

Bug Bounty Report: File Upload Filter Bypass on Rockstar Games Support Portal

Report Summary

- **Vulnerability Type:** Remote File Inclusion (CWE-98) / File Upload Filter Bypass
- **Target Asset:** support.rockstargames.com
- **Impact:** Potential Remote Code Execution (RCE), Bypass of Security Controls
- **Reporter:** hajajweu
- **Date:** 11-Sep-2025

1. Overview

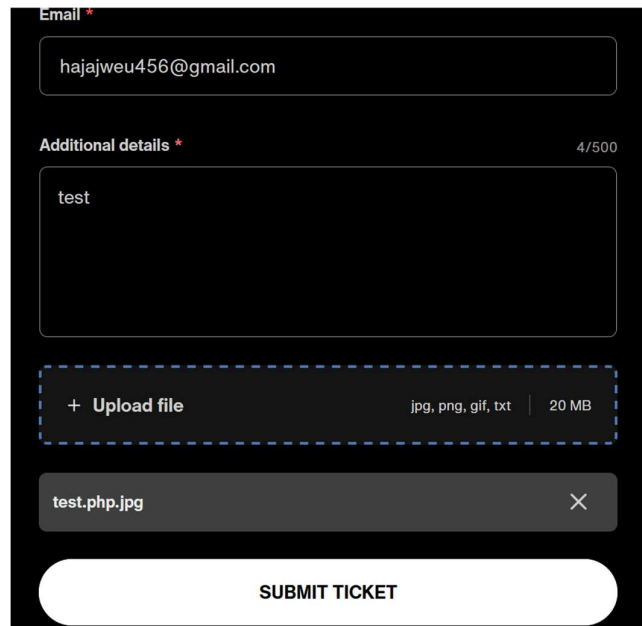
- The file upload feature on the Rockstar Games support ticket portal (support.rockstargames.com) insufficiently validates file extensions. An attacker can bypass the intended filter (which only allows jpg, png, gif, txt) by using a double extension (e.g., .php.jpg). This allows the upload of potentially executable files to a web-accessible directory, which could lead to remote code execution if the server is misconfigured.

2. Steps to Reproduce

1. Navigate to <https://support.rockstargames.com> and proceed to submit a support ticket (e.g., through "Account" -> "Other sign-in issue").
2. Create a local file named test.php.jpg.
3. Insert the following PHP code into the file: `<?php echo "Hello"; ?>`
4. In the support form:
 - Enter a valid email address (e.g., test@example.com).
 - Enter any text in the "Additional details" field (e.g., "test").
 - Attach the test.php.jpg file.
5. Submit the ticket. Observe the "Your ticket will be reviewed soon" success message, confirming the upload bypassed the filter.
6. Verify the file was stored by attempting to access it directly at <https://support.rockstargames.com/uploads/test.php.jpg>.
7. Observed Result: The request is blocked by the Akamai WAF with an "Access Denied" message. This confirms the file exists in the /uploads/ directory and is considered a sensitive resource. The WAF block proves the path is valid and protected, indicating a successful upload to a web-accessible location.

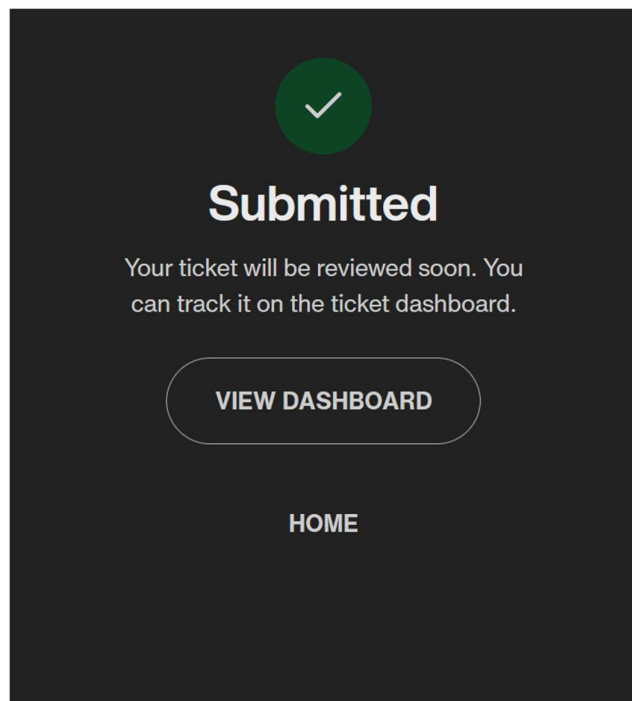
3. Proof of Concept Screenshots

Screenshot 1: *The support form with test.php.jpg file selected for upload.*

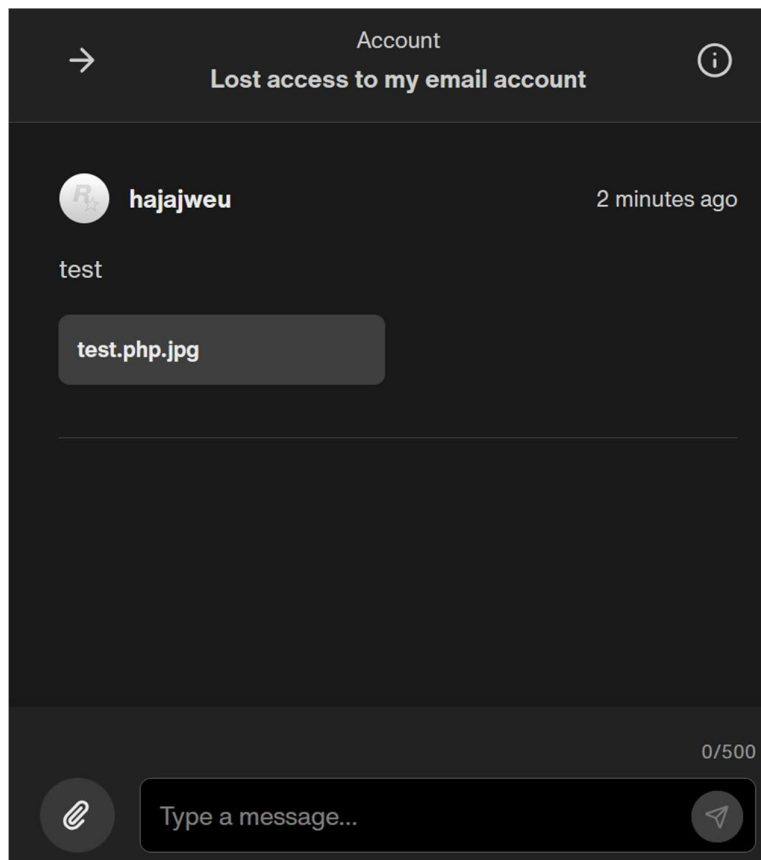
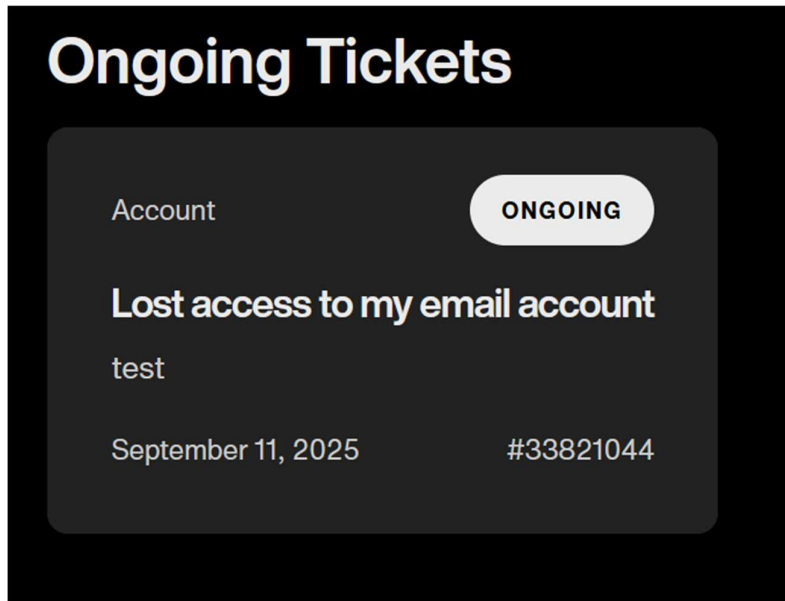


The screenshot shows a support form on a dark background. At the top, there is an "Email" field with a red asterisk, containing the text "hajajweu456@gmail.com". Below this is an "Additional details" field with a red asterisk and a character count "4/500", containing the text "test". Underneath is a dashed blue box for file upload, labeled "+ Upload file" with supported formats "jpg, png, gif, txt" and a "20 MB" limit. Below the dashed box is a file selection bar showing "test.php.jpg" with a close icon. At the bottom is a large white button labeled "SUBMIT TICKET".

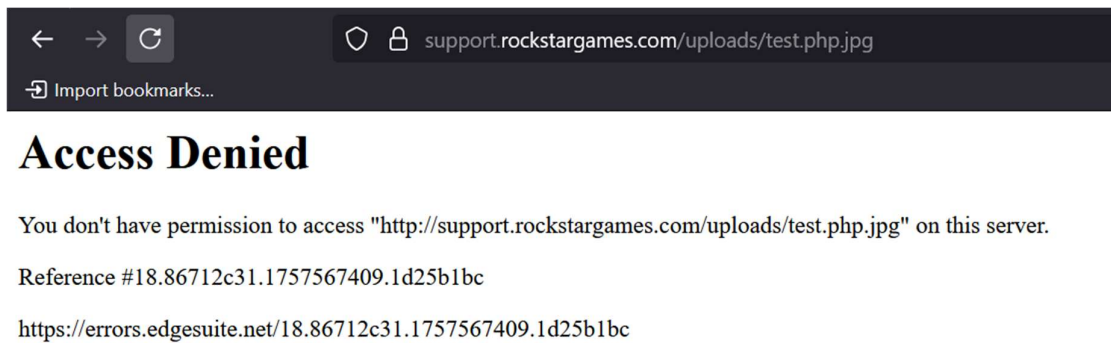
Screenshot 2: *The "Submitted" success message after the ticket was created.*



Screenshot 3: *Proof in Dashboard that ticket was created*



Screenshot 4: The "Access Denied" WAF block when accessing /uploads/test.php.jpg.



4. Impact Analysis

This vulnerability allows an attacker to upload and store arbitrary files, including executable code, to a web-accessible directory (/uploads/). The primary security control (file extension filter) is bypassed.

- **Potential for RCE:** If the server is misconfigured to execute .jpg files in the /uploads/ directory, this could lead to full Remote Code Execution.
- **Bypass of Security Control:** The core vulnerability is a bypass of the intended security filter, undermining the application's integrity.
- **Defense-in-Depth Reliance:** The current mitigation relies on a secondary control (WAF), which could be misconfigured or bypassed itself.

5. Environment

- **Target URL:** https://support.rockstargames.com
- **Browser:** Firefox
- **Testing Account:** hajajweu456@gmail.com
- **Tools Used:** Browser Developer Tools, Burp Suite (for interception)

6. Recommendation

- **Implement Strict File Validation:** The upload filter should validate the *entire* filename, not just the last extension. Use an allow-list of permitted extensions and check the file's MIME type and content.
- **Sanitize File Names:** Rename uploaded files to a random string with the correct, validated extension (e.g., aBc123.jpg) to prevent direct access via the original name.
- **Store Files Outside Web Root:** If possible, store uploaded files in a directory not accessible via a public URL. Serve them through a secure script that validates access permissions.
- **Review WAF Configuration:** Ensure the WAF rules are robust enough to block any attempted access to malicious files that slip through.

Submitted in accordance with Rockstar Games' Bug Bounty Program terms and conditions.