

What this report contains

Executive Summary

This section provides a non-technical overview of the scope of the project, the key findings from the testing work carried out and highlights the areas we see as most important or urgent in needing attention to improve the security of the system under test.

Target Information

This lists what was tested and the type of testing carried out.

Methodology

This is a short, technical description, of physical way that the testing was undertaken.

Issue Summary

This is a high level, technical, summary of each issue found. It's common for this section to contain multiple examples of the same type of issue as we list every instance of an issue we find so that there is a clear list of everything that needs to be fixed – i.e. If we find a form that is insecure it's possible that each individual field within the form has the same type of issue and so if the form has five fields; First name, Last name, Email address, Company name and comments these would be listed as 5 issues as each field needs attention.

Vulnerability by Severity

This is a visual representation of the different levels of issues found to provide a quick overview of the number of each type individually and in relation to the application as a whole.

Details of Individual Issues

This is a detailed list of each issue found containing, where possible:

- Affected assets – what in the system has the issues
- Severity – on a 5-point scale from 'Critical' to 'For information only'
- OWASP / CWE mapping – how the issue relates to security testing methodologies or databases
- Technical description – a detailed technical explanation of the issue
- Steps to reproduce – how to verify the issue exists and to check that it has been fixed
- Business impact – a non-technical description of the potential impact of the issue
- Solution and/or mitigation – what's required to fix the issue
- External technical reference links – additional information in the public domain to provide more insight into the type of issue, other ways to fix it etc.

As with the issue summary, every instance of an issue is listed individually to provide a detailed, step-by-step guide for all fixes required.

Copyright © Appsecco



Executive Summary

Appsecco was contracted by Law Firm Services to conduct Web Application Security Testing to determine if there were security weaknesses in the LFS Hemmings and Walker application, the implementation of the Identity Server, the backend LFS API server and the tenant applications that can render the environment insecure and allow an attacker to gain access to any data that is accessible via them or gain access to the underlying operating system.

The testing was carried out between 10th February 2020 to 14th February 2020 on the testing setup of the LFS Hemmings and Walker application, LFS Identity Server and the associated API servers and web applications.

OWASP Top 10 2013, OWASP Top 10 2017 and CWE were used as the reference framework to evaluate and categorise the discovered security issues.

Common Vulnerability Scoring System (CVSS) 3.0 standard has been used to determine the severity of the discovered issues.

Approach

Our testing approach entailed using a demo implementation of the environment with two tenants created on the server and activated by following an email containing an Opt-In link. The application environment uses API endpoints to authorize, retrieve and update data while a standard Angular based web application is used to update the UI retrieved by backend JS requests.

Once the setup was completed and access obtained, the following testing approach was used along with the standard approach to testing for the OWASP Top 10 2013, OWASP Top 10 2017 and API testing:

1. The application was tested for cross-tenant access using tampered tokens, tampered host headers and tampering of various GET and POST parameters
2. As the environment uses an IdentityServer for the authentication and authorization, an attempt was made to check for token expiration and revocation and usage after the token is supposedly not valid
3. As the application uses IdentityServer, we have tested for common mis-configurations such as Open redirection on `redirect_uri` parameter to steal authorisation tokens
4. The access tokens and other JWT tokens were subjected to a manual revocation to check if the token continued to provide access
5. The application's logout functionality was tested for both the cookie session being destroyed and for the tokens to be revoked
6. Various parameters were tested extensively for injection attacks in an attempt to invoke errors or cause user supplied data to mix with a back-end query
7. An attempt was made to perform authorization bypass by reusing expired tokens, removing tokens completely, tampering tokens using tokens belonging to different account
8. The application was also subjected to various input that could be used to potentially leak information to the client

9. The JavaScript was reviewed in an attempt to discover secrets, tokens, keys and hardcoded variables that may cause information to be disclosed to the client
10. The JWT token implementation has been tested for mis-configurations that can lead to potential unauthorised access to sensitive data
11. An attempt was made to discover possible hidden API endpoints
12. The PDF generation feature has been tested thoroughly for various security issues that might arise such as XSS and SSRF
13. The file upload feature has been tested for issues such as arbitrary file upload
14. The Cross-Origin Resource Sharing (CORS) implementation on the API calls has been tested for common mis-configurations
15. As the application uses Azure Cloud services, we have performed extensive reconnaissance to identify any Azure services related to the applications that are mis-configured

Copyright © Appsecco



Summary of Results

No critical, high or medium severity issues have been discovered. One low severity issue has been identified.

One low severity issue occurs because the Identity Servers use jQuery, a JavaScript library and the version of the library used is outdated which puts the application at the risk of a Cross Site Scripting attack.

The testing showed that the application has robust authentication and authorization mechanisms using IdentityServer that prevent any unauthorised access to sensitive data. The application is built with security in mind and can withstand attacks against common web application security vulnerabilities.

The following is a quick summary of the issues discovered during the assessment:

- A JavaScript library used in the environment is of an older version that has been shown to be vulnerable to a Cross Site Scripting attack. Given the complexity of the exploit and the conditions under which this becomes dangerous, this issue has been rated low.

Copyright © Appsecco