DESIGN
ENGINEERING

「これだけは押さえたい！」

現場で役立つ Webセキュリティ 基本用語

アプリやシステムの**脆弱性**は、
設計やプログラムのミス、設定不備、古い部品（ソフトウェア・ライブラリ）の使用など、
さまざまな理由で生まれます。

こうした脆弱性を放置すると、
個人情報の漏えいやシステムの不正利用など深刻な被害につながるため、
開発や運用の際には十分な注意と対策が必要です。

エンジニアになるなら、
「よくある攻撃や脆弱性」とその対応を
知っておくことが大切です。

ここでは現場で役立つWebセキュリティ基本用語として、
「XSS」「CSRF」「SQLインジェクション」など
有名どころの攻撃原理や手法を解説します。

XSS（クロスサイトスクリプティング）Cross-Site Scripting(CSSと被るからXSS)

XSSの仕組み
攻撃者が**悪意あるスクリプト**を入力フォームやリンクに仕込む
→ユーザーが該当ページを閲覧・クリックする
→ブラウザ上でスクリプトが実行され、情報流出や不正操作が行われる

主な種類

- 反射型XSS：リンク経由で仕込まれたスクリプトが即座に実行される。
- 格納型（持続型）XSS：スクリプトがサーバに保存され、アクセスするたびに利用者全員に影響。
- DOM Based XSS：サーバではなく、ブラウザ側（JavaScriptの処理）を利用して攻撃。

不正スクリプトを実行させる攻撃

XSS（クロスサイトスクリプティング）Cross-Site Scripting(CSSと被るからXSS)

想定される被害

- 正規サイト上に偽情報を表示
- 個人情報やCookie（ID・パスワード・閲覧履歴など）の窃取
- クレジットカード不正利用やネットバンキング被害
- 不正プログラムのダウンロードによるマルウェア感染

有効な対策

- サニタイジング（無害化処理／**エスケープ処理**）による危険な文字の変換
- 入力値の制限（利用可能な文字種や桁数を明確化）
- URL制御（http/https のみ許可、javascriptスキームを拒否）

CSRF（クロスサイトリクエスト偽造フォージェリ）シーサーフとも呼ばれる

「CSRF」は、ユーザーがログイン中のWebアプリケーションに対し、攻撃者が用意した不正なリクエストをユーザーの意図に反して送信させてしまう脆弱性を利用した攻撃

攻撃者はユーザーを罠サイトに誘導し、不正リクエストを送信させることで、Webサイトが「ログイン中のユーザーからの正規リクエスト」と誤認し、不正な処理を実行

被害には、不正な送金、意図しない投稿、パスワード変更などがある

正規のユーザーのリクエスト（命令）偽造

CSRF（クロスサイトリクエストフォージェリ）シーサーフとも呼ばれる

CSRF攻撃に対する対策：ユーザー側ではログアウトの徹底や不審なリンクへのアクセス回避

サービス提供者側の対策

- **CSRFトークン**を生成・検証し、リクエストの照合を強化する
- セッションIDと機密情報（hiddenパラメーター）を両方使い、リクエストの正当性を判断する
- Refererヘッダをチェックし、正しいリンク元からのリクエストかを確認する
- 重要な処理の前にパスワード認証を求める
- 定期的な脆弱性診断を実施し、リスクを早期に発見・修正する
- CAPTCHA（画像認証）などを導入し、不正リクエストを防ぐ
- 信頼できるフレームワークやライブラリを使用して対策を組み込む

SQLインジェクション

Webの入力欄などに**不正なSQL文を注入**し、**データベースを不正操作**する攻撃手法

仕組み

- SQL文はユーザーの入力に基づき生成され、データベースを操作する命令文です。例えば、IDを条件にデータを抽出します。
- 攻撃者は入力欄に「' or '1'='1」などの不正文字列を挿入し、SQL文の条件を操作。これにより**本来の条件を無効化し、全データの抽出などが可能**となります。

被害例

・企業の非公開情報や個人情報の漏えい（顧客データ、クレジットカード情報など）。
・Webサイトの改ざんや削除、コンピュータウイルスの埋め込み。

SQLインジェクション

対策方法
・プレースホルダ（パラメータ化クエリ）を使用して、ユーザー入力をSQL文の一部として扱わず、安全な値として処理する。
・入力データに対するエスケープ処理で、特殊文字を無害化する。

その他の押さえておくべきワード

・セッションハイジャック

ユーザーのセッションIDを盗み、それを使って本人になりすまし不正アクセスを行う攻撃です。
これにより、権限を持つユーザーとしてシステムを操作されるリスクがあります。

対策：

- セッションIDをURLに含めない（Cookieで管理）
- 推測困難な長く複雑なセッションIDを使う
- HTTPS通信で暗号化し、盗聴を防ぐ

その他の押さえておくべきワード

・ディレクトリ・トラバーサル

Webアプリケーションのファイルパス指定の脆弱性を悪用し、本来アクセスできないサーバー内のファイルやディレクトリに不正アクセスする攻撃です。攻撃者は「../」などの相対パスを使い、機密ファイルの閲覧や改ざんを行う恐れがあります。

対策：

- 入力値の検証と正規化を徹底し、不正なパス遷移を防止
- 絶対パス利用やホワイトリスト方式で参照先を制限

その他の押さえておくべきワード

・CORS（クロスオリジンリソースシェアリング）コルスとも呼ばれる

異なるオリジン（ドメイン）間で安全にリソースを共有するための仕組みです。設定が不適切だと、本来アクセスできない外部サイトからデータが盗まれるなどのリスクがあります。

対策：

- Access-Control-Allow-Originヘッダーを適切に設定（信頼できるドメインのみに制限）
- プリフライトリクエストの適切な処理
- セキュリティヘッダーの活用（例：Content-Security-Policy）
- CORS設定は最小限の許可にとどめる