第三章:クライアントサイドのインジェクション

クライアントサイドのインジェクションおよびコードへの影響について学ぶ

クライアントサイドのインジェクションとは？

JavaScriptの場合

概要

クライアントサイドのインジェクションは、信頼性のないソースからのデータが十分にサニタイズされずに、Javascriptの**eval()**関数を使用して、そのまま解釈された場合に発生します。

攻撃

脆弱な例について、順を追って説明しましょう。

例として、航空会社の乗客のプロフィール写真を参照すると、その乗客のユーザプロフィールが表示されるというWebアプリケーションがあるとします。プロフィールには、選択したユーザの利用ステイタスが表示されますが、データのユーザ入力は十分にサニタイズされていません。

プロフィールデータにアクセスすると:

https://airlinecarrier.com/api/users/update/profiledata.json

次のようなレスポンスが返るとします:

```JavaScript
{
"Benefits": "ステイタス",
"Level": "ブロンズ"
}

var data = eval("(" + resp + ")");
```

```
document.getElementById("#Benefits").innerText = data.Benefits;

document.getElementById("#Level").innerText = data.Level;
```

データは、Jsonの**eval()**関数を使用して読み込まれ(解釈され)て、挿入されます。

この欠陥を利用して、攻撃者は以下のコードを入れ込むことで、クライアントサイドのインジェクション攻撃を仕掛けることができます:

```
JavaScript
プラチナ."});alert(1);({"Benefits":"ステイタス","Level":"プラチナ".
```

この引数が**eval()**関数で実行されると、新しい出力は以下のようになります:

```
JavaScript
{"Benefits": "ステイタス","Level": "プラチナ."});alert(1);({"Benefits":"ステイタス
","Level":"プラチナ"}
```

これで、ユーザによってこの航空会社のステイタスレベルが引き上げられたことになります。

影響

攻撃者がこの脆弱性を利用して、別のユーザに代わって、意図しない操作を処理する可能性があります。このような脆弱性は、クロスサイトスクリプティング(XSS)などの他の危険な攻撃につながる可能性もあります。

対策方法

JSONのインジェクションを防ぐ最も効果的な方法は、信頼性のないソースからのデータを含む文字列がJSONとして解釈されないようにすることです。

また、JSONデータの評価に**eval()**関数は使用せずに、代わりに**JSON.parse()**を使用してJSONのレスポンスデータが安全に解釈されるようにしてください。

おつかれさまでした！

クライアントサイドのインジェクションとは何であるか、そしてお使いのシステムでそれにどう対策すべきかが、お分かり頂けたと思います。ここで学んだ新しい知識をうまく応用しながら、コーディングをしてください。これをあなたのネットワークで自由に共有してください。また、一般的な脆弱性に関する弊社のその他のレッスンも是非ご覧ください。
この学習モジュールに関して修正などがある場合は、こちらをクリックして、プルリクエストを作成してください！

関連記事:

ブログ:The Top 10 app-attack trends in the financial sector in 2022

ブログ:Find JavaScript cyber-vulnerabilities for free with CodeSec