SecureLayer7

Time and Again, Securing you

Real Media Library WordPress Plugin

Vulnerability Assessment and Penetration Testing Report

Table of Contents

Executive Summary

SecureLayer7 is pleased to present the results of the Penetration Testing Assessment conducted on the Real Media Library: A WordPress Plugin and service from 1th January 2023 to 31st January 2023. The assessment was conducted by a team of 1 Penetration Tester, 1 Team Lead, and 1 Project Manager. The assessment was performed remotely, and the methodology and approach used for the assessment was White Box Penetration Testing.

The assessment began with the Penetration Testers and Team Lead working together to identify potential vulnerabilities in the application and service. This was done by utilizing a combination of automated tools and manual testing methods. Once potential vulnerabilities were identified, they were analyzed and evaluated by the Penetration Testers and Team Lead to determine the impact and likelihood of exploitation.

The Project Manager was responsible for managing the overall project plan, monitoring, and controlling project documentation, and providing high-quality deliverables. The Project Manager also worked closely with the Kimai Time Tracking team to ensure that the assessment was conducted in a manner that was both thorough and non-disruptive to their daily operations.

The results of the assessment were compiled into a comprehensive report that includes an executive summary, detailed descriptions of the vulnerabilities identified, and recommendations for remediation. The report also includes detailed information on the methodology and approach used for the assessment, as well as the qualifications and experience of the SecureLayer7 team members who conducted the assessment.

It is important to note that the findings in this report reflect the conditions found during the assessment and do not necessarily reflect current conditions. The security test results and findings provided in this report are valid for the period during which the assessment was carried out and are based on the information provided for the assessment.

SecureLayer7 would like to thank the Real Media Library team for their cooperation and support during the assessment. We are confident that the information provided in this report will be valuable in helping the Real Media Library: A WordPress Plugin team to improve the security posture of their application and service. We stand ready to assist in any way possible as the WordPress Plugin team works to address the issues identified during the assessment.

Scope

Scope Details
White-Box Tests against Real Media Library Plugin https://wordpress.org/plugins/real-media-library-lite/

Penetration Test Details:

Activity Date(s)	1st January 2023 – 31st January 2023

Coverage

White-Box Tests against Real Media Library WordPress Plugin

- The Real Media Library WordPress plugin has file upload functionality. When tested for unrestricted file upload bypass using the double extension technique, the application responded for the uploaded file as the **File type is not valid** error message. Hence the upload functionality is secured.
- The Real Media Library WordPress plugin is tested against the directory listing vulnerability for sensitive information disclosure through the hidden or outdated files on the application server. But it was observed that the application gives a 404 error message for the backup and database files. When trying to bypass the 404-error message using the null byte technique to access the files, the application checks for the null byte payload and has implemented the proper security.
- The Real Media Library WordPress plugin is tested against unauthenticated access by removing the authentication tokens for creating folders and uploading file functionalities. It was observed that the application responded with a 302 found message and redirected to the login page. Hence the plugin is secured.
- The Real Media Library WordPress plugin was tested for Time-based SQL injection vulnerability by supplying time-based NoSQL injection payload to the 'item_id' parameter in the save request. It was observed that there was no time delay in the response after executing the payload. Hence the application is secured.
- The Real Media Library WordPress plugin was tested for client-side template injection by supplying the mathematical payloads such as {{4*4}} to the 'folder' and 'title' user input fields. It was observed that the application did not process the payload and considered it plain text. Hence the plugin is secured.
- The Real Media Library WordPress plugin was tested for business logic issues such as the number of times a user can access the functionality (i.e., rate limiting), start/end date configuration, and input type validation. But it was observed that the application implemented a proper business logic mechanism for rate limiting and data type validation. Hence the plugin has security against business logic issues.
- The Real Media Library WordPress plugin was tested for open redirection vulnerability on the 'redirect_uri' parameter for navigational HTTP requests by supplying the unknown or malicious host to the parameter. The plugin was observed to not redirect to the provided malicious host. Hence the next parameter in the given URL is secured against the open redirection issue.
- The Real Media Library WordPress plugin is tested against PHP object injection by supplying the phpinfo() payload in the serialized object code of the plugin. But it was observed that the plugin could not bypass the unsterilized() function due to proper input sanitization implementation. Hence the plugin is secured.
- The Real Media Library WordPress plugin was tested for host header injection attack by adding the 'X-Forwarded-Host' header in the request. But it was observed that the application redirects to the original host after manipulating the request for host header injection. Hence the application is secured.
- The application was tested for Cross-Site Scripting (XSS) vulnerability on the "File name" parameter of the "New Media Upload" module by introducing XSS payloads in the "file name" field. It was observed that the application sanitizes the user input correctly and has implemented server-side validation. Hence the application is secured.
- The application was tested for SQL injection vulnerability on the "Settings" module by adding a quick entry, injecting time-based SQL injection on all parameters, and observing the response time. After injecting the time-based SQL injection payload, no delay was observed in response time, so the application is not vulnerable to SQL injection.
- The application was tested for Path Traversal vulnerability on the "Edit Post" module inside the POST parameter to check whether accessing internal files is possible. Still, the application redirects to the edit.php page. So the application is secured for a given attack scenario.
- The application was tested for Server-Side Template Injection (SSTI) on the "Edit Post" module by introducing the SSTI payload in all parameters. This attack aimed to check and observe whether the application backend server renders the payload supplied. But the server treats the payload as a string and does not execute it. Hence, the application is not vulnerable to SSTI.

Identified Vulnerabilities

The following sections list both vulnerabilities and implementation issues spotted during the testing period. Note that findings are listed in chronological order rather than by their degree of severity and impact. The aforementioned severity rank is simply given in brackets following the title heading for each vulnerability. Each vulnerability is additionally given a unique identifier (e.g., RML-01-001) for the purpose of facilitating any future follow-up correspondence.

1. RML-01-001: Stored Cross-Site Scripting in Folder Name under File Upload Module (*Medium*)

The WordPress Plugin WordPress Real Media Library did not implement any input sanitization checks on both the client and server sides. It is prone to a stored cross-site scripting vulnerability because it fails to sanitize user-supplied input properly. The Folder Name reflects without any encoding on the File Upload Page. After successfully uploading the file, Folder Name was displayed on the Upload page without sanitization. Further analysis revealed that attackers could inject malicious JavaScript code into these user-controllable input fields that would later become permanently embedded into the web page. Thus eventually triggering the malicious script whenever interacting with the affected page.

Affected Module:

- *Folder Name in File Upload Page*

Affected File:

Below is the sample of the affected source code file demonstrating the missing sanitization on user-controlled input.

- *\real-media-library-lite\inc\view\View.php*
- *\real-media-library-lite\public\dist\rml.lite.js.map*

Affected Code:

```
[...]
$name = empty($options['name']) ? '' : 'name="' . $options['name'] . '"';
     return '<input type="hidden" value="' .
      esc_attr($options['selected']) .'" ' .$name .
<div class=\"media-item-rml-folder\">' + title + \"</div>\";
[...]
```

Mitigation:

It is recommended to sanitize user input using the **esc_attr()** WordPress function whenever it requires displaying the input to the user.

2. RML-01-002: Cross-Site Scripting in Alt Text under Folder Details Cover Image (*Low*)

The WordPress Plugin WordPress Real Media Library did not implement any input sanitization checks on both the client and server sides. It is prone to a cross-site scripting vulnerability because it fails to sanitize user-supplied input properly. The **Alt Text** reflects without any encoding in the Folder Cover Image Section. After successfully uploading the file, the user can edit "**Alt Text**," which gets displayed in Folder Details Cover Image without sanitization that executed a malicious script.

Affected Module:

- *Folder Cover Image Alt Txt*

Affected File:

Below is the sample of the affected source code file demonstrating the missing sanitization on user-controlled input.

- *real-media-library-lite\public\lib\wp-media-picker\wp-media-picker.min.js*

Affected Code:

The third party library was concatenating the string instead of escaping it

```
[...]
 if ( src ) {
   preview_content += '<img src="' + src + '" alt="' + attachment.alt + '"
/>';
}  else {
   preview_content += '<div class="mime-type-icon"><img src="' +
attachment.icon + '" /><span>' + attachment.filename + '</span></div>';
}
[...]
```

Mitigation:

It is recommended to sanitize user input using the esc_attr() WordPress function whenever it requires displaying the input to the user.

Conclusion

Summarizing the attack narrative, SecureLayer7 used the Burp Proxy Suite of tools to record, intercept, and replay the requests to the application. Once the application had been mapped out with general requests, Burp was also used to identify the classic vulnerabilities.

As for the approach chosen for the assessment, the involved parties agreed upon the benefits of a Grey Box methodology. This means that the SecureLayer7 team was provided full access to all relevant infrastructure. With this focus and scope in mind, SecureLayer7 conducted extensive tests against Real Media Library Plugin Web application running on these. It should also be noted that the parts of the software involving web front ends were specifically tested and checked for Unrestricted File Upload, Directory Listing, Unauthenticated Access, Time-Based SQL Injection, Client-Side Template Injection, Rate Limiting, Open Redirection, PHP Object Injection, Host Header Injection, Cross-Site Scripting (XSS), SQL injection, Path Traversal, Server-Side Template Injection (SSTI), Cross-Site Scripting (XSS), and similarly dangerous attacks.

The pentest activities began the engagement by using the application the same way a typical user would. This exercise was proxied through the Burp Suite proxy and included passive analysis for identification of any misconfigurations and various different vulnerabilities. By visiting each page, SecureLayer7 created a record of the Application in the Burp Suite proxy for use with manual and automated testing techniques later in the engagement.