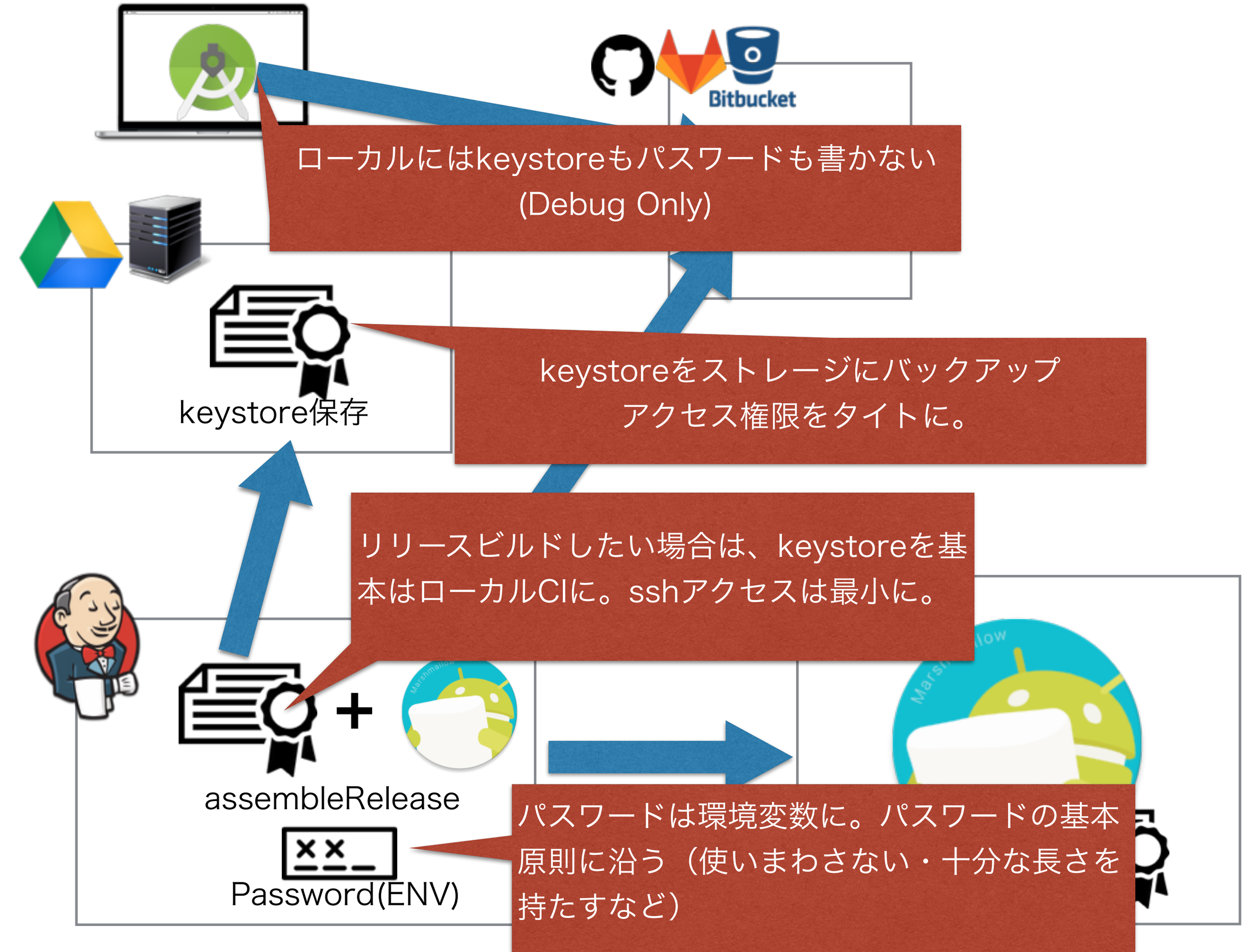
Bitbucket

keystore保存

+

assembleRelease

Password(ENV)


Bitbucket
ローカルにはkeystoreもパスワードも書かない
(Debug Only)
keystore保存
keystoreをストレージにバックアップ
アクセス権限をタイトに。
リリースビルドしたい場合は、keystoreを基本はローカルCIに。sshアクセスは最小に。
assembleRelease
Password(ENV)
パスワードは環境変数に。パスワードの基本原則に沿う（使いまわさない・十分な長さを持たすなど）

アウトライン

- 背景とゴール
- Androidにおけるセキュアコーディングは必要？
- どこから始める?
- 設計フェーズ、実装フェーズ、運用フェーズ
- **まとめ**

まとめ

- セキュリティは品質です
- よってユーザー体験に深く関係します
- セキュアコーディング自体はJSSECの「Android アプリのセキュア設計・セキュアコーディングガイド」を参照すれば間違いない
- サービス・アプリ開発の段階に応じて、どういうセキュリティ対応をしなければ考えるのがコツ（だと思う）

ご清聴

ありがとうございました