Unrestricted File Upload

Introduction

This scenario involves **absolutely no restrictions on file uploads**. Neither the frontend nor the backend performs any validation. **Users can upload any file** - regardless of its extension, content type, or size.

Such a configuration is a **major vulnerability** and can lead to:

- **uploading malicious server-side scripts** (e.g. `PHP`, `JSP`, `Python`) - potentially enabling remote code execution, hence often full control over server.
- **denial of Service (DoS)** - uploading extremely large files that exhaust storage or processing resources.
- **corrupting server-side data integrity** - uploading unexpected or improperly handled file types.
- **bypassing application logic** - since the server implicitly trusts the uploaded content.

Examples

- reading files from server's filesystem

```php
<?php echo file_get_contents('/path/to/target/file'); ?>
```

- enabling `GET` request to perform payload specified in `command` parameter

```php
<?php echo system($_GET['command']); ?>
```

LAB

goal: explore vulnerable file upload function with near zero validation

requirements: Internet connection, BurpSuit, PortSwigger account

LAB: link

↓ **hints below** ↓

Hints

hint 1: find unrestricted file upload function and try to exploit it.

hint 2: based on example scripts above, prepare payload file and upload it.

hint 3: using `Proxy` | `HTTP history` find response to Your `GET` request generated by uploading malicious file.