

Unrestricted File Upload of Malicious Image File Leading to potential RCE

WOCS'Hack 2025

Submitted by **DirtyBst3rd** on **2025-04-27**

REPORT DETAILS

9.6 **CRITICAL**

CVSS

Bug type Unrestricted Upload of File with Dangerous Type (CWE-434)

CVE ID**Impact**

Scope *.3xploit.me

Endpoint ttps://75ebd1701299.3xploit.me/uploads/

Severity Critical

CVSS vector string CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:H/I:H/A:H

Vulnerable part others

Part name Profile Picture Upload

Payload The payload can be described as an image file (jinx_with_php.png) containing malicious PHP code or a PHP file disguised as an image file.

Technical env. Firefox ,xxd

App. fingerprint

BUG DESCRIPTION

DESCRIPTION

The vulnerability discovered is related to the improper handling of image file uploads, which allows for the upload of files that could potentially execute malicious code if certain server configurations change in the future. This issue occurs within the profile picture upload functionality of the application, where it is possible to upload files with unexpected content types (e.g., PHP code disguised as a PNG image). Although the system currently does not execute any code due to error handling, this could present a risk if server configurations change (e.g., allowing PHP execution in the /uploads/ directory).

EXPLOITATION

Navigate to the Profile Page or the section where users can upload a profile picture.

Upload a valid image file, such as jinx.png, and modify the file header to inject PHP code while preserving the image's validity.

The server accepts the file, and it's stored in the /uploads/ directory with a valid image MIME type.

When accessing the file via its URL (e.g., https://example.com/uploads/jinx_with_php.png), the image is displayed because the file is still recognized as a valid image. However, the injected PHP code is still present.

Attempt to trigger the execution of the PHP code by manipulating the query string of the URL, such as appending ?cmd=phpinfo(). The system currently blocks execution (e.g., by displaying an error message like "invalid image"), but the issue may be exploitable if configurations change.

POC

Uploaded Image: jinx_with_php.png
Steps:

Modify a valid image (e.g., jinx.png) by injecting PHP code into the image’s header.
Upload this modified image as a profile picture.
The image is successfully uploaded and displayed, but the PHP code is still embedded in the file.
Try to execute the PHP code via a manipulated URL (e.g., https://example.com/uploads/jinx_with_php.png?cmd=phpinfo()), which could be executed if server settings allowed PHP execution.

RISK


This vulnerability poses a medium risk to the application as it could allow attackers to upload files that contain malicious PHP code, which could potentially be executed if server configurations change or if the file handling is modified. Although the code is not executed in the current configuration, the vulnerability opens a door to future exploitation. Moreover, it can be exploited for storing malicious code on the server, leading to possible further attacks, such as remote code execution (RCE), if other vulnerabilities are present.

REMEDIATION

Ensure that the application only allows a strict list of image file types (e.g., image/png, image/jpeg) and rejects any files that do not conform to this list. You can verify file types based on their MIME type and not just the file extension.

Sanitize file uploads
This prevents malicious files from being uploaded in the first place. One method to further protect the application is to sanitize the content of the image file to ensure that even if the file is a valid image, it doesn't contain hidden PHP code or malicious content. This could be done by re-encoding the image or by stripping out any potential executable code that may have been embedded in the file.

COMMENTS




DirtyBst3rd

on 2025-04-27 13:41:01

Everyone



New




WOCSA

on 2025-04-27 13:43:32

Everyone



New → Under Review




WOCSA

on 2025-04-27 13:43:39

Everyone




Under Review → Accepted



WOCSA

on 2025-04-27 13:43:42

Everyone



5 pts awarded for report quality.