これだけはやりたいWebシステムのセキュリティ検証

2006年5月12日
加藤 大受

講師紹介

- **加藤大受**
  - ボーランド株式会社、サイボウズ株式会社にてQAマネージャを担当
  - 現在は日立製作所で組み込みRDBMS『HiRDB Embedded Server Entier』の開発を担当
  - 開発ツール、Webアプリケーションの QA、アプリケーション開発、データベース開発など、幅広く経験
  - テストからアプリケーション開発、Web+DB開発に関する記事・書籍執筆を行っている
  - JaSST実行委員
  - JTCB Technical Committee

Webシステムのセキュリティが確保されていないと

- **Webサイトのダウン**
  - サイトへの多大なアクセス
- **Webアプリケーションへの不正アクセス**
  - SQLインジェクション
  - OSコマンドインジェクション
  - クロスサイトスクリプティング（XSS/CSS）
  - セッション・ハイジャック
  - パラメータ改ざん
- **Webサイトへの不正アクセス**
  - 強制ブラウジング
  - パスの乗り越え

ファイアウォールやIDSではほとんど守ることはできない

用語の解説

Webアプリケーションに関する攻撃

- **Buffer Overflow（バッファオーバーフロー）**
  - 確保したメモリ領域を超えてデータが入力された場合に、データがあふれてプログラムが暴走してしまうこと。バッファオーバーランとも呼ばれる。バッファオーバーフロー攻撃とは、バッファに対して許容量を超えるデータを送り付けてシステムを機能停止にしたり、意図的にバッファをオーバーフローさせ、あふれ出たデータを実行させてしまう攻撃。OSで見つかっているセキュリティ・ホールの半数以上がこのセキュリティ・ホール。
- **Cross Site Scripting（クロスサイトスクリプティング）**
  - 悪意を持ったユーザがフォームなどを通してJavaScriptなどのスクリプトコードを入力し、そのスクリプト内容がそのままHTMLに埋め込まれ、ページを閲覧したコンピュータでスクリプトが実行され、Cookieの盗聴や改ざんなどが行われる攻撃。

Webアプリケーションに関する攻撃

- **パラメータ改ざん**
  - 悪意のあるユーザーが、Webアプリケーションの URLパラメータやhidden、Cookieなどを入れた値を書き換えてサーバに送り返す攻撃。これにより価格などの改ざんや他人へのなりすまし攻撃を受ける可能性がある。
- **Backdoor & Debug Options（バックドア・デバッグオプション）**
  - デバッグ用に用意された開発者が正当な手続きなしでシステムにアクセスするための手段で、この裏口からのアクセス方法を利用して悪意のあるユーザーがWebアプリケーションに不正アクセスする攻撃。

Webアプリケーションに関する攻撃

- **Forceful Browsing（強制ブラウジング）**
  - Webサーバーなどの設定の不備などをついて、正当な認証プロセスを通らずに、Webシステムの特定のページにアクセスする攻撃。Cookieの改竄などと組み合わせることで、サーバーの内部情報の取得などが行われることが多い。
- **Session Hijacking（セッション・ハイジャック）**
  - 他のユーザーのSession IDやSession Cookieを盗んで、悪意のあるユーザーがそのユーザーであるかのようになりすまし、Webシステムへアクセスする攻撃。

Webアプリケーションに関する攻撃

- **Path Traversal（パスの乗り越え）**
  - アプリケーションに渡されるパラメータを基にファイルの読み込みを行っているシステムなどで、渡されるパラメータを改竄することで、システム内のパスワードファイルや設定ファイルなどにアクセスする攻撃。
- **SQL Injection（SQLインジェクション）**
  - SQLデータベースを利用しているWebシステムで、テキストボックスやパラメータにSQL文を挿入し、Webアプリケーションへ不正アクセスする攻撃。入力値やパラメータの処理の脆弱性をついた攻撃。

Webアプリケーションに関する攻撃

- **OS Command Injection（OSコマンドの挿入）**
  - Webアプリケーションのパラメータに OSコマンドを挿入し、Webサーバーの脆弱性をついて、Webシステムの不正アクセス、情報の取得、Webシステムのダウンなどを行う攻撃。
- **Client Side Comment（クライアント側コメント）**
  - クライアント側に配信されたWebコンテンツ内のコメント欄に書かれたシステム情報を改竄し、Webアプリケーションへの不正アクセスや情報の改竄を行う攻撃。

セキュリティテストの実施のタイミング

では、セキュリティテストはいつから行うか

~~テストフェーズに入ってからで問題ない~~

Webアプリケーションでのセキュリティ関連の不具合はシステムの変更が必要となる

仕様設計フェーズからレビューを実施する必要あり

Webアプリケーションのセキュリティテスト

- **設計書による確認（レビューの実施）**
  - Cookie、セッションIDの有効範囲の確認
    - セッション・ハイジャックの脆弱性がないか
  - パラメータの取得方法および利用方法の確認
    - パラメータ改竄の脆弱性がないか
  - テキストボックスおよびパラメータをそのままSQL文に挿入していないか
    - SQLインジェクションの脆弱性がないか
  - フォームで送信された値をそのまま変数に代入していないか
    - クロスサイトスクリプティングの脆弱性がないか
  - デバッグ用のアカウントが設定されていないか
    - バックドアの脆弱性がないか
  - クライアントのコメントにシステム情報が書かれていないか
    - クライアントコメントの脆弱性がないか

同一の検証を単体テストでも実施する。コードレビューを実施することで、さらに脆弱性検証を深く実施することができる

Webアプリケーションのセキュリティテスト

- **単体テストによるテストの実施**
  - Cookie、セッションIDの有効範囲の確認
    - セッション・ハイジャックの脆弱性がないか
    - セッションが再利用できないかを確認b
  - パラメータの取得方法および利用方法の確認
    - パラメータ改竄の脆弱性がないか
    - 実際に改竄してアクセスする
  - テキストボックスおよびパラメータをそのままSQL文に挿入していないか
    - SQLインジェクションの脆弱性がないか
    - POSTだけでなくGETについても確認を行う
  - フォームで送信された値をそのまま変数に代入していないか
    - クロスサイトスクリプティングの脆弱性がないか
    - <SCRIPT>タグや<OBJECT>タグで確認する
  - デバッグ用のアカウントが設定されていないか
    - ソースコードを確認
  - クライアントのコメントにシステム情報が書かれていないか
    - クライアントコメントの脆弱性がないか
    - ソースコードを確認

Webアプリケーションのセキュリティテスト

- 統合テストでのセキュリティテスト
  - Cookieに書かれている情報の確認
    - 情報を確認し、変更してのアクセスの実施
  - パラメータの改竄の実施
    - パラメータを変更することで、情報が書き変わったりしないか
  - SQLインジェクションの実施
    - テキストボックスやパラメータ部分にSQL文を挿入して不正アクセスやDBの改竄が起きないかどうか
  - クロスサイトスクリプティングの実施
    - フォーム内にJavaScriptsやActiveScriptsを埋め込んで情報を取得することができるか

表示されるエラーメッセージが正しいかどうかも確認する。可能な限り、テストパターンを洗い出し、テストを実施する。

Webシステムのセキュリティテスト

- Webサーバーおよびシステムの設定の確認
  - サーバー設定などについても検証範囲
    - 未確認で、強制ブラウジングやOSコマンドインジェクションなどの脆弱性が発見される可能性がある
  - OS、ミドルウェアのバージョンの確認
    - セキュリティホールのあるバージョンを利用していないか
    - 一覧を作成し、確認を行う
  - httpd.confなどの設定ファイルの確認
    - ディレクトリインデックスなどが表示されることはないか
    - アクセス権が正しく設定されているか
  - OSのトラスティの確認
    - 不必要なトラスティが設定されていないか

開発しているWebアプリケーションをインストールする前に必ずWebサーバーやミドルウェアの設定を確認する

セキュリティテストは何回行えばいいのか

- **Webアプリケーションのセキュリティテスト**
  - 開発用サーバーとステージングサーバーで2回実施する必要がある
    - ステージングサーバーとは、本番環境と同一のLANの中に存在する最終テストサーバーのこと
- **Webシステムのセキュリティテスト**
  - ステージングサーバーにWebアプリケーションをインストール前とインストール後の2回実施する

開発している会社と運用する会社が異なるときは、必ず開発している会社にステージングサーバーでのWebアプリケーションの検証を行ってもらう必要がある。開発サーバーと本サーバーとの環境の違いによる新たな脆弱性が発見される可能性があるため

運用開始後のセキュリティテストは

- **Webアプリケーションのアップデートのケース**
  - 総合テストで行ったセキュリティテストを実施する
    - コードレビューは必ず行うこと
- **Webシステムのアップデートのケース**
  - ミドルウェアのアップデート時はシステムの設定確認を行う。
  - 負荷分散装置やファイアウォールの更新時はWebアプリケーションへの接続状況をネットワークレベルで確認し、セキュリティに問題ないかを確認する

運用開始後の作業はステージングサーバーで、すべてのテストを行い、問題ないことを確認してから本システムに対して行う。これは負荷分散装置やファイアウォールについても同様である

セキュリティを確保するためには

セキュリティを確保するためには

- **設計フェーズからのレビューの実施**
  - 早いタイミングで脆弱性の問題をなくす
- **手を抜かない**
  - 受け取った値をそのまま代入するようなことはしない
- **テストパターンを洗い出しテストを実施する**
  - 必要となるテストパターンを洗い出しテスト仕様書を作成
- **設定の確認もQA担当者の作業**
  - Webサーバー、ミドルウェアの設定を確認し、脆弱性を防ぐ
- **Webシステムの設定をきちんと資料化する**
  - 資料がなければ脆弱性の検証はできない
- **ステージングサーバーで最終確認**
  - 開発用のサーバーだけでなく、ステージングサーバーで最終確認を実施

ご静聴ありがとうございました