付録

アプリケーションへの代表的攻撃

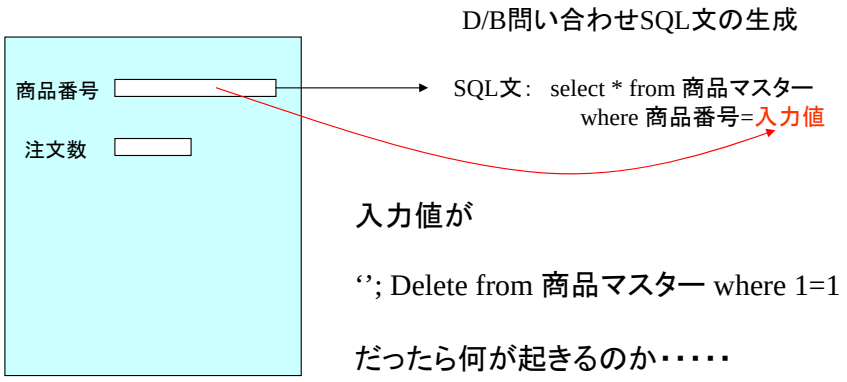
- （SQL）インジェクション
  - データベースを利用するアプリケーションに対して、外部からの入力で不正にデータベースを操作、参照する手法
  - 入力がSQL以外のシステムのコマンド等として処理される場合に、それらを利用する、コマンドインジェクションなども類似の攻撃
- パラメータ改ざん、セッションハイジャック
  - ブラウザ、アプリケーション間で受け渡される各種のパラメータを改ざんすることで、想定外の動作を誘発させたり、セッションIDなどの情報を横取りして再利用することで、認証を回避してアプリケーションを操作するなどの手法
- クロスサイトスクリプティング（XSS）
  - アプリケ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。
- クロスサイト・リクエスト・フォージェリー（XSRFまたはCSRF）
  - 不正なスクリプトによって、利用者の権限で異なるサイトに対して操作を行うような攻撃手法。XSSの応用として行われることが多い。

JNSA Web アプリケーションセキュリティWG 2005（抜粋）

ここで、最近注目されている、Webアプリケーションへの攻撃手法をいくつか見てみよう。これらはすべて、アプリケーションがこうした攻撃に対して脆弱である（つまりは、不正な入力などを許してしまう）ことが前提となっている。逆の言い方をすれば、このような攻撃を許さないためには、攻撃に利用される不正な入力やデータ改ざんをきちんとチェックし、許さないことが必要だ。

アプリケーション攻撃例

- SQL インジェクション



JNSA Web アプリケーションセキュリティWG 2005（抜粋）

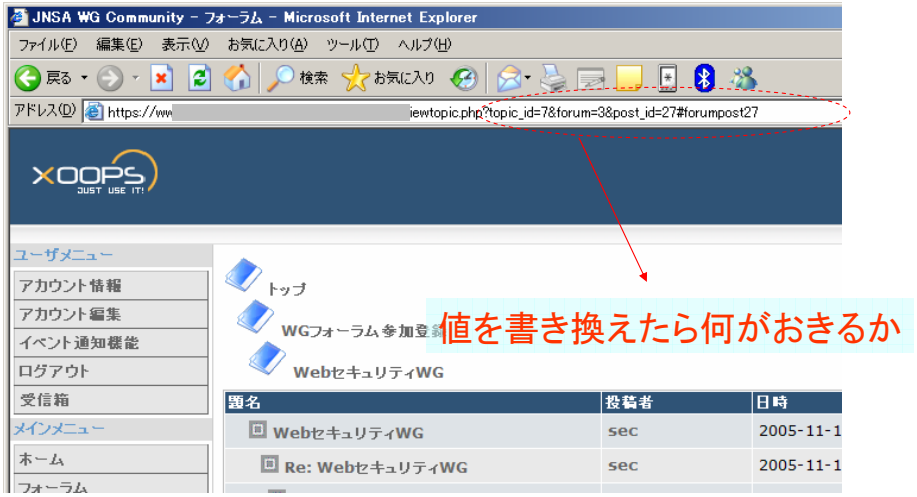
これはごく単純なSQLインジェクションの例だ。

この例では、アプリケーションは入力された商品番号から商品マスターを検索するが、その際に入力値を利用して検索のためのSQL文を構成する。この処理をなんらチェックも行わずに単純な文字列結合で行ってしまうと、入力を操作することで、構成されるSQL文を任意の処理を実行するように改ざんできてしまう。

上の例では、商品マスターの内容は消去されてしまう。このような破壊行為や、本来表示されてはいけない個人情報などのデータを表示させるような攻撃にSQLインジェクション脆弱性を利用することができる。また、使用するデータベースによっては、オペレーティングシステムのコマンドをSQL文から実行可能な場合もあり、このようなケースでは、システムへの侵入やWeb改ざんなどの不正操作に利用できる可能性もあるので注意が必要だ。

アプリケーション攻撃例

- パラメータ改ざん



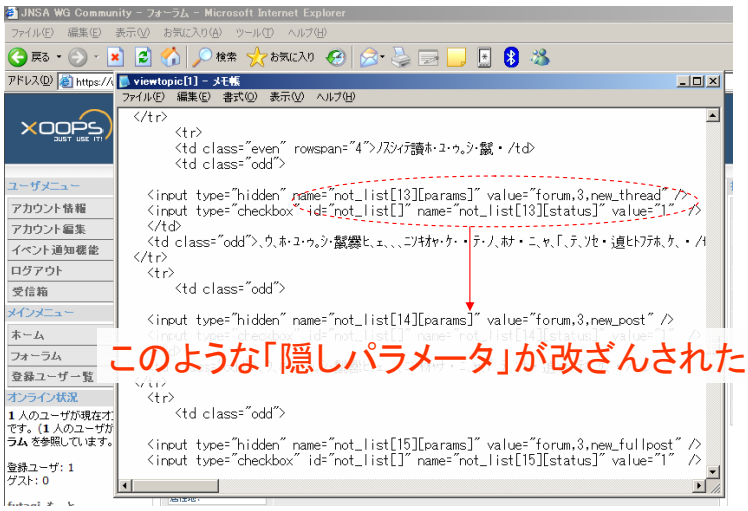
JNSA Web アプリケーションセキュリティWG 2005（抜粋）

よく、対話型のサイトをアクセスした際に、このようなパラメータがブラウザ上に表示されることがある。これらは、たとえば選択されたメニューの番号であったり、処理順序であったりと、アプリケーションの処理を制御するための値だ。このような形のパラメータはユーザが簡単に書き換えられる。つまり、アプリケーションを作る側は、それを前提に、書き換えられても致命的な動作を誘発しないようきちんとチェック等の処理を行っておかなければならない。

たとえば、ある値を9に書き換えたら、本来表示されてはならないはずの管理者画面が表示されてしまった、というような極めて単純な例もある。

アプリケーション攻撃例

- パラメータ改ざん

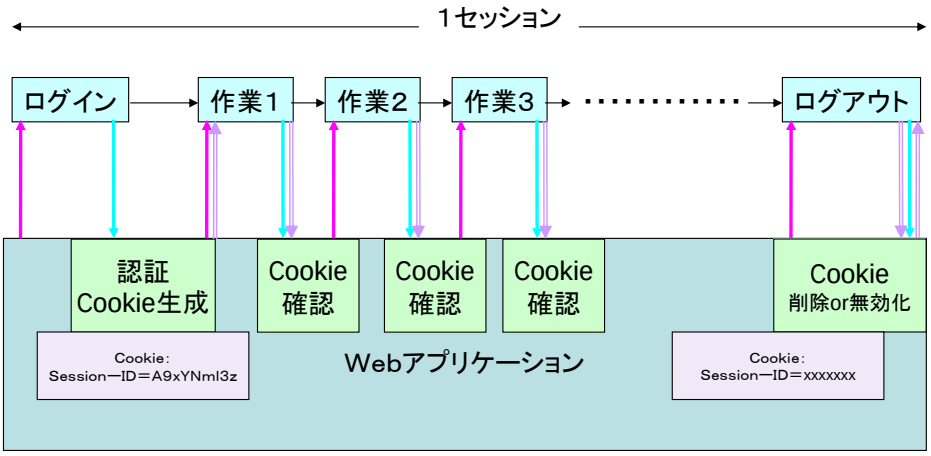


JNSA Web アプリケーションセキュリティWG 2005（抜粋）

もう少し複雑な例がこれだ。アドレスバーには現れないが、表示されているページの中に、パラメータが隠されていることもある。Hiddenフィールドと呼ばれるものや、選択肢のためのパラメータなどである。このような「隠しパラメータ」はフォーム送信時にアプリケーションに送り返され、そのフォームの識別や次の処理のための入力として使われる。ブラウザ上でこれらを書き換えることは困難だが、このような値を表示し、書き換えて送信できるようなツールも簡単に入手できる。

従って、開発者はこのような値すら、書き換えられる前提で仕様を考えなければならない。

Cookieを使ったセッション管理



JNSA Web アプリケーションセキュリティWG 2005（抜粋）

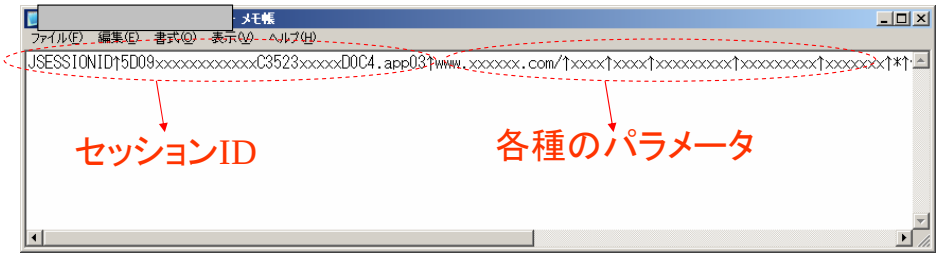
URL引数や隠しパラメータ以外に多用されるのが、Cookie（クッキー）と呼ばれるものだ。これは、Webサーバとブラウザ間で交換されるデータのヘッダと呼ばれる部分に格納されるテキストデータだ。特にこのCookieは、Webアプリケーションにログインしたユーザに認証情報を一連の処理（セッション）の間保持したり、隠しパラメータの受け渡しなどに使われることが多い。

この例では、ユーザが認証を受けてログインし、一連の作業を行ってログアウトするまでのCookieの使われかたを模式的に表している。

このデータもブラウザ上では見ることが困難だが、ツールを使用したり、通信をモニタリングすることで、取得することが可能だ。

Cookie横取り、改ざん、再利用

- Cookieに含まれる内容例

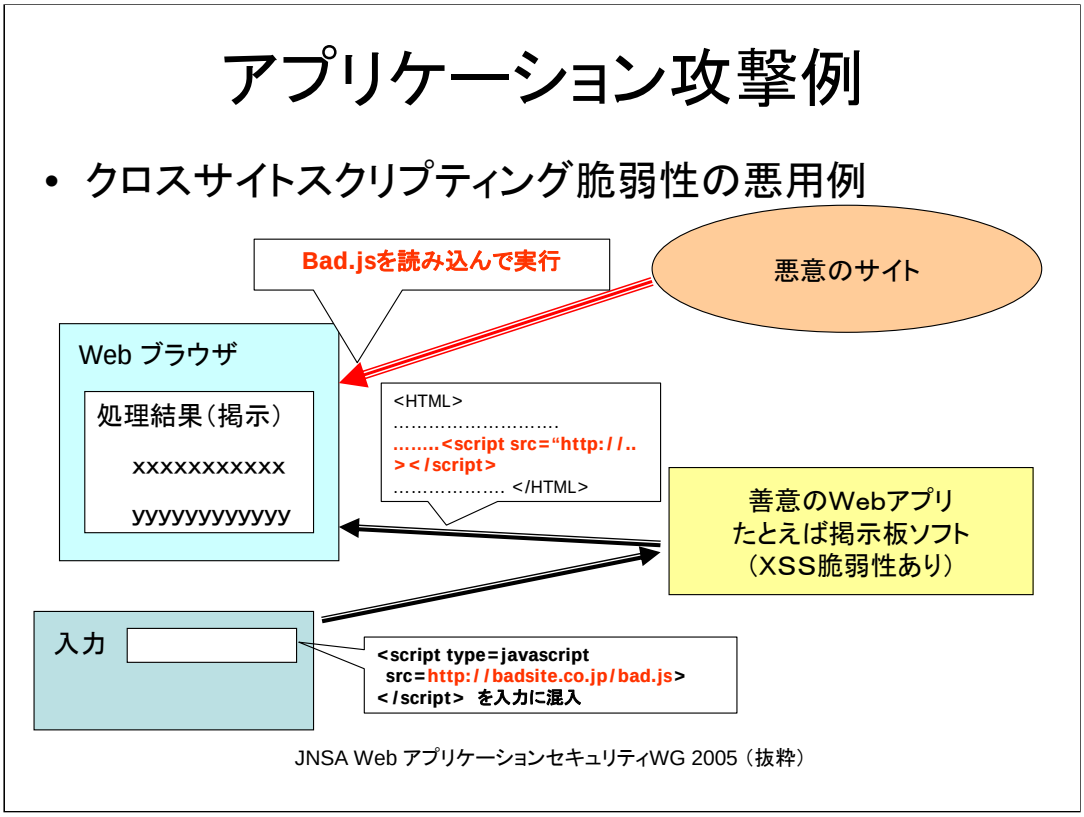


これらを改ざんされたり、横取りして再利用されたら？？

JNSA Web アプリケーションセキュリティWG 2005（抜粋）

たとえばこれは、あるSNSサイトの認証情報を格納したCookieの例だ。（安全上の理由からサイト名、内容はマスクしている）

最も危険なのは、このセッションIDを盗まれて再利用されることで、認証をバイパスしてログイン状態を作り出すことが可能になる点だ。開発者はこうした点を考慮し、経路上で盗まれないようにSSLの使用を強制したり、セッションIDに時刻情報や利用者のIPアドレスなどの情報を含めて暗号化することで、異なるIPアドレスからの利用を防止したり、一定時間で無効になるような仕組みを作り込んでおく必要がある。



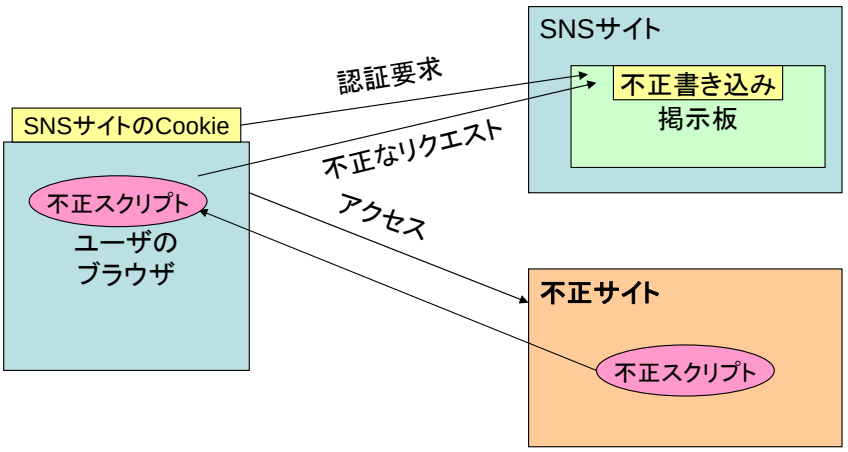
クロスサイトスクリプティングは、最も難解な脆弱性の一つと言われているが、その理由は様々な形で悪用が可能であるからだ。しかし、使われ方の難解さとは裏腹に、その仕組みはきわめて単純である。

これは最も単純な例の一つだ。

たとえば、掲示板ソフトの記事入力部分に、この脆弱性が存在したとしよう。攻撃者は、入力の中に、<Script>タグを混入し、あるサイトから不正な行為を実行するスクリプトを読み込むような仕掛けを作ることができる。このような仕掛けを埋め込まれても、掲示された記事の外見上はまったくわからない。しかし、この記事が表示された時点で、閲覧者のブラウザは不正なスクリプトを実行しているのだ。

アプリケーション攻撃例

- クロスサイト・リクエスト・フォージェリー



JNSA Web アプリケーションセキュリティWG 2005（抜粋）

クロスサイトスクリプティングを応用した攻撃のひとつに、XSRF（Cross Site Request Forgery)がある。これは、クロスサイトスクリプティングで読み込まされた不正なスクリプトが、ブラウザに保存されている特定サイトへの認証用 Cookieを利用して、そのサイトに対して不正なリクエストを行うような連鎖攻撃だ。

たとえば、SNSの掲示板にXSS脆弱性があるような場合、攻撃者は掲示板に不正な script タグを仕込むことで、アクセスしたユーザに不正スクリプトをダウンロードさせ、そのスクリプトによって、アクセスしたユーザの権限でSNSサイトに対して操作を行うことが可能だ。これは、そのユーザが自分のアカウントでSNSにログイン済みであることから、SNSサイトに対するスクリプトからのリクエストが、正当なユーザ認証を経て実行されてしまうためである。

2005年、米国の MySpace.com で実際に行われた攻撃は、こうした不正なスクリプトタグを、ユーザのプロフィール情報に組み込み、それをアクセスしたユーザのプロフィールにも不正タグを伝染させるという、一種のワーム的な動きをするものだった。（SNSワーム事件として話題になった） XSRF攻撃はこのような操作を可能とする。