IT Security Assessment Executive Summary Report

Commercial In Confidence

Patronus Safeguarding

Patronus Safeguarding - Web Application Test

Testing Conducted

Web application

Testing Dates

11 May 2026 - 22 May 2026

This report was generated on 18th May 2026 at 14:04 PM UTC by Martin Chapman.

Test Overview

Test Ratings

16

Number of Findings

CRITICAL	HIGH	MEDIUM	LOW	INFO
0	0	0	6	10

Test Targets

The following lists the targets that were authorised for testing.

Web App URLS

https://platform.patronus-safeguarding.com/

Finding Summary

The table below lists the findings detailed in the report below including the number of targets that they affect and their severity.

FINDING	AFFECTED TARG.	SEVERITY
OS-26-193360: Missing HTTP Security Headers	1	Low
OS-26-193540: JWT Uses HS256	2	Low
OS-26-193541: No Email Verification Following Registration	2	Low
OS-26-193546: User's Session Does Not Expire After Password Update	2	Low
OS-26-193544: [Not Fixed] Weak Password Requirements - Lack of Deny List	2	Low
OS-26-193515: Insufficient Rate Limiting	1	Low
OS-26-193359: Missing API HTTP Security Header	1	Info
OS-26-193350: [Fixed] Privilege Escalation and Tenant Isolation Bypass via Mass Assignment	2	Info
OS-26-193422: [Fixed] Account Takeover via Stored Cross-Site Scripting	2	Info
OS-26-193457: [Fixed] Weak Access Controls & IDORs	2	Info
OS-26-193545: [Fixed] Sessions Not Expired Server Side On Logout	2	Info
OS-26-193425: [Fixed] Insecure Account Lockout Policy	2	Info
OS-26-193429: [Fixed] MFA Bypass via Pre-MFA Bearer Token	2	Info
OS-26-193354: [Fixed] Username/Email Enumeration	2	Info
OS-26-193533: [Fixed] Stored Cross-Site Scripting	2	Info
OS-26-193446: [Fixed] Gratuitous Information Disclosed In Error Messages	2	Info

Executive Overview

Overall Rating

The overall rating is calculated using the **16** findings and other information contained within the rest of this report.

A total of **1** targets were tested between **11 May 2026** and **22 May 2026** with a total effort of **30 hours**. The following testing methodologies were used: **web application**.

Executive Summary

This section provides an overview of the web application penetration test completed on the Patronus Safeguarding project; Patronus Safeguarding - Web Application Test between the dates; 11th May 2026 and 18th May 2026.

The following assets were in-scope for the engagement:

- **Patronus Safeguarding; Patronus Safeguarding - Web Application Test**
  - **Patronus Safeguarding | Web Application**
    - https://platform.patronus-safeguarding.com
    - https://api.patronus-safeguarding.com

The following testing approach was taken for this engagement:

Authenticated Web Application Penetration Testing

Grey and Black box web application penetration testing from the Internet. Simulating an external attack by malicious authenticated and unauthenticated actors from the Internet.

The following provided test accounts were used for testing purposes:

- onsecurity@test.com
- onsecurityadmin@test.com
- onsecuritydepartmentadmin@test.com
- Numerous testing accounts were also self-registered under the `@onsecurity.co.uk` domain.

Testing Caveats

None.

Findings Overview

In total, four **High** severity issues were discovered, all of which had been remediated by the conclusion of the assessment:

- A mass assignment vulnerability that could be exploited to escalate privileges to administrator and switch into an arbitrary tenant, exposing the victim organisation's data and configuration.
- A stored cross-site scripting (XSS) vulnerability via SVG file uploads that could be exploited to execute malicious JavaScript in the browsers of other users, potentially enabling credential harvesting of administrator accounts.
- A stored cross-site scripting (XSS) vulnerability via the form template editor that enabled the injection of malicious scripts into publicly accessible form URLs, facilitating cross-tenant authentication token exfiltration and account takeover.
- Weak access controls and insecure direct object references that enabled interaction with resources and users outside of the authenticated user's own tenant boundary.

Three **Medium** severity issues were also identified, all of which had similarly been remediated by the conclusion of the assessment:

- The absence of account lockout policies or rate limiting on the login and MFA verification endpoints, meaning an unlimited number of password guesses and MFA codes could be submitted without restriction, enabling brute-force attacks against user accounts.
- A flaw in the authentication flow that could be exploited to bypass multi-factor authentication and gain unauthorised access to protected accounts.
- Verbose error messages that disclosed the full internal representation of cross-tenant organisation objects, including sensitive fields such as invite tokens, third-party integration credentials, and internal configuration.

All remaining issues were deemed **Low** or **Informational** risk in nature and were related to security best practices not being followed. Although these posed no immediate significant security risk, addressing them would help harden security further. These included:

- The application and its API lacked certain HTTP response headers intended to harden security, increasing the risk of attacks such as cross-site scripting and clickjacking.
- User sessions were not correctly terminated on the server side upon logout, increasing the risk of session hijacking. This issue has been remediated.
- Differing responses in login and password reset functionalities enabled the identification of valid user accounts. This issue has been remediated.
- The lack of a password deny list meant common and easily guessable passwords could be used, increasing the risk of brute-force attacks.
- A weak signing algorithm was used for authentication tokens, increasing the risk of token forgery.
- Rate limiting was either absent or ineffective across API requests, increasing the risk of abuse and resource exhaustion.
- Email addresses were not verified during registration, increasing the risk of account impersonation and spam.
- User sessions did not expire after a password update, increasing the risk of continued unauthorised access.

Addressing these issues would significantly enhance the security posture of the application and reduce the likelihood of exploitation.

General Recommendations

Further to the recommendations made for each individual issue, we would like to make the following general recommendations:

- Enforce strict parameter allowlisting on all API endpoints to prevent mass assignment of sensitive attributes such as roles and organisation identifiers.
- Sanitise and encode all user-supplied input before storing and rendering it, and restrict file uploads to safe formats where applicable.
- Implement server-side validation to ensure that referenced resources belong to the authenticated user's own organisation.
- Implement account lockout policies on authentication endpoints and ensure that the MFA verification flow cannot be bypassed by directly accessing authenticated endpoints with a pre-MFA bearer token.
- Ensure that error messages and API responses do not expose internal object representations, credentials, or configuration data.
- Implement key HTTP security headers across the application and its API to mitigate against common attack vectors.
- Ensure that logout functions and password changes correctly terminate or blacklist all active session tokens on the server side.
- Ensure that application responses do not divulge the existence of user accounts.
- Implement a password deny list to prevent the use of commonly used passwords.
- Migrate JWT signing to an asymmetric algorithm, such as RS256 or ES256.
- Apply consistent and effective rate limiting across the API.
- Ensure that users must verify their provided email address during the registration process.

Testing Methodology

OnSecurity is a CREST (Council of Registered Ethical Security Testers) accredited UK pen-testing vendor. This ensures our testers and our processes adhere to best in class manual pen-testing processes and methodologies. Our in house developed testing methodologies are based on OWASP and OSSTMM. For this test the following core activities were conducted.

Web Application Penetration Testing

Testing was conducted using the OnSecurity OWASP based testing methodology and included coverage of the following key security areas:

- Web server and supporting infrastructure configuration review
- Application Mapping
- Encryption/Cryptography Review
- Session Handling Review
- Input Validation Review
- Information Leakage Review
- Application Logic Review
- Overall application code quality
- Environmental/ configuration/ integration web server issues
- Authentication review
- API Security Review (where applicable)
    - General security misconfiguration checks
    - Authentication checks
    - Access control checks
    - Object level authorisation checks
    - Data leakage checks
    - Mass assignment checks
    - Input validation and Injection checks