Webサイトが今、危ない！！

Webアプリケーションのセキュリティを考える

JNSA
Webアプリケーションセキュリティ
WG

頻発する情報流出

- 狙われるWebアプリケーション
  - 旅行会社Webサーバからの個人情報流出
  - 人材派遣会社Webサーバからの個人情報流出
  - 価格比較サイトWeb改ざんと情報流出

2005年春ごろから事件が急増・・・・・

事例紹介・・・

- 旅行会社などのサーバーに侵入容疑で容疑者逮捕（新聞報道などから）
  - 旅行会社のサーバに不正アクセスし情報を盗んだ疑い（不正アクセス禁止法違反容疑）
  - 価格比較サイト、人材派遣会社などのホームページにも「侵入」、情報を盗んだ疑いが浮上
  - 自宅のパソコンから旅行会社が管理するサーバーに不正に侵入。特殊な指令を入力し、ネットでツアーを申し込んだことがある会員のIDやパスワード、住所、氏名、性別、生年月日、メールアドレスなどを盗んだ疑い。（日経7/6付記事から一部引用）
  - 人材派遣会社サイトから、SQLインジェクションにより情報を盗んだ。（http://www.asahi.com 9/8付記事より）

ビジネスインパクトは？

- 収入源であるサイトの長期停止
  - 原因究明、捜査のための証拠保全・・・
  - 再発防止策、ソフトウエア修正
  - システムの再構築・・・・
- 個人情報漏洩の場合は、お詫びや補償も・・・
  - 一人５００円～数万円・・・・
- 中長期的業績や株価への影響
  - プライバシーに敏感な世相とユーザの不安
  - 社会的な批判、制裁
  - コンプライアンス違反、所轄官庁からの処分

「最高水準」だったはずの対策

- ファイアウォールによるWebサーバ保護
- 侵入検知システムの導入
- 最新パッチ適用によるセキュリティホール対策
- ウイルス対策ソフトの導入と常時更新
- いったい何が足りなかったのか・・・・・

思わぬ落とし穴・・・

- インフラは完璧だったが・・・
  - Web上に公開されていたアプリケーションは？
    - 自社開発の対話型アプリケーション
    - 背後にあるデータベースと連動してユーザ情報を管理
    - はたして問題はなかったのだろうか・・・
  - アプリケーションへの想定されていない操作、入力の影響
    - 入力チェックに問題があると「誤動作」を引き起こす可能性が生じる。（想定外の入力の悪影響）
    - 「誤動作」の内容が問題・・・

なくならないアプリの脆弱性

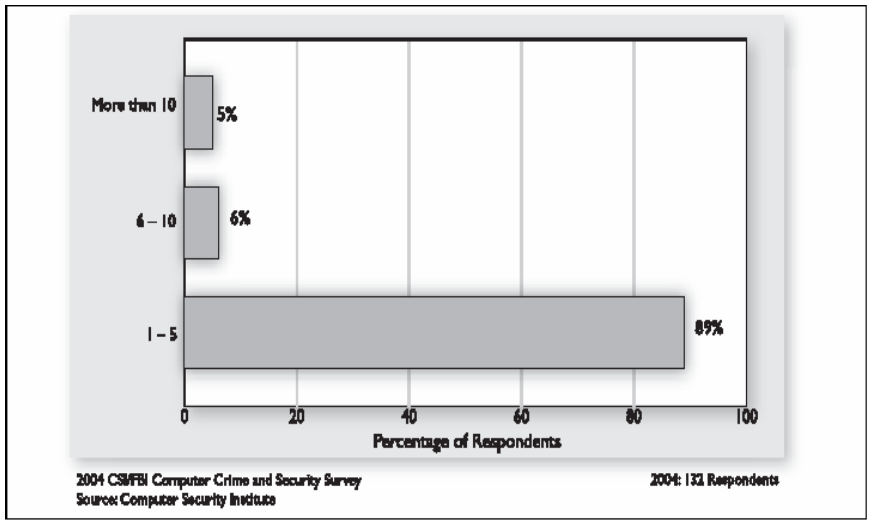
- 次々発覚するWebアプリケーション脆弱性
  - ソフトウエア等の脆弱性関連情報に関する届け出状況（IPA JPCERT/CC）
  - http://www.ipa.go.jp/security/vuln/report/documents/vuln2005q2.pdf
  - 2004年9月～2005年6月までに277件の報告

CSI/FBI Survey 2005

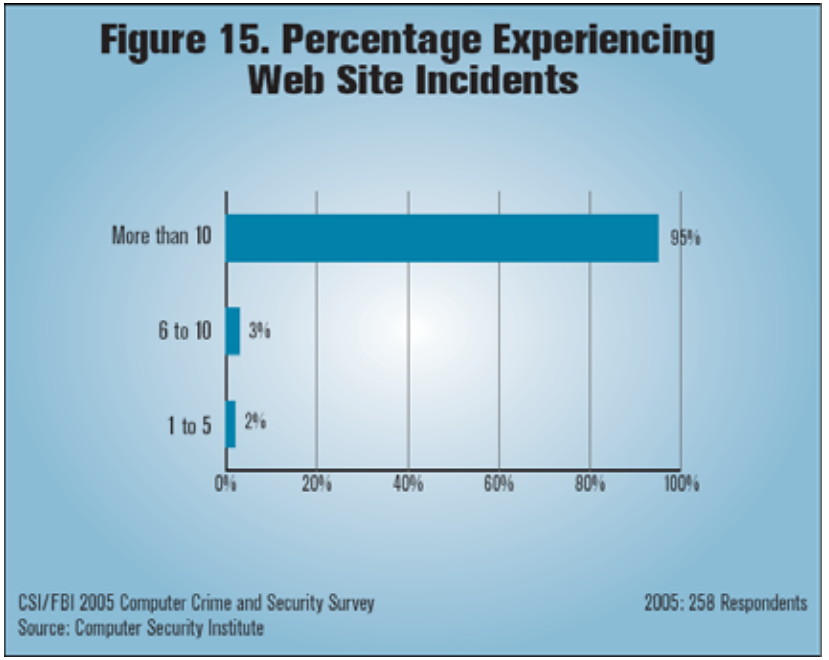
- **世界的なWebサイトへの攻撃増加傾向**
  - **アプリケーションを標的とした、より深刻な攻撃がこの1年間で激増している**

米国CSI (http://www.gocsi.com/) FBI 共同調査から

2004年調査



2005年調査



世界的なWebアプリ攻撃増加

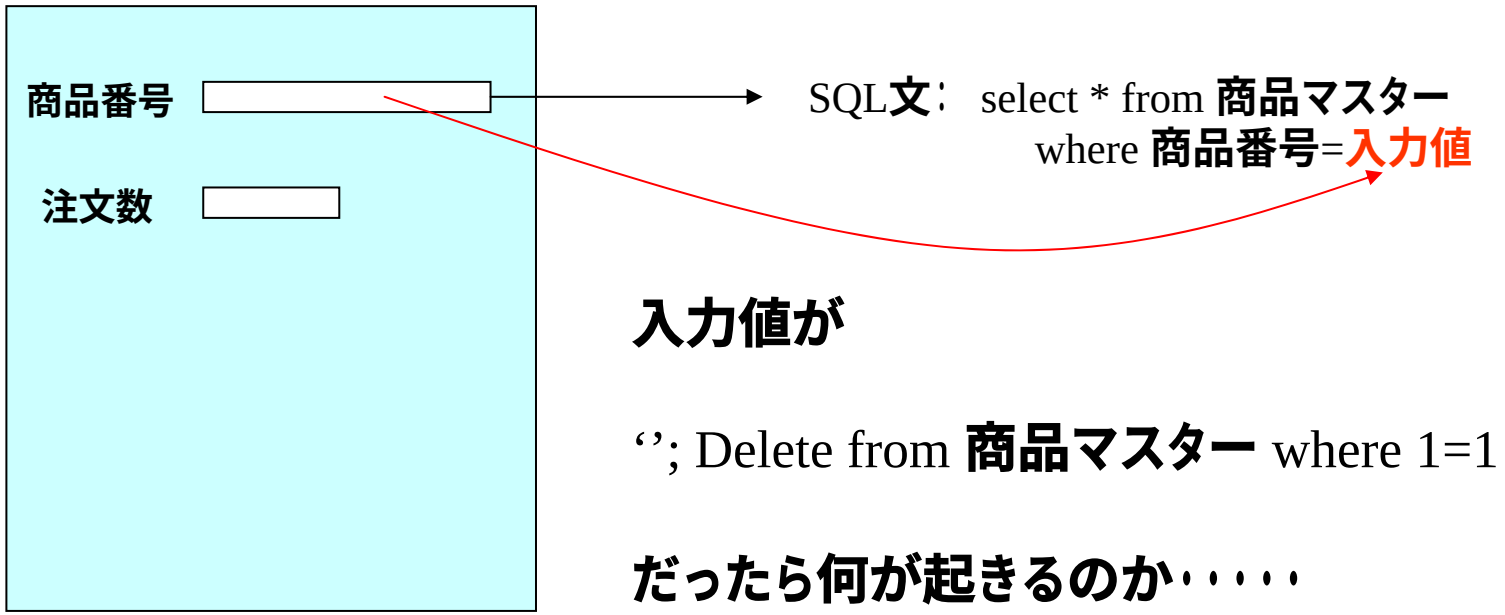
- CSI/FBI Computer Crime and Security Survey 2005年版より
  - http://www.gocsi.com/press/20050714.jhtml
  - Web にかかわるインシデントが前年比で激増、アンケート回答組織の９５％が過去１０件以上のインシデントを経験 (昨年調査では、８９％が5件以下)
  - 不正アクセス被害激増で米国内被害額ランキング第2位に

アプリケーションへの代表的攻撃

- SQLインジェクション
  - データベースを利用するアプリケーションに対して、外部からの入力で不正にデータベースを操作、参照する手法
- パラメータ改ざん、セッションハイジャック
  - ブラウザ、アプリケーション間で受け渡される各種のパラメータを改ざんすることで、想定外の動作を誘発させたり、セッションIDなどの情報を横取りして再利用することで、認証を回避してアプリケーションを操作するなどの手法
- クロスサイトスクリプティング
  - アプリケーションが生成するWebページに外部からの操作で不正なスクリプトを埋め込み、それを参照した人が、気づかずに不正なサイトに誘導されたり、不正プログラムをダウンロードさせられたりするような手法。

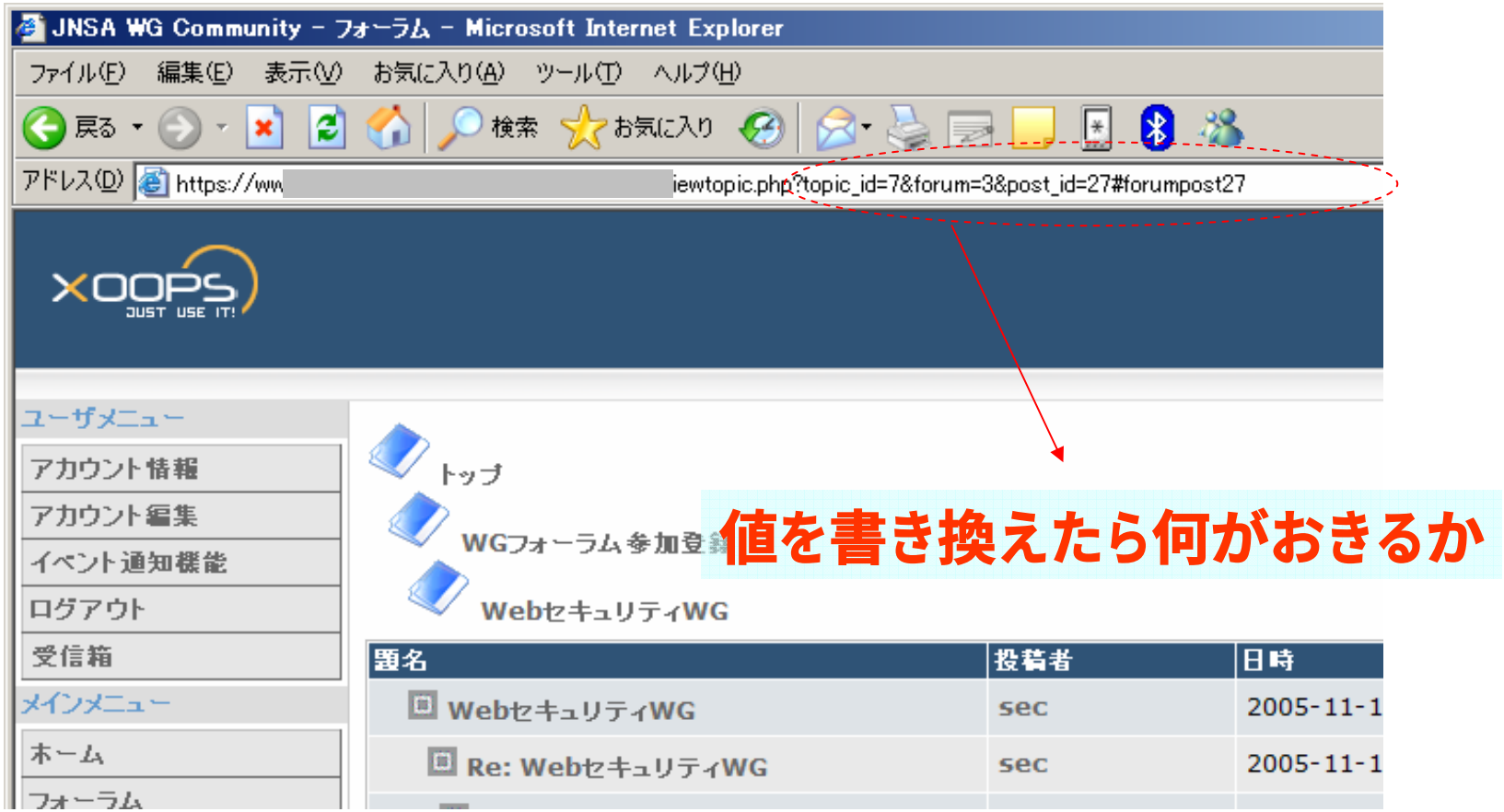
アプリケーション攻撃例

- SQL インジェクション



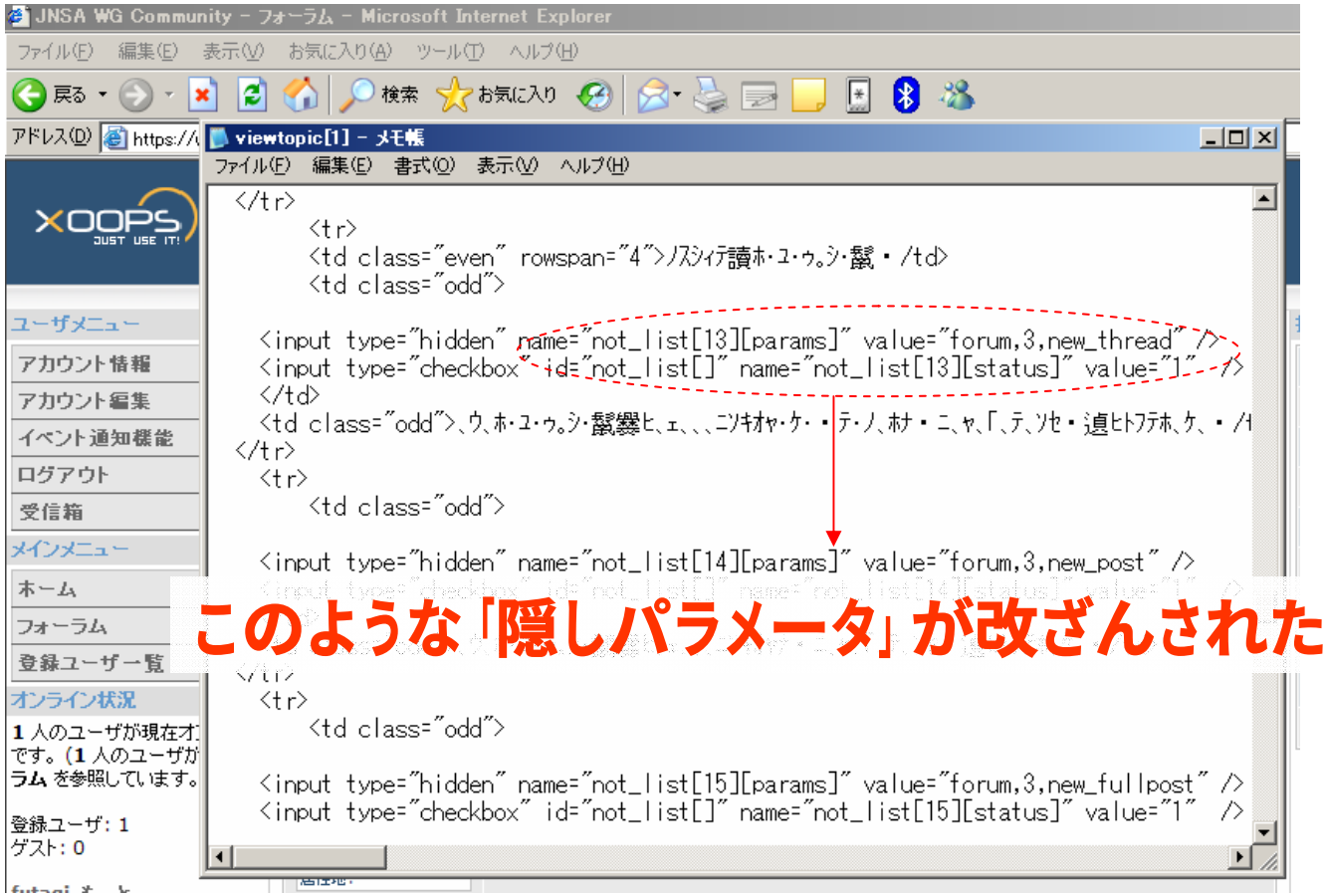
アプリケーション攻撃例

- パラメータ改ざん

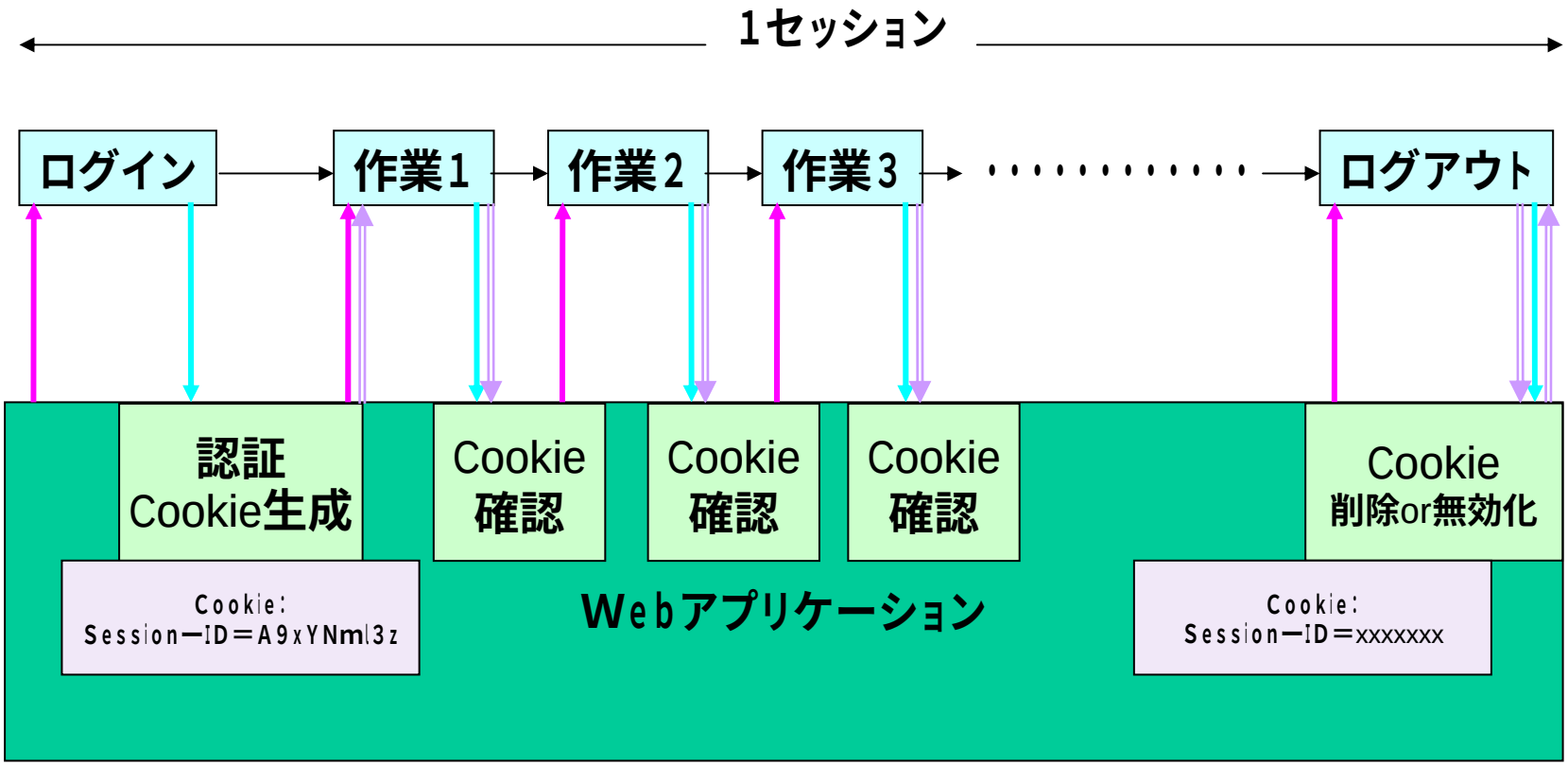


アプリケーション攻撃例

- パラメータ改ざん



Cookieを使ったセッション管理



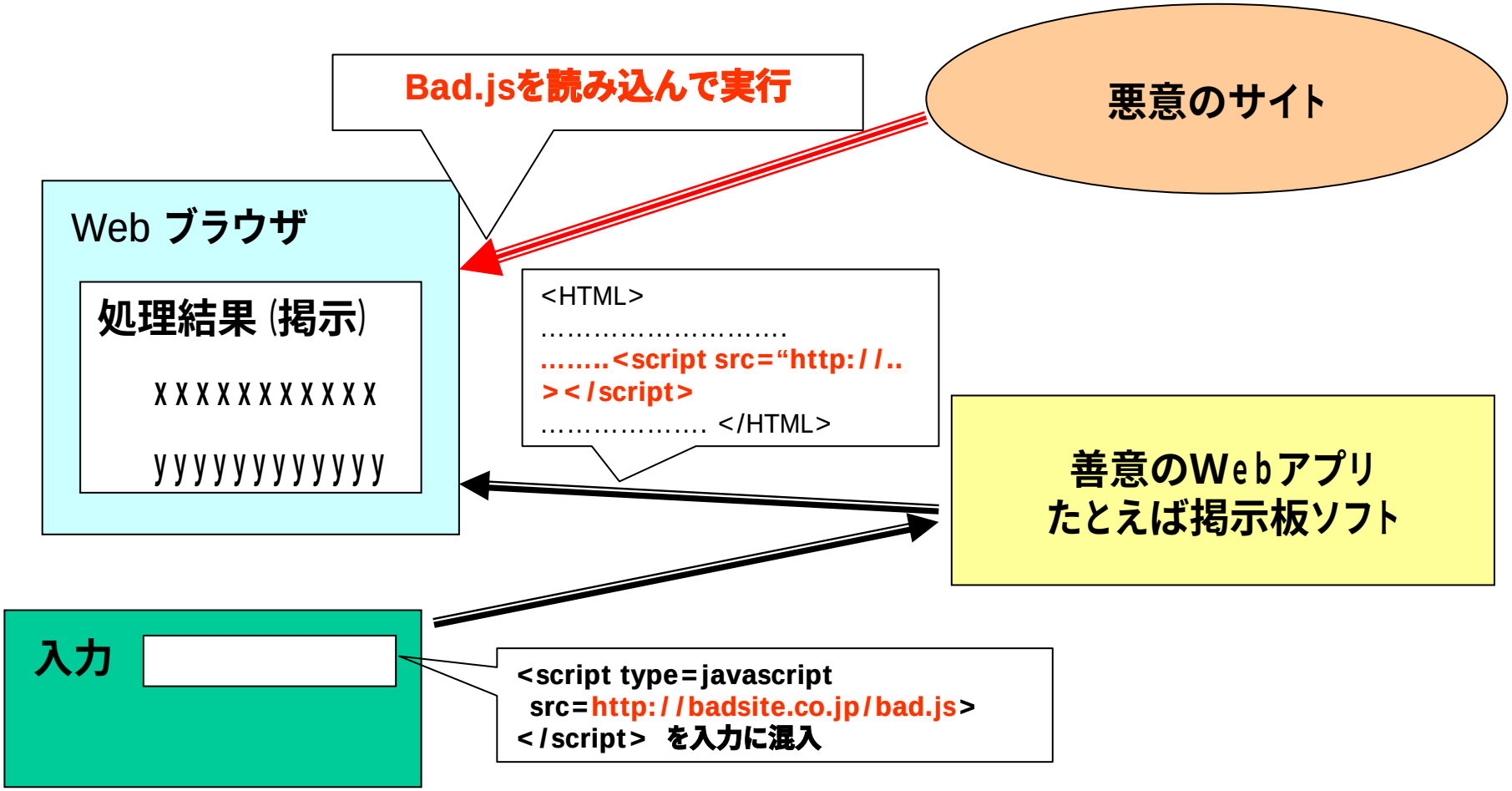
Cookie横取り、改ざん、再利用

- Cookieに含まれる内容例

これらを改ざんされたり、横取りして再利用されたら？？

アプリケーション攻撃例

- クロスサイトスクリプティング脆弱性の悪用例



攻めやすい「アプリケーション」

- インフラへの攻撃による情報抜き取りには煩雑な手順、道具が必要
- アプリケーションから情報を抜き取るにはブラウザさえあればいい
- プロの「泥棒」は、もはやインフラではなくアプリケーションを標的にしている
- インターネット上での情報、ツールの流通

個人情報への脅威

- 「泥棒」の標的として見たWebアプリの特性
  - 多数のコンシューマを相手にするものが多い
  - 多くの場合、なんらかの個人情報を取り扱う
  - 汎用的なデータベースを利用し、入力された内容を利用して検索、更新を行うことが多いため、アプリケーションが大規模なデータベースと直結していることが多い

脆弱性はどうして生じるか

- 存在しうる脅威に対する認識の欠落
  - 要求仕様、設計からのセキュリティ要件欠落
- 安全を考慮したプログラミング技法、ツールなどへの知識の不足
- 検査方法、体制の不備（脅威を意識した検査）
- 基本的には開発側の問題のように見えるのだが・・・・

「環境」「風土」の問題

- アウトソーシング主体の開発
  - 受注側、発注側ともにセキュリティに関する知識、経験が不十分
- セキュリティ要件（ポリシー）を誰が決めるのか
  - 発注側主導で決めるのが理想だが・・・
  - 受注側が提案→発注側が承認という形もある（但し、責任をどこが、どこまで持つかという点は協議しておく必要あり）
- 最低限の専門知識を持つ人材と予算確保も必要
  - 自分たちのセキュリティを考える（決める）のはユーザ自身の責任

現実的な脆弱性対策とは？

3つの重要な視点

- 安全なWebアプリを開発するには
- Webアプリが安全かどうかを確認するには
- 安全でないWebアプリを守るには

開発時の対策

- 開発者の教育
- 仕様へのセキュリティ要件の組み込み
  - 発注者と受注者の合意に基づく具体的な要件
- セキュリティを考慮した標準化の推進
  - 安全なプログラミング手法、ツールの研究
- ＱＡ（品質保証）手順の確立
  - セキュリティ要件をQAプロセスに組み込む
  - 適切な検査手法、ツール等の導入

安全性の検査

- 発注側による受け入れ検査
  - 発注時の要件に基づく安全検査合格を検収条件に
    - 要件が具体的かつ発注時に合意した内容である必要性
  - 重要度の高いシステムについては、第三者による検査も
    - 要件に定められていなかった問題は判明した場合について、受発注側双方で、あらかじめ取り扱いを協議しておく

運用中のシステムへの対策

- 運用中のシステムの検査
  - 脆弱性検査
    - 発現のしかたが単純な脆弱性は検査ツールを利用して発見
      - アプリケーション開発プラットホームやミドルウエア等の既知の脆弱性に関するチェックは充実
    - 複雑な発現をする脆弱性には専門技術者によるマニュアル検査で対応
    - 基本的にできる限り網羅的に脆弱性を発見することを目的とする
  - ペネトレーション検査
    - 網羅性よりも発見された脆弱性の深刻度（リスク）評価が目的
    - 侵入や情報窃取などのシナリオを実行可能かどうかを見極めるもの
- 検査に伴うリスク回避
  - システムに悪影響を与える可能性があるため、運用中のシステムに対して行える検査は限定的
    - システムにデータを入力して検査するため、データベースに無意味なデータを残す可能性があるなど
  - 予備系システムがあれば、そちらで実施したほうがよい
    - 少なくとも使用するソフトウエアについては同一の環境に

発見した脆弱性への対応

- ただちに修正する・・・が原則、だが・・
  - すぐにできない理由も・・・
    - コスト、リソース
    - 社内にソースコードを扱える人間がいない。外注先の会社が消滅、もしくは担当者が退職などの理由で対応できない
    - 修正できるが、様々な原因で修正、テスト、リリースまでに長時間を要してしまう。
  - 運用を止められない
    - 修正の影響（副作用）も心配
- 修正せずに長期間放置するのは、きわめて危険
  - 修正できない場合は、なんらかの回避策を

攻撃からの防御

- 修正できないアプリケーションへの攻撃回避策
  - 脆弱な機能の利用停止や代替機能の提供
  - Webアプリケーションファイアウォールを使用した攻撃防御
    - Webアプリケーション保護に特化した侵入防御製品
    - 不正な入力をアプリケーション（Webサーバ）に渡す前にブロックする

段階的なアプローチ

開発段階	導入段階	運用段階
セキュリティ要求の仕様化、要件化	受け入れ検査 運用前検査 の実施	脆弱性検査 ペネトレーション検査 の実施
安全を考慮した 開発（設計）手法 開発ツール の利用		
開発者への教育		発見された脆弱性の 修正 防御

大切なWebサイトを守るには

- アプリケーションに対する脅威を理解しよう
- 開発～導入～運用の各ステップで対策を

Webアプリケーションセキュリティ WG
啓発コンテンツ分科会

- 参加企業、参加者 (敬称略)
  - IIJテクノロジー株式会社　加藤
  - NTTコムウエア株式会社　丸尾
  - 富士通株式会社　奥原
  - 住商情報システム株式会社　二木
  - ＪＮＳＡ
    - 安田 (ディアイティ、事務局) 、佐藤 (IRI, 技術部会長)