ファイルのアップロード処理

アップロードに関数する問題

アップロードに対する攻撃

1.アップロード機能に対する**DoS攻撃**

大量のファイルをアップロードすることで、Webサイトへ負荷をかける。

・対策（PHPの場合）

php.iniファイルの設定で容量の制限を行う。

upload_max_filesize(ファイルの最大容量：2MB)、**max_file_uploads**(送信できるファイル数の上限：20)などの設定を変更する。

Webサーバなどでも制限を行うことができる。<例>Apache・・・LimitRequestBodyなど。

2.アップロードされたファイルをサーバ上**でスクリプト**として実行する。

Webshellが該当する。**OSコマンド・インジェクション**と同じ影響がある。

3.仕掛けを含むファイルを利用者にダウンロードされる。

クライアント側でのスクリプトの実行、マルウェアに感染させる。

4.閲覧権限のないファイルのダウンロード

URLの推測によりファイルがダウンロードできることがある。

サーバ上でのスクリプト実行

スクリプトのファイルの拡張子・・・php、asp、aspx、jspなど。

スクリプトファイルの実行

↓

OSコマンド・インジェクションと同じ影響を受ける。

ファイルの閲覧、改ざん、削除

外部へのメール送信

別のサーバへの踏み台

サーバ上での暗号通貨の採掘(マイニング)

対策

アップロードされたファイルは**公開ディレクトリに置かず**、アプリケーション経由で閲覧させる。

ファイルの拡張子をスクリプト実行の可能性のないものに制限する。

＊webshell

Webバックドアと呼ばれる**任意のコマンドをサーバ上で実行**させるプログラム。ファイルのアップロードや削除など実行することができる。

＜例＞**<?php** echo shell_exec($_GET['cmd'].' 2>&1'); **?>**

原因

公開されているディレクトリに保存される。**スクリプトを示す拡張子**をアップロードできる。

＊XSS

PDFの中にHTMLタグが含まれていると条件によっていは**ブラウザがHTMLと認識する**ことができる。

↓

JavaScriptが埋め込んでいる。

↓

XSSが実行される。

対策

Contet-Typeを正しく設定

レスポンスヘッダX-Content-Type-Options:nosniffを設定する。

レスポンスヘッダとしてContent-Dispositionを指定する。