ハイブリッドモバイルアプリの セキュリティ

アシアル株式会社
代表取締役 田中正裕

masahiro@asial.co.jp

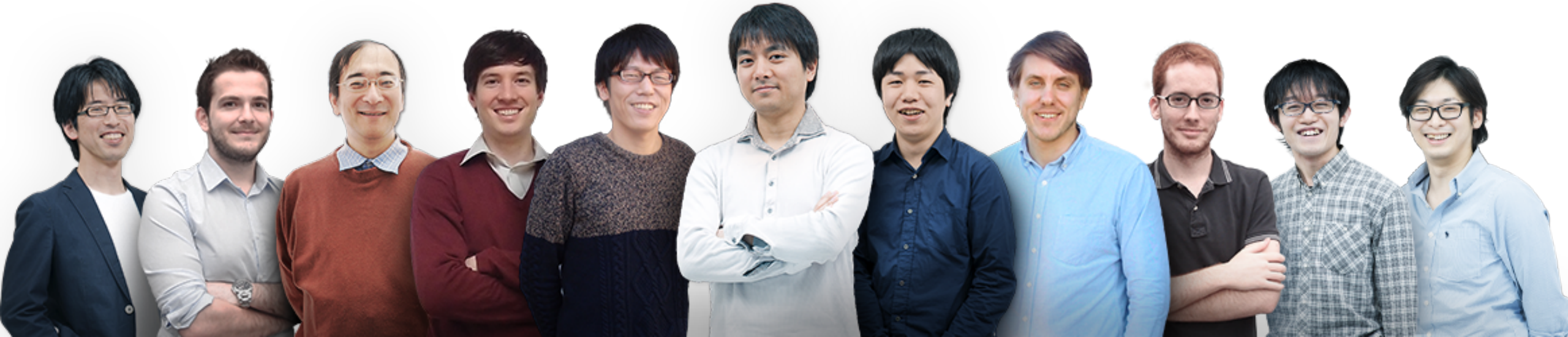
2017年2月8日
JSSEC セキュリティフォーラム 2017

会社概要

- 創業：2002年
- 資本金：1000万円
- 事業所：東京（本社）、サンフランシスコ
- 従業員数：約40名（8国籍）

事業内容

- 開発プラットフォーム事業：開発ツール、UIフレームワーク
- 開発支援事業：アプリ開発、サーバーサイド（PHP）開発
- 教育事業：トレーニング、執筆



HTML5モバイルアプリ開発基盤 Monaca

HTML5、JavaScript
でアプリ開発

セットアップ不要の
クラウド開発環境

UIフレームワーク
Onsen UI搭載

安心の日本語サポート

モバイルデバイスのセキュリティ

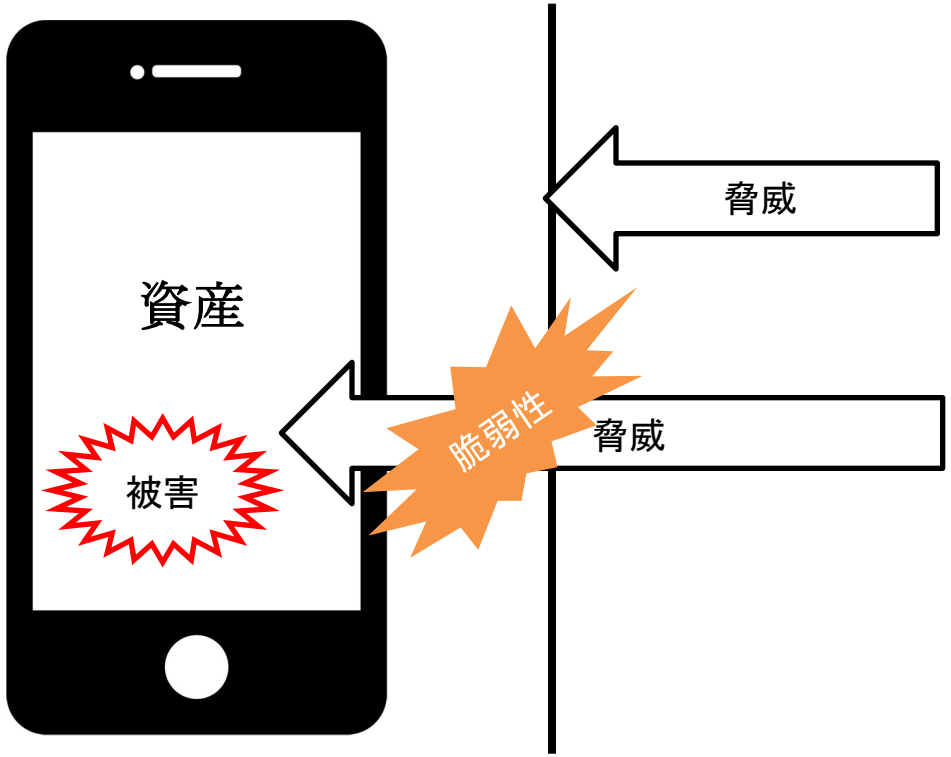
情報資産

- [ ] 電話番号
- [ ] 通話履歴
- [ ] アカウント情報
- [ ] メールアドレス
- [ ] Web閲覧履歴
- [ ] 電話帳

機能資産

- [ ] 電話をかける機能
- [ ] ネットワーク通信機能
- [ ] SDカード読み書き機能
- [ ] SMS送受信機能
- [ ] カメラ撮影機能
- [ ] Bluetooth通信機能
- [ ] GPS機能

資産を守る



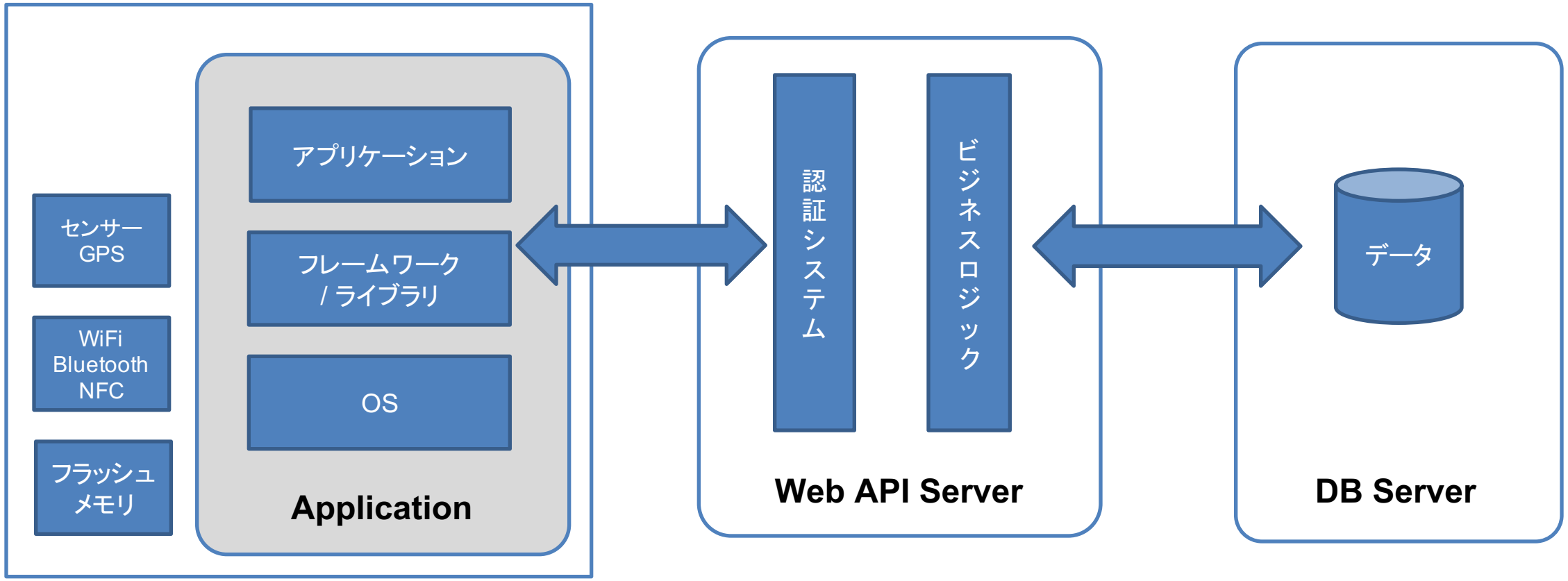
情報資産が脅かされる

- 攻撃者により情報を参照される
- 攻撃者により情報を改ざんされる
- 攻撃者により情報を削除される

機能資産が脅かされる

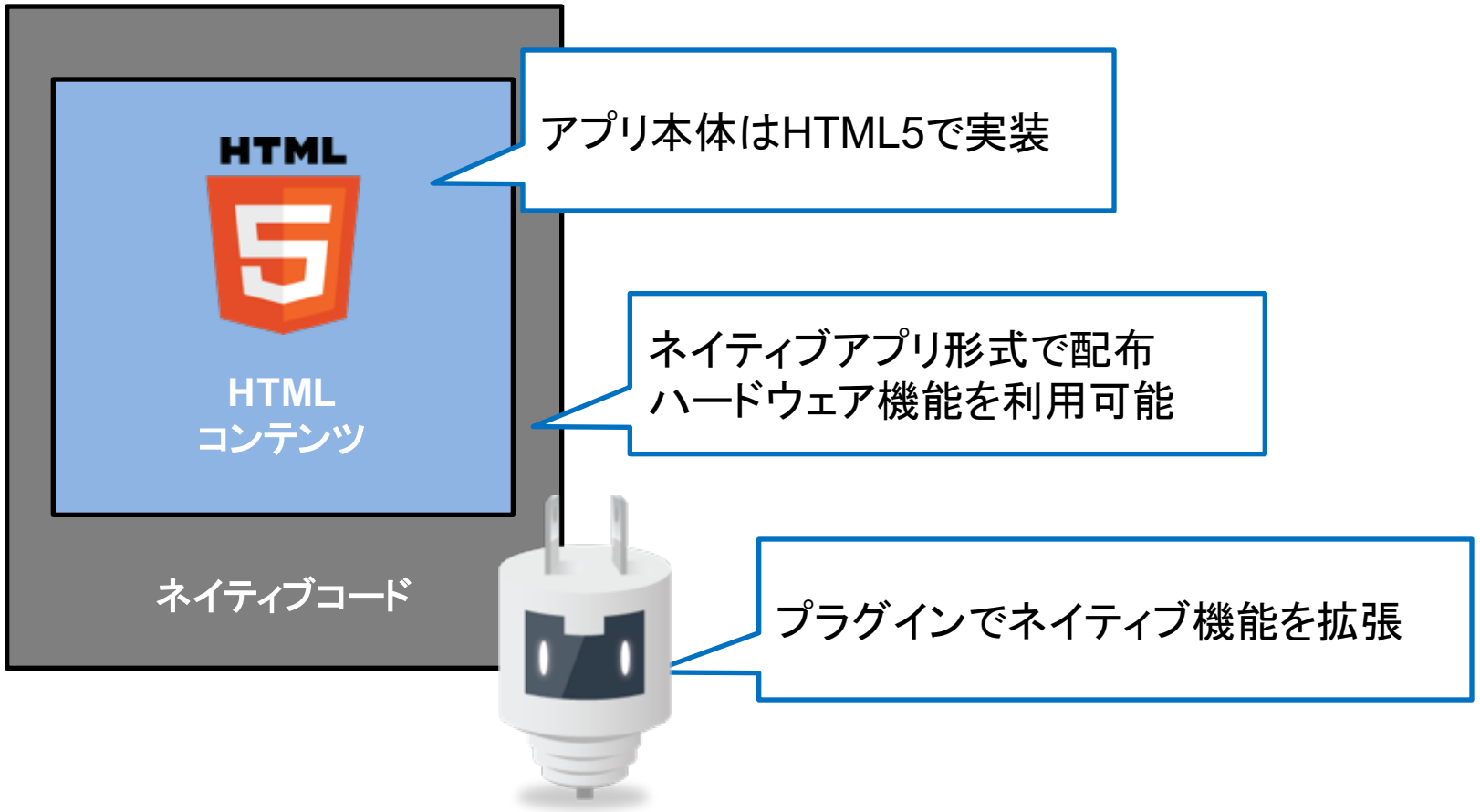
- 攻撃者により機能を使用される

モバイルアプリにおけるAttack Surface

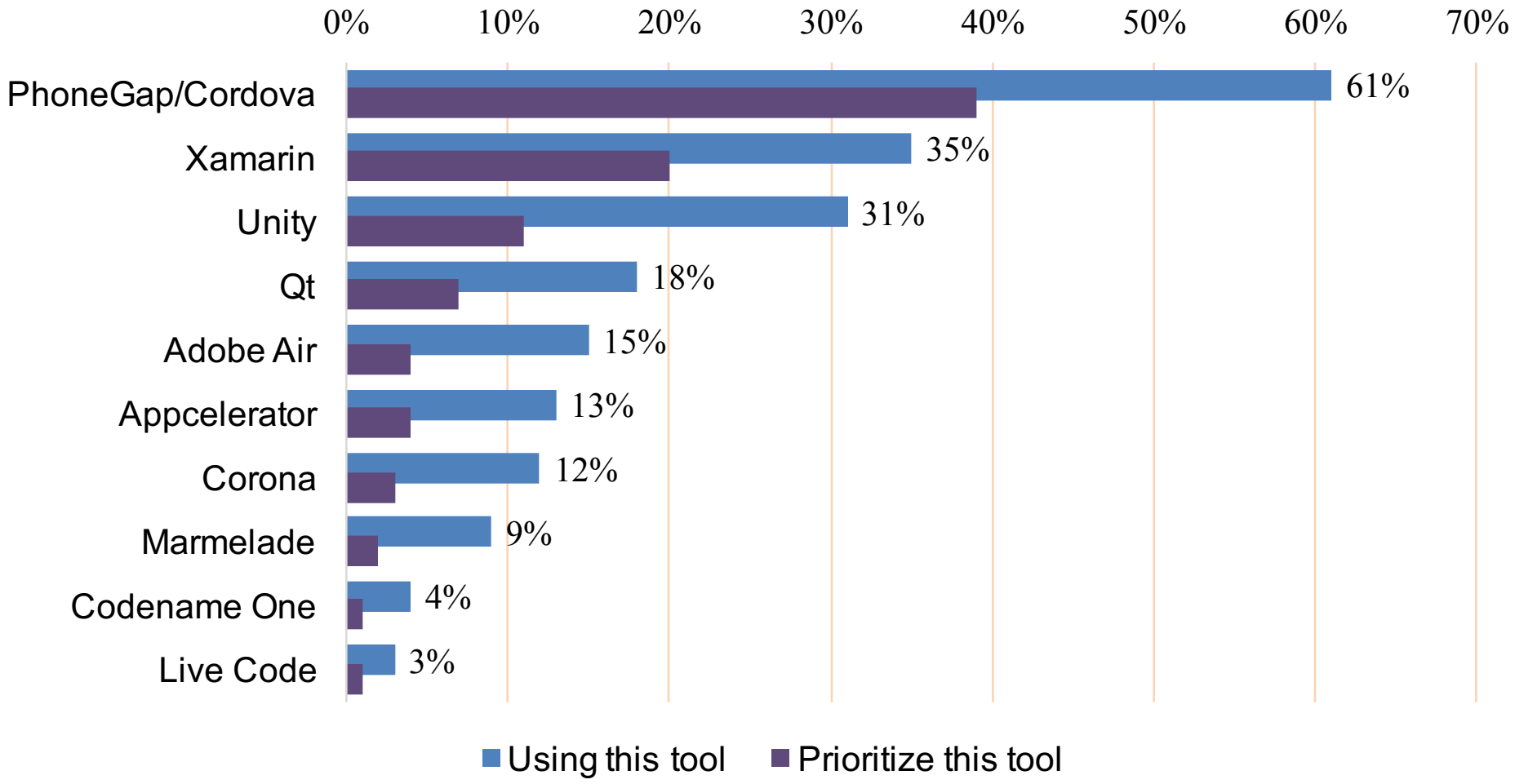


ハイブリッドアプリとは

ハイブリッドアプリ＝HTML5＋ネイティブ

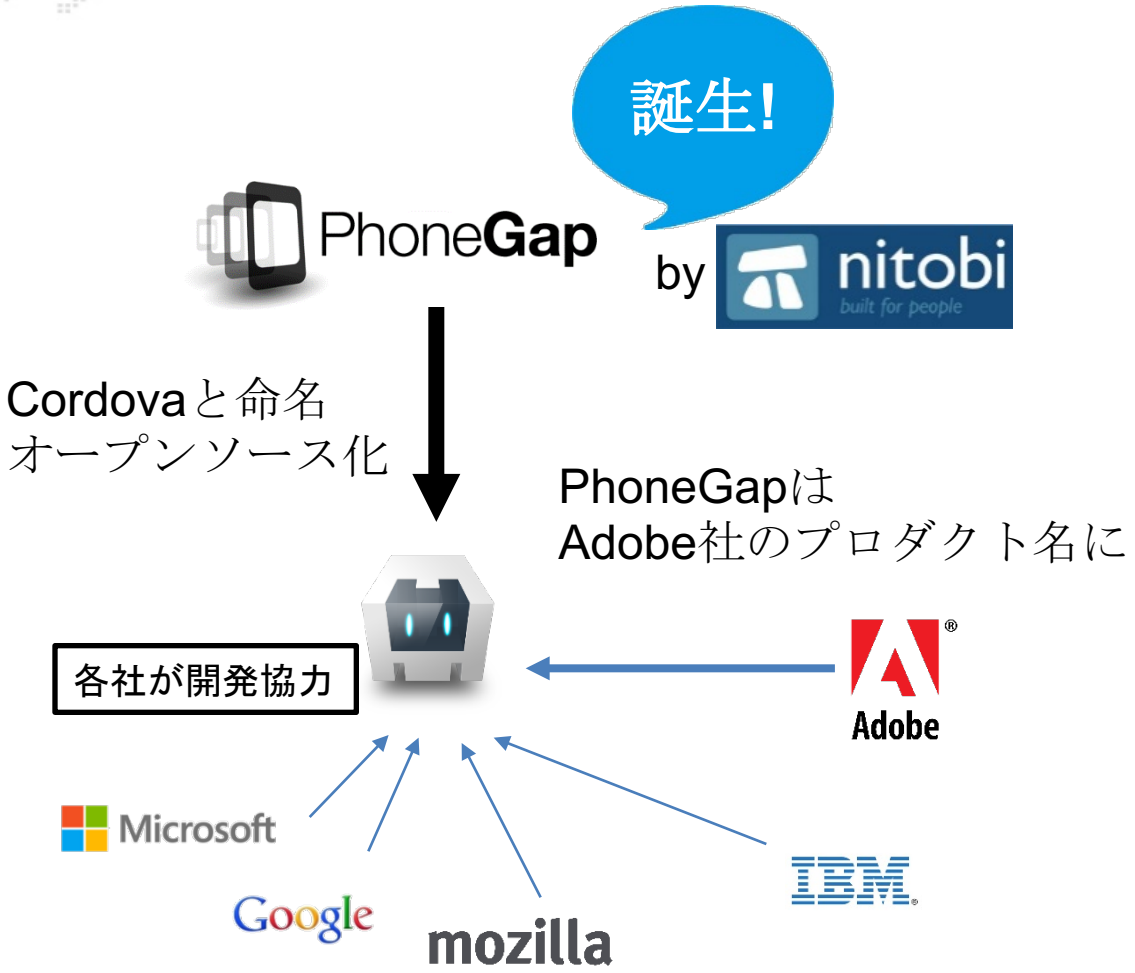


Cross Platformツールのシェア



Vision Mobile Analysis of Cross-Platform Development, July 2015

CordovaとPhoneGap

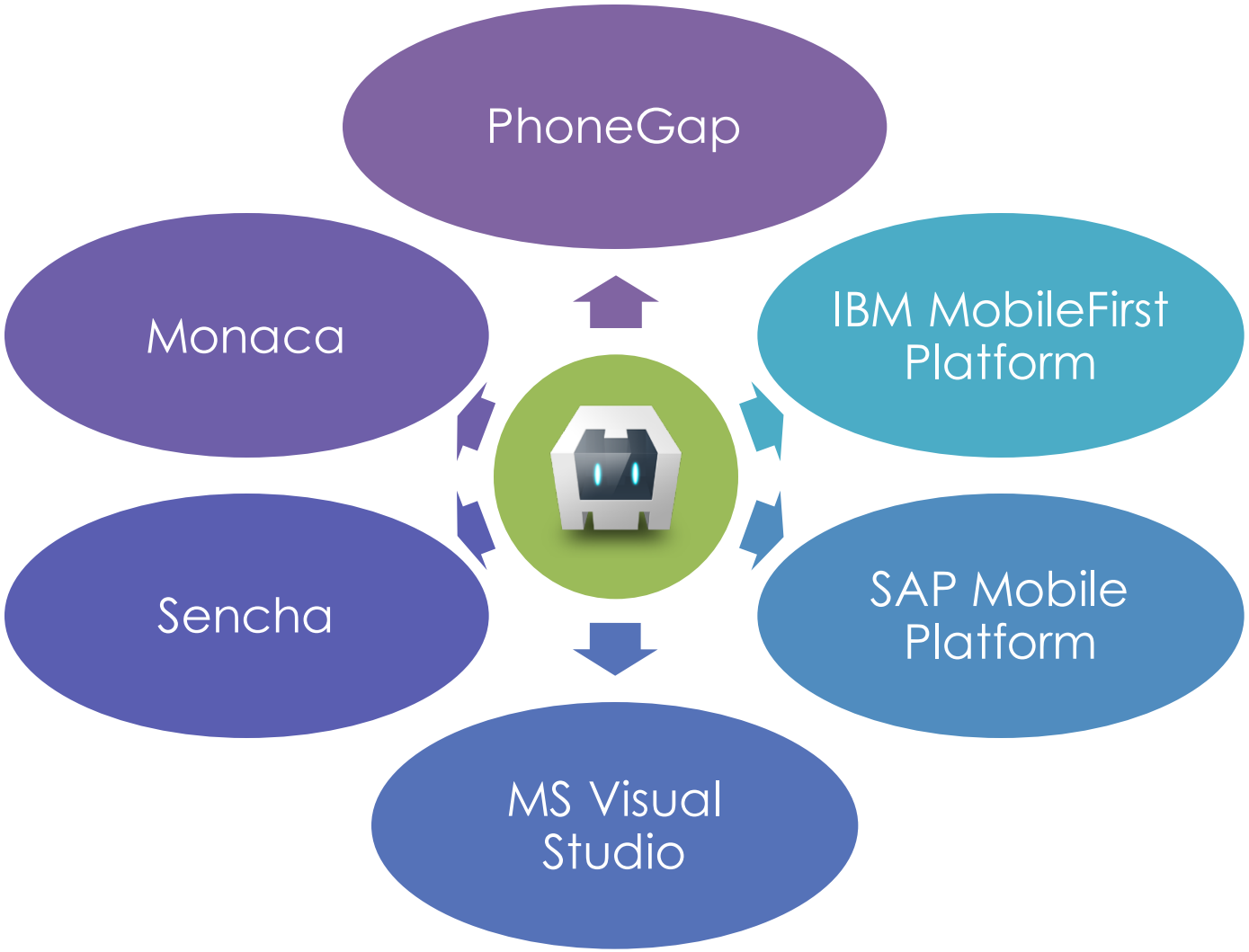


2009年
Nitobi社がPhoneGapを開発
オープンソース製品

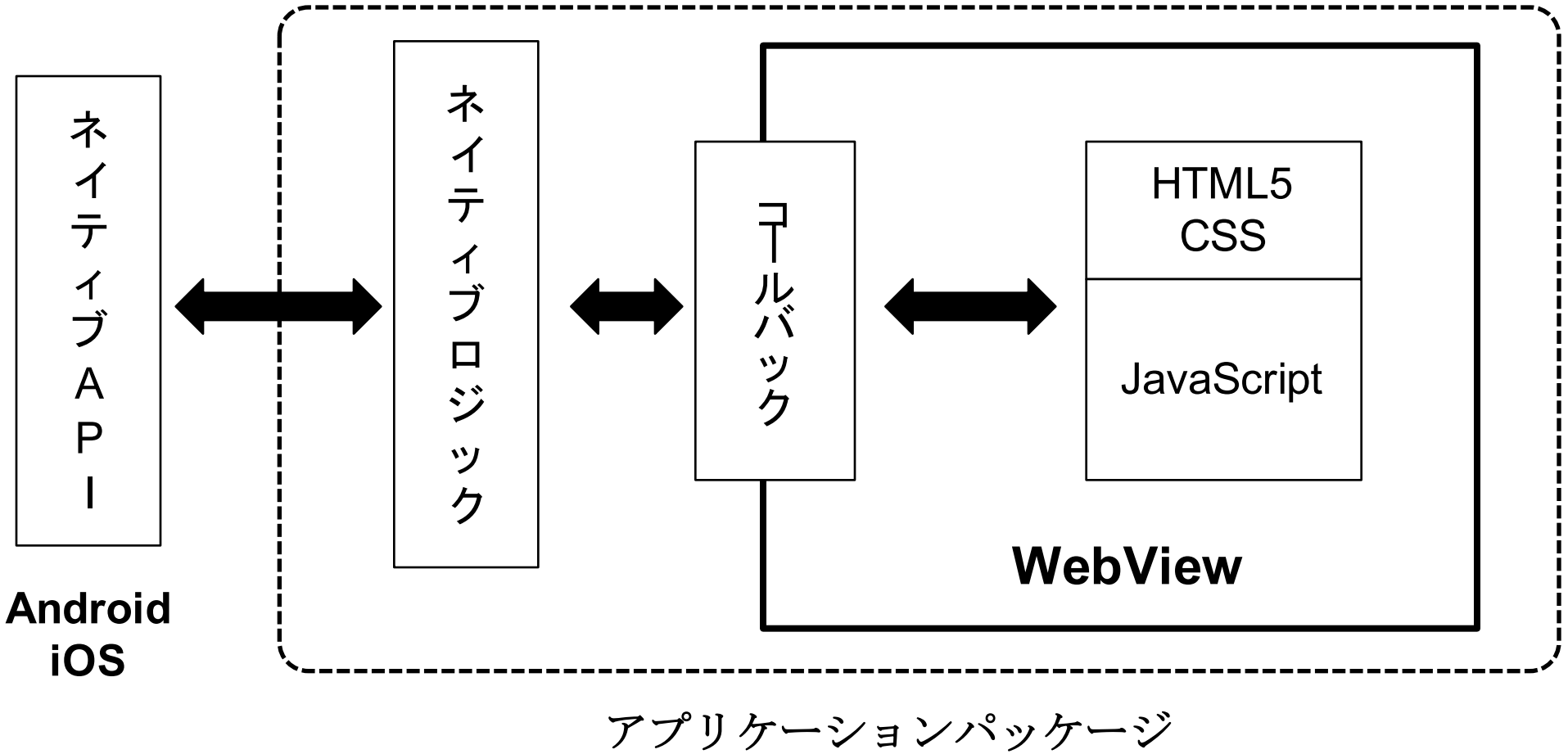
2011年
Adobe社がNitobi社を買収
PhoneGapは「Cordova」に

その結果
多くの企業がCordova開発に参加

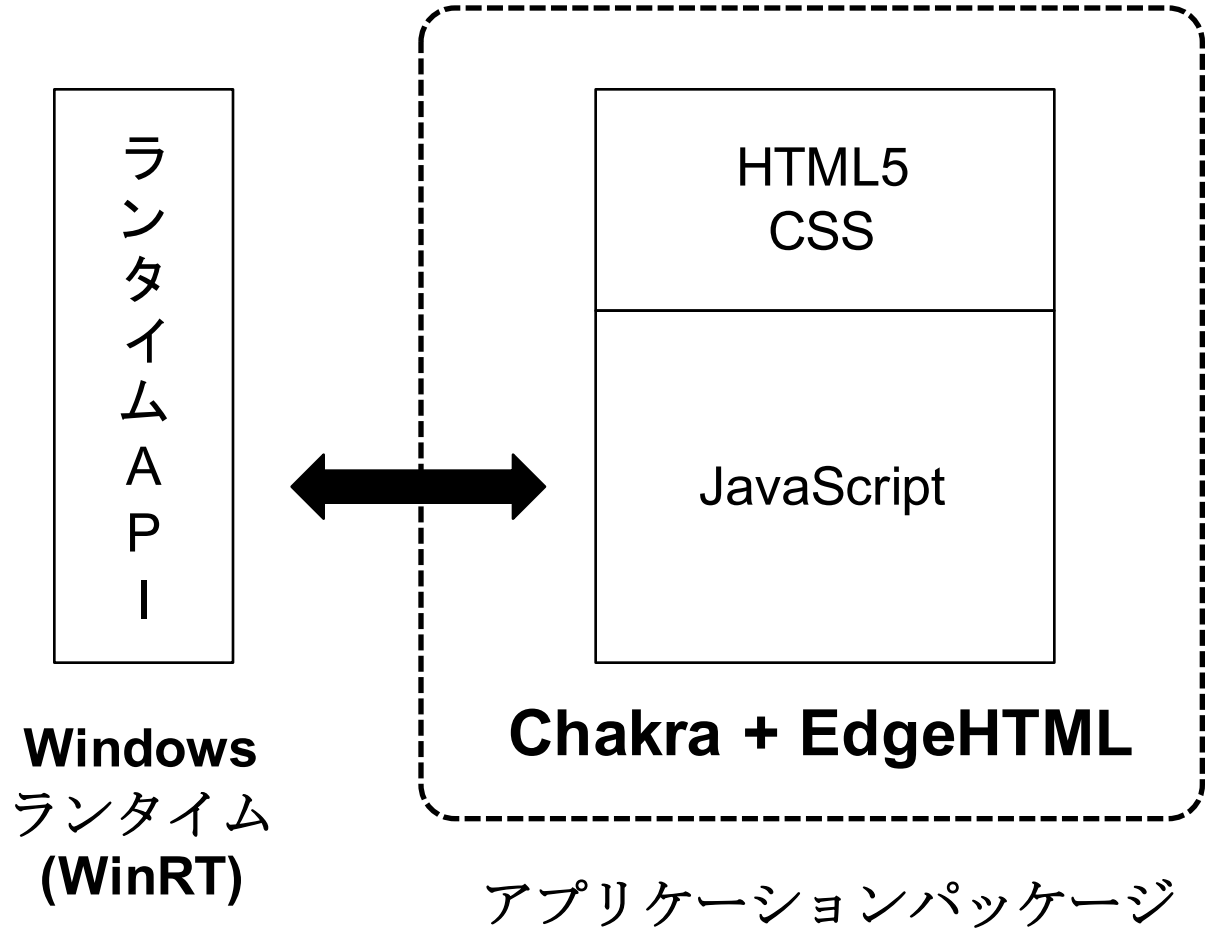
各社のエンタープライズ向けモバイル開発基盤はCordovaを採用



Cordovaフレームワーク （iOS/Android）



Cordovaフレームワーク （Windows/Windows Phone）

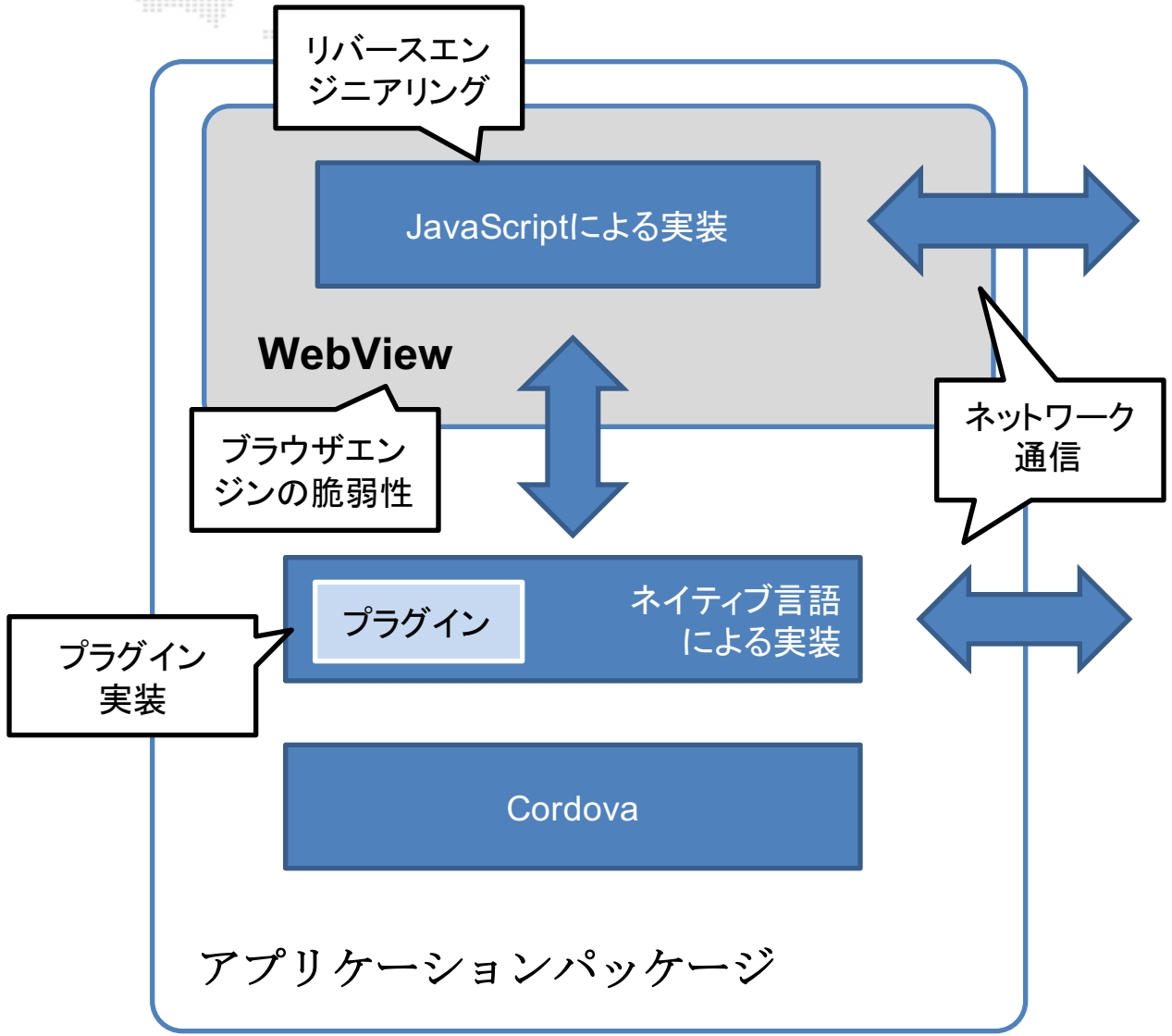


ハイブリッドアプリのセキュリティ

'2016 OWASP Mobile Top 10 Candidate

Improper Platform Usage	OS機能やセキュリティの仕組みの誤用。Androidインテントやパーミッションの間違い、Keychainの誤用など。
Insecure Data Storage	安全でないデータ領域の使用や意図しないデータ漏洩など。
Insecure Communication	ハンドシェイクの間違いや古いSSLバージョンの使用、暗号化されていない通信など。
Insecure Authentication	認証処理の不備やセッション管理の問題など。
Insufficient Cryptography	保護されるべきデータに対して、不完全な暗号を使用。
Insecure Authorization	認可処理の不備。クライアントサイドでの認可処理や、強制ブラウジングなど。
Client Code Quality	コードレベルの実装問題。バッファオーバーフローや文字列処理の間違いなど。
Code Tampering	バイナリファイルの改ざんやリソースファイルの変更、動的なメモリの改ざんなど。
Reverse Engineering	バイナリファイルからのソースコード復元、使用ライブラリやアルゴリズムの抽出など。
Extraneous Functionality	隠しバックドアの存在や、リリースアプリに含まれてはいけないコメントの混入など。

ハイブリッドアプリのセキュリティ懸念



- ネットワーク通信
  - WebViewから（ページの読み込み、XHRでのAjax）
  - ネイティブから（Android/iOS）
- WebView
  - ブラウザエンジンの脆弱性
- Cordova本体
  - Cordovaの脆弱性
- プラグイン
  - プラグインの脆弱性
- リバースエンジニアリング
  - ロジックの漏洩やコードの改ざん
- ソースコード
  - ソースコードの脆弱性

ネットワーク通信に対するセキュリティモデル

Same-Origin Policy

Cordovaが標準的に使用するiOSのUIWebViewやAndroidのWebViewは`file://`プロトコルがOriginとなっておりSame-Origin Policyは適用されない。よって、Content-Security Policyの指定が必要となる。

ただし、iOS 9 + Cordova iOS 4.0から利用できる「WKWebView」を用いる場合、WebViewの不具合からSame-Origin Policyが適用されるため、サーバー側にCORSヘッダーを付与する必要がある。

Access OriginとAllow Navigation

Access Origin

設定ファイル(config.xml）にて記述する。初期バージョンより存在。
デフォルトは * （すべて許可）なので注意が必要。<video>タグやWebSocket等は制限されないため、Content Security Policyの利用が推奨される。

```
<access origin="http://example.com" />
<access origin="*" />
```

Allow Navigation

Cordova iOS/Android 4.0以上に導入されたセキュリティモデル。Androidは[cordova-plugin-whitelist]の導入が必要（デフォルトで組み込み済）。WebViewで遷移可能なページを制限する。
デフォルトは`file://`から開始される（端末内に存在する）URLにナビゲーションを許可する。

```
<allow-navigation href="http://example.com/*" />
```

Content Security Policy

HTML内に<meta>タグを用いて記述する。KitKat（Android 4.4）以降、およびiOS 7以降で対応。ただしCrosswalkを用いることで、Android 4.0以降で対応可能。

```
<meta http-equiv="Content-Security-Policy" content="default-src 'self' data: gap:
https://ssl.gstatic.com; style-src 'self' 'unsafe-inline'; media-src *">
```

- gap://プロトコル（iOSのUIWebView）およびhttps://ssl.gstatic.com（Androidで必要）を許可。
- eval()関数やインラインスクリプトを禁止。

```
<meta http-equiv="Content-Security-Policy" content="default-src *; style-src 'self' 'unsafe-
inline'; script-src 'self' 'unsafe-inline' 'unsafe-eval'">
```

- CSSとJavaScriptは同じオリジンのものを許可。
- eval()関数とインラインスクリプトも許可。

iOS Application Transport Security (ATS)

iOSの持つセキュリティ基準。将来的に設定が必須化される予定。ATSが有効の場合HTTP通信は許可されず、TLS 1.2以上が必須。
Cordovaでは<access>要素もしくは<allow-navigation>要素を指定すると、ATS設定を記述するInfo.plistファイルが上書きされ、設定が適用される。

iOS 10以降の指定に対応するため、下記のように<access>要素が拡張された。

```
<access origin='*' allows-arbitrary-loads-in-media='true' allows-arbitrary-loads-in-web-content='true' allows-local-networking='true' />
```

allows-arbitrary-loads-in-media
- trueの場合、AV Foundation Framework経由で読み込むメディアに対して接続制限が解除される。

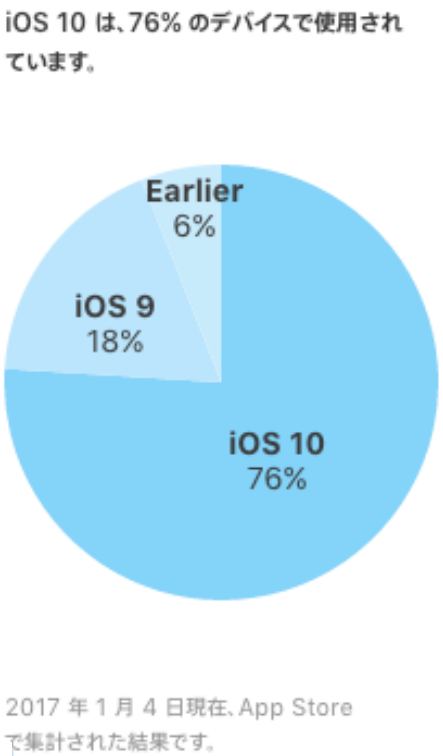
allows-arbitrary-loads-in-web-content
- trueの場合、WebViewから行われるリクエストに対して接続制限が解除される。

allows-local-networking
- trueの場合、ローカルネットワークにあるリソースに対して接続制限が解除される。

WebViewの脆弱性

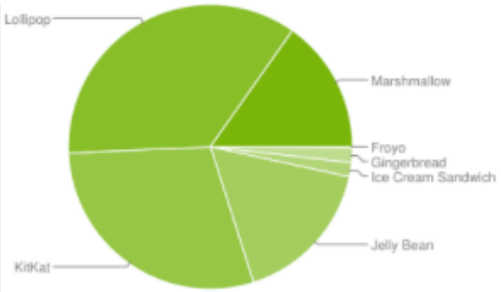
WebViewの脆弱性

自動アップデートが行われるiOSと比べ、ベンダーの対応に依存するAndroidではシステムコンポーネントであるWebViewがアップデートされない問題がある。ただし、Android 5からは、WebViewがOSと切り離され、Google Play Storeを通じて自動的にアップデートされるようになった。



Version	Codename	API	Distribution
2.2	Froyo	8	0.1%
2.3.3 - 2.3.7	Gingerbread	10	1.7%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	1.6%
4.1.x	Jelly Bean	16	6.0%
4.2.x		17	8.3%
4.3		18	2.4%
4.4	KitKat	19	29.2%
5.0	Lollipop	21	14.1%
5.1		22	21.4%
6.0	Marshmallow	23	15.2%

2016 年 8 月 1 日までの 7 日間に収集されたデータ。分布が 0.1% 未満のバージョンは表示されません。



iOSおよびAndroid OSのバージョン別シェア（各社Webサイトより）

iPhone/iPadでは比較的スムーズに世代交代が進むがAndroidでは約半数がバージョン4以下。

WebViewに脆弱性を持つAndroid 4.3以前のシェアは依然として高い（20%）。

WebViewの脆弱性への対応

ChromiumをWebViewとして使用できるようにしたサードパーティ製ライブラリである Crosswalkエンジンをアプリに組み込むことで、 Android 4以降でブラウザエンジンを最新に保つことが可能。

Cordova/プラグインの脆弱性

Cordovaに関する脆弱性の報告

CVE-2015-5208

Apache Cordova iOS before 4.0.0 allows remote attackers to execute arbitrary plugins via a link.

CVE-2015-5207

Apache Cordova iOS before 4.0.0 might allow attackers to bypass a URL whitelist protection mechanism in an app and load arbitrary resources by leveraging unspecified methods.

CVE-2015-5204

CRLF injection vulnerability in the Apache Cordova File Transfer Plugin (cordova-plugin-file-transfer) for Android before 1.3.0 allows remote attackers to inject arbitrary headers via CRLF sequences in the filename of an uploaded file.

CVE-2015-8320

Apache Cordova-Android before 3.7.0 improperly generates random values for BridgeSecret data, which makes it easier for attackers to conduct bridge hijacking attacks by predicting a value.

CVE-2015-5256

Apache Cordova-Android before 4.1.0, when an application relies on a remote server, improperly implements a JavaScript whitelist protection mechanism, which allows attackers to bypass intended access restrictions via a crafted URI.

Google Play Storeでの対応

脆弱性のあるCordovaが用いられている場合、Google Play Storeにて公開停止が行われる。

Google

Google ヘルプ　GOOGLE

Apache Cordova の脆弱性を含むアプリを修正する方法

この情報は Apache Cordova の 4.1.1 より前のバージョンを利用しているアプリのデベロッパーを対象としています。4.1.1 より前のバージョンにはセキュリティの脆弱性があり、この脆弱性はコンテンツ ポリシーの「危険なプロダクト」条項とデベロッパー販売 / 配布契約の第 4.4 項への違反にあたります。

できる限り早急にアプリを Apache Cordova v.4.1.1 以降に移行し、アップグレードした APK のバージョン番号を増やすようにしてください。Apache Cordova を含むサードパーティ製のライブラリを使用している場合は、そのサードパーティに通知し、連携して問題を解決するようにしてください。

脆弱性と修正期限

- CVE-2015-5256: バージョン 4.1.1 より前の Apache Cordova が対象です。これらのバージョンは、Android でホワイトリストによる制限が不正に適用されやすくなっています。この結果、ホワイトリスト制限が正しく適用されないという脆弱性が生じます。不適切に作成された URI を利用してホワイトリストが回避されることにより、ホワイトリストに登録されていない JavaScript の実行が可能になる恐れがあります。**2016 年 7 月 11 日以降、Google Play では、バージョン 4.1.1 より前の Apache Cordova を使用するすべての新規アプリおよびアップデートの公開が禁止されます。**
- CVE-2015-1835: バージョン 4.0.2 より前の Apache Cordova が対象です。これらのバージョンは、Android で Apache Cordova の二次設定変数がリモートで悪用されやすくなっています。問題となるアプリでは、Config.xml に明確な値が設定されておらず、未定義の設定変数がインテントで設定される可能性があります。この結果、不要なダイアログがアプリに表示されたり、アプリの動作に変化（アプリの強制終了など）が生じたりすることになりかねません。**2016 年 7 月 11 日以降、Google Play では、バージョン 4.1.1 より前の Apache Cordova を使用するすべての新規アプリおよびアップデートの公開が禁止されます。**
- CVE-2014-3502: バージョン 3.5.1 より前の Apache Cordova が対象です。脆弱性には、重大度の高いクロスアプリケーション スクリプティング（XAS）の脆弱性が含まれます。状況によっては、脆弱性のあるアプリはリモートで悪用され、ユーザーのログイン認証情報などの機密情報を盗まれる恐れがあります。**この脆弱性の修正期限は過ぎています。Google Play では、この脆弱性を含むすべての新規アプリおよびアップデートの公開が禁止されます。**

アップグレードや技術的なことに関してご不明な点がある場合

アップグレードについて詳しくは、Apache Cordova のウェブサイトをご覧ください。Apache Cordova に関するその他の技術的な質問については、https://www.stackoverflow.com/questions に、「android-security」および「cordova」というタグを使用して投稿してください。

正しくアップグレードされたことを確認するには、更新したバージョンをデベロッパー コンソールに送信して、5 時間後に再度確認してください。アプリが正常にアップグレードされていない場合、警告が表示されます。

注: こうした問題は、バージョン 4.1.1 より前の Apache Cordova を使用するすべてのアプリに影響するとは限りませんが、あらゆるセキュリティ パッチを常に最新の状態にしておくことをおすすめします。この機会に、古い従属ライブラリやその他の脆弱性を含むアプリを更新してください。

アプリを公開する前には、そのアプリがデベロッパー販売 / 配布契約とコンテンツ ポリシーに準拠していることをご確認ください。Apache Cordova の脆弱性に関する警告が誤って届いたとお考えの場合は、Google Play デベロッパー ヘルプセンターからサポートチームまでお問い合わせください。

現在はCordova Android 4.1.1以前で作られたパッケージはアップロード不可。

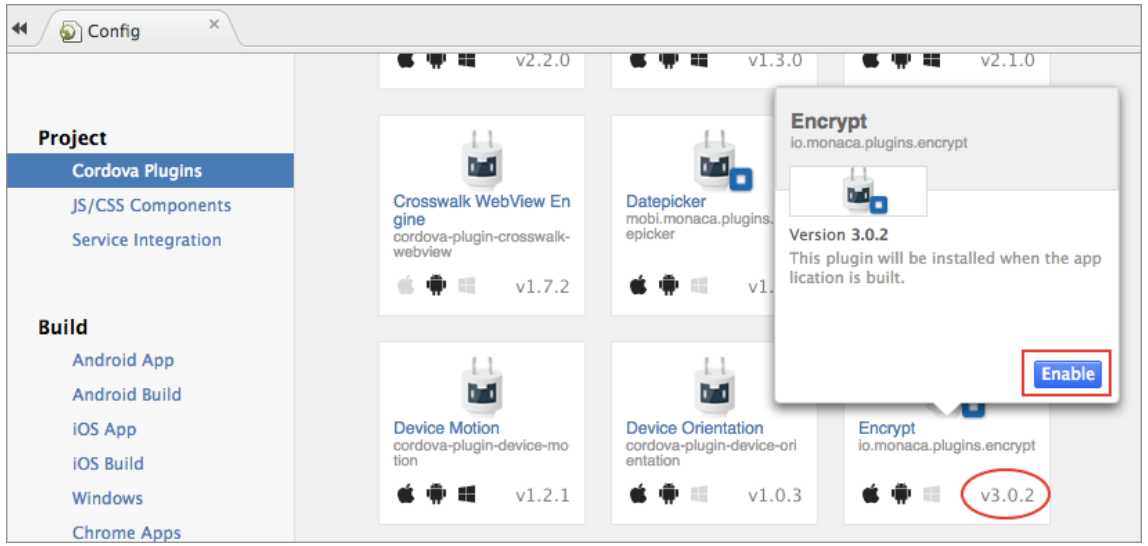
Cordovaの脆弱性が発生すると今後も対応が必要になる。メンテナンス体制を整えておく必要がある。

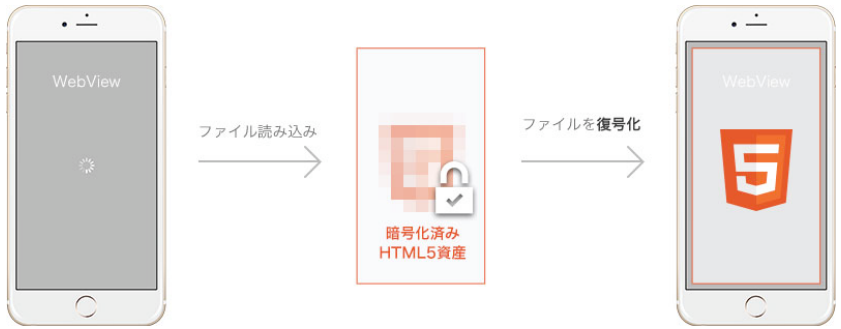
Google Play Storeでの説明記事
https://support.google.com/faqs/answer/6325474?hl=ja

リバースエンジニアリング

ハイブリッドアプリのアセットファイル（HTML/JavaScript等）はネイティブコードにコンパイルされず、ソースコードの状態で端末内に格納される。無用な記述を削除するため、コメントの削除や難読化（最適化）を実施することが望ましい。

より進んだ対策としては、ビルド時に暗号化を行い実行時に復号する、サードパーティ製の暗号化プラグインを使用する。復号に使うキーは、リモートサーバー等の安全な場所から取り出す必要がある。

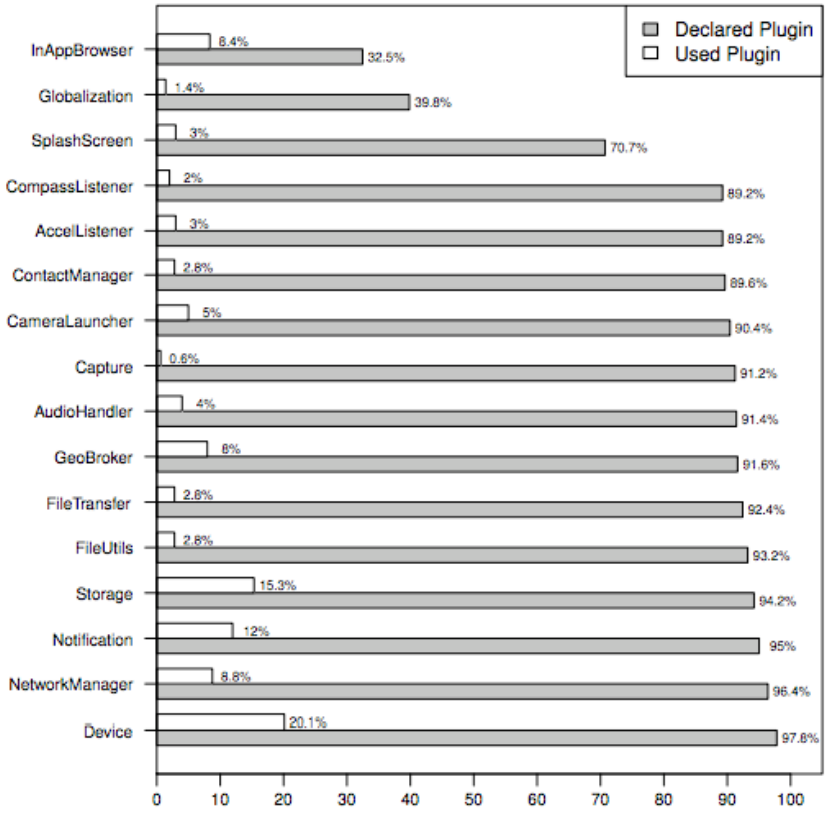




Monacaにて提供されるアセット暗号化プラグイン
https://ja.monaca.io/enterprise.html

不要なプラグインの除去

Cordova向けアプリテンプレートの多くが、Cordova Core Pluginsと呼ばれる基本プラグインをすべて同梱している。実際には使用していないコードが含まれてしまい、脆弱性の温床になる可能性がある。



PhoneGapのWebサイトに掲載されていた622件のアプリに対しての調査（2014年）では、プラグインの組み込みと実際の利用に大きな開きが見られている。

Mohamed Shehab et. al, **Reducing Attack Surface on Cordova-Based Hybrid Mobile Apps**, MobileDeLi 2014

セキュアコーディング

コーデイングの指針

Cordovaアプリの大半はSPA型であり、モバイルWebサイトと比べてもJavaScriptのウェイトが高い。セキュアコーディング対策としてはHTML5向けのプラクティスを適用できる。

- ユーザー入力データのバリデーション
- 適切なエスケープ処理
- JavaScriptインジェクションの回避
- APIキーの適切な管理

エスケープ処理

コンテキストに応じたエスケープ処理が必要であるが、適切な記述を行うのは困難。

SPAフレームワークやテンプレートエンジンの採用が望ましい。

エスケープ処理を自前で実装するのは困難であるため、フレームワークを採用することが望ましい。

下記の例ではすべての変数に別のエスケープ処理が必要

```
<div onmouseover="${data1}">${data2}</div>
<a href="${data3}">link</a>
<iframe srcdoc="${data4}"></iframe>
```

Nishimura Muneaki et al, ハイブリッドアプリケーションの脆弱性に関する分析 2015

バインディングをサポートするフレームワークやライブラリ

2-Way Bindingを持つもの
- Angular 1
- Angular 2

1-Way Bindingを持つもの
- React

テンプレートエンジン
- Handlebars
- Mustache

参考情報、ドキュメント等

- Cordovaドキュメント
  - Security Guide
  - Whitelist Guide

- OWASP HTML5 Security Cheat Sheet

- ハイブリッドアプリケーションの脆弱性に関する分析
  - 西村宗晃 et al, 第14回情報科学技術フォーラム（2014）

ありがとうございました

問い合わせ先：

アシアル株式会社 田中正裕
masahiro@asial.co.jp