

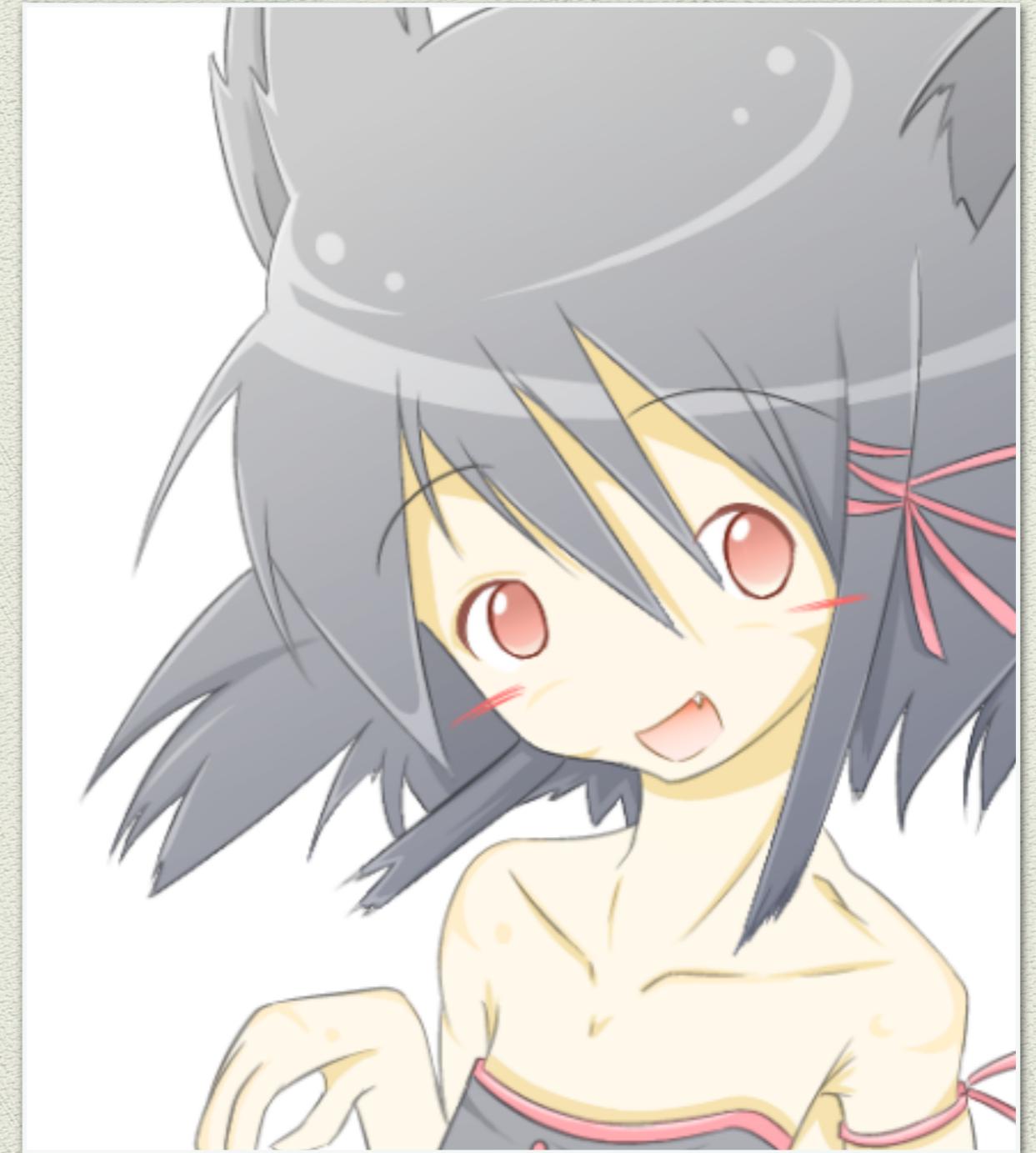
セキュリティ ガイダンス

Mobile Factory, Inc.

ice break

だれ？

- ◆ Kenta SATO
 - ◆ id:karupanerura
 - ◆ age:24 (1990/11/21)
- ◆ Perl/Swift/Java/JS/etc..
- ◆ Mobile Factory, Inc.
 - ◆ Lead Engineer
 - ◆ 2011/04/01~



趣味や興味

- ◆ お酒好き
 - ◆ 特に日本酒とウイスキーと芋焼酎がすき
- ◆ 好きな銘柄
 - ◆ 鍋島、屋守、竹鶴、富士山麓、富乃宝山、赤霧島、etc..
- ◆ 美味しいお酒をちびちび呑むのが好きです

趣味や興味

- ◆ 音楽好き
 - ◆ ロック/ブルース/Popsが好き
 - ◆ でもなんでも聴く
 - ◆ 曲作ったりバンド組んだり歌ったり
 - ◆ レコーディングやミキシングなどサウンドのエンジニアリングも好き

趣味や興味

- ◆ 技術好き
 - ◆ プログラミング、データベース、etc..
 - ◆ 最近は運用などの自動化に関心が高い
 - ◆ いちばん好きな言語はPerl

セキュリティ関係でやったこと

- ◆ 内製フレームワークの脆弱性の発見と対策
- ◆ ソーシャルゲームのチート対策
- ◆ ソーシャルゲームの認証システムの構築
- ◆ ユーザーのなりすまし対策
- ◆ etc..

本題

Agenda

- ◆ セキュリティとは
- ◆ あってはならないことはなにか
- ◆ どのように攻撃するのか
- ◆ どのように防ぐのか
- ◆ まとめ

セキュリティとは

セキュリティリスク

- ◆ プログラム改ざん
- ◆ コンテンツ改ざん
- ◆ プログラムの不正実行
- ◆ コマンドの不正実行
- ◆ システムの破壊

セキュリティリスク

- ◆ プログラム改ざん => 情報漏えいなど
- ◆ コンテンツ改ざん => 詐欺など
- ◆ プログラムの不正実行 => 情報漏えいなど
- ◆ コマンドの不正実行 => 情報漏えいなど
- ◆ システムの破壊 => 営業妨害など

セキュリティリスク

- ◆ プログラム改ざん => 情報漏えいなど
- ◆ コンテンツ改ざん => 詐欺など
- ◆ プログラムの不正実行デシント情報漏えいなど
- ◆ コマンドの不正実行 => 情報漏えいなど
- ◆ システムの破壊 => 営業妨害など

セキュリティとは

- ◆ プログラム改ざん
- ◆ コンテンツ改ざんのリスクから
プログラムの不実行や顧客を
- ◆ コマンドの不実行
- ◆ システムの破壊

要するに..

情報の安全保障

セキュリティの考え方

リスクから守るために

- ◆ あってはならない事はなにか
 - ◆ ユーザー視点、サービス提供者視点
- ◆ どのように攻撃するのか
 - ◆ 攻撃者視点
- ◆ どのように防ぐのか
 - ◆ 開発者視点

あつてはならないことはなにか

- ◆ サービスによります（極論）
- ◆ 情報漏えい
- ◆ データ改ざん
- ◆ ユーザーなりすまし
- ◆ その他、もろもろ

あつてはならないことはなにか

- ◆ サービスによります（極論）
- ◆ 情報漏えい
- ◆ データ修復不可能な被害
- ◆ ユーザーなりすまし
- ◆ その他、もろもろ

どのように攻撃するのか

- ◆ ブラウザのバグを悪用する
- ◆ アプリケーションのバグを悪用する
- ◆ アプリケーションの設計を悪用する
- ◆ ミドルウェアのバグを悪用する
- ◆ サービスの性質を悪用する
- ◆ 人間を悪用する

どのように攻撃するのか

- ◆ ブラウザのバグを悪用する
- ◆ アプリケーションのバグを悪用する
- ◆ アプリケーションの設計を悪用する
- ◆ ミドルウェアのバグを悪用する
- ◆ サービスの性質を悪用する
- ◆ 人間を悪用する

どのように攻撃するのか

- ◆ ブラウザのバグを悪用する
- ◆ アプリケーションのバグを悪用する
- ◆ アプリケーションの設計を悪用する
- ◆ ミドルウェアのハックを悪用する
- ◆ サービスの性質を悪用する
- ◆ 人間を悪用する

脆弱性

脆弱性の見つけ方

- ◆ 技術仕様の穴を見つける
 - ◆ e.g.) DNSキャッシュポイズニング/etc..
- ◆ アプリケーションの実装の穴を悪用する
 - ◆ e.g.) XSS/SQLi/Session Fixation/etc..
- ◆ アプリケーションの仕様を悪用する
 - ◆ e.g.) XSS/SQLi/Open Redirector/etc..

脆弱性の昨今

- ◆ フレームワークやライブラリレベルで対策済
- ◆ 正しく使えばXSS/SQLi等は自然に防げる
- ◆ フロントエンドの実装や設計の穴による被害
- ◆ DOM Based XSSなど
- ◆ 複数の仕様を組み合わせたケースのバグ

攻撃に必要な知識

- ◆ 攻撃に応用できるレベルの知識
 - ◆ 技術的な基礎に対する深い理解と応用力
 - ◆ e.g.) OS/プロトコル/言語/etc..
- ◆ メジャーな攻撃方法と対策の知識
 - ◆ 多くの攻撃はメジャーな攻撃方法の応用
 - ◆ 対策を避けるあるいは無効化する手段

攻撃に必要な知識

- ◆ 最新の技術、トレンドに対する理解と研究
- ◆ 新しい技術にはたいてい穴がある
- ◆ むかし安全だったものが技術的変化により安全でなくなる場合もある
- ◆ リスクの小さかった脆弱性が技術的変化によりリスクが高まる場合もある

めっちゃ大変では？？？

なんのために攻撃するの？

攻撃者のモチベーション

- ◆ 金銭的なメリット
 - ◆ e.g.) ネットバンキング/ECサイト/etc..
 - ◆ e.g.) ゲームのアイテムの不正取得/etc..
- ◆ 政治的なメリット
- ◆ 個人的なうさ晴らし/技術的な腕試し/etc..

他人ごとではない！

多くのWebサイトは攻撃を
常に受けている

どのように防ぐのか

- ◆ セキュリティに配慮した設計をする
- ◆ セキュリティに配慮したライブラリを使う
- ◆ セキュリティ上問題がある機能を避ける
- ◆ 与える権限を最小限にする

リスクとコスト

- ◆ 脆弱性を防ぐためにはコストがかかる
- ◆ 間雲に堅くしてもコストが無駄になるだけ
- ◆ 対策が甘いとインシデントに繋がる
- ◆ バランスの良い落とし所を考える
- ◆ ビジネスとして大事なことです

リスクとコストと優先順位

- ◆ ローリスクの脆弱性対応はオミット可能な場合も
- ◆ ハイリスクでも対処しにくいケースがある

表1: リスクとコストに対する対処の優先順位の付け方の例

(例)	ハイリスク	ローリスク
ハイコスト	2	4
ローコスト	1	3

具体的にどのようなことを
学ぶべきか

この研修のゴール

セキュアなコーディングとは?
というところに悟りを開く

身に付けてもらいたいこと

- ◆ 攻撃者の視点と考え方
- ◆ セキュリティを高めるための機能の知識
- ◆ メジャーな攻撃方法とその対策
- ◆ 技術的な基礎に対する理解
- ◆ 技術トレンドの変化への関心

この後の予定

- ◆ プロトコルとブラウザの基礎知識の解説
- ◆ 関連するメジャーな脆弱性の解説
- ◆ アプリケーションの実装に起因する脆弱性
- ◆ いくつかのパターンの紹介とその対策

質疑応答