Saxophone US Web App - October 2020 Penetration Test Report

TARGET(S)

https://pentest1.staging.saxophone.com
https://pentest2.staging.saxophone.com

TEST PERIOD

Oct 1, 2020 ⟶ Oct 15, 2020

STATUS

Final

TEST PERFORMED BY

Frederic Chopin

Pentester

Contents

Executive Summary

A gray box penetration test of the Saxophone Web application was conducted in order to assess its risk posture and identify security issues that could negatively affect GlobalSaxophone - East's data, systems, or reputation. The scope of the assessment covered Saxophone main portal and the Saxophone Process Orchestrator and included credentials for various levels of privilege within the application(s). The pentest was conducted by 2 pentester(s) between Oct 1, 2020 and Oct 15, 2020.

This penetration test was a manual assessment of the security of the application's functionality, business logic, and vulnerabilities such as those catalogued in the OWASP Top 10. The assessment also included a review of security controls and requirements listed in the OWASP Application Security Verification Standard (ASVS). The pentesters leveraged tools to facilitate their work; however, the majority of the assessment involves manual analysis.

The pentesters identified 3 High, 3 Medium, and 3 Low risk vulnerabilities.

During the assessment the pentesters found a critical issue related to input validation which lead to a server side request forgery (SSRF). In addition to this multiple best practice issues were noted such as a lack of session termination at logout, and no account lockout policy. Due to the nature of the findings as a whole an attacker may gain access to sensitive information such as AWS credentials. Additionally, a combination of issues could lead to an attacker successfully guessing valid accounts and compromising those with weak passwords.

Significant findings include:

- Server Side Request Forgery
- Session Management Flaws
- Missing Tenant Segregation
- Missing Account Lockout/Rate Limiting Controls

Specific recommendations are provided for each finding. As a whole, the recommendations indicate gaps that can be addressed by improvements to input validation and access controls.

Scope of Work

Coverage

This penetration test was a manual assessment of the security of the app's functionality, business logic, and vulnerabilities such as those cataloged in the OWASP Top 10. The assessment also included a review of security controls and requirements listed in the OWASP Application Security Verification Standard (ASVS). The pentesters conduct manual analysis assisted by tools.

The team had access to authenticated users, enabling them to test security controls across roles and permissions. This included attempting "vertical" privilege escalation (access to information not authorized within the container/project) and "horizontal" privilege escalation (access to information in other containers/projects without authorization).

The following is a brief summary of the main tests performed on the Web Application:

- Authenticated user testing for session and authentication issues
- Authorization testing for privilege escalation and access control issues
- Input injection tests (SQL injection, XSS, and others)
- Platform configuration and infrastructure tests
- OWASP Top 10 testing

Below is the summary of methodologies used to assess the security at the mentioned endpoints including inventory of web service endpoints, manual and automated fuzzing of web service endpoints, and others.

Inventory of web service endpoints:

We inventoried the calls made by the Stark web application during all user activities. We then proceeded to analyze requests and responses to observe the underlying technology and any possible vulnerabilities. **observation**: we observed a protocol based on JSON. The endpoints

suggest that the application is using many modern frameworks.

Manual and automated fuzzing of web service endpoints:

We proceeded to reverse engineer the endpoints and perform modified calls using manual and automated methods. We attempted the following:

- Parameter manipulation (adding / modifying parameters to perform new functions, horizontal privilege escalation, etc.)
- Code injection (SQL injection attempts, Template injection, Cross Site Scripting attacks, etc.). **Observation:** During testing it was discovered that the application is well protected. But it was susceptible to SSRF.
- XML External Entity attacks **Observation**: the XML format allows for defining / calling XML Entities, but the loading of external DTD's and SYSTEM entities was not possible.
- Header manipulation
- Path traversal
- Malicious file upload

Test Cases that successfully thwarted exploitation

Here, focus on any tests that were performed by the team that resulted in validating *good* qualities observed within the application.

We successfully enumerated the attack surface, then fuzzed for XSS/SQLi and other input injection-related vulnerabilities. The results showed that successful security controls and/or design patterns were implemented consistently throughout the enumerated attack surface with relation to most user input. There was a single SSRF identified. Furthermore, there were some access control issues given the multi-tenant environment and multiple best practice recommendations that could improve the security posture.

During testing many positive controls were observed to be in place within the application. A brief overview of these includes:

- The sessionid session cookie has the HttpOnly and Secure flags set

- The application uses several methods to prevent Cross-Site Request Forgery (CSRF) attacks - it uses both a 'csrftoken' cookie and a X-CSRF token header. For important/sensitive profile update actions the HTTP PUT and PATCH methods are used.
- Security headers such as X-Frame-Options, X-XSS-Protection and Strict-Transport-Security are in use. **Observation:** The Content Security Policy is in Report-Only mode.
- Upon password change the user receives an email, and at the next action in the app he/she is signed out and has to re-authenticate.
- The forgot password token cannot be used twice.
- The 'sessionid' expires at logout and cannot be reused
- It is not possible to access another users documents on the API when supplying their direct object reference.

Target description

Application:
https://pentest1.staging.stark.com
https://pentest2.staging.stark.com

Environment:
Staging

Assumptions/Constraints

No constraints were placed on testing the application.

Methodology

The test was done according to penetration testing best practices. The flow from start to finish is listed below.

Pre Engagement

- Scoping
- Customer
- Documentation
- Information
- Discovery

Penetration Testing

- Tool assisted assessment
- Manual assessment of OWASP top 10 & business logic
- Exploitation
- Risk analysis
- Reporting

Post Engagement

- Prioritized remediation
- Best practice support
- Re-testing

Risk Factors

Each finding is assigned two factors to measure its risk. Factors are measured on a scale of 1 (very low) through 5 (very high).

Impact

This indicates the finding's effect on technical and business operations. It covers aspects such as the confidentiality, integrity, and availability of data or systems; and financial or reputational loss.

Likelihood

This indicates the finding's potential for exploitation. It takes into account aspects such as skill level required of an attacker and relative ease of exploitation.

Criticality Definitions

Findings are grouped into three criticality levels based on their risk as calculated by their business impact and likelihood of occurrence, `risk = impact * likelihood`. This follows the OWASP Risk Rating Methodology.

High

Vulnerabilities with a high or greater business impact and high or greater likelihood are considered High severity. Risk score minimum 16.

Medium

Vulnerabilities with a medium business impact and likelihood are considered Medium severity. This also includes vulnerabilities that have either very high business impact combined with a low likelihood or have a low business impact combined with a very high likelihood. Risk score between 5 and 15.

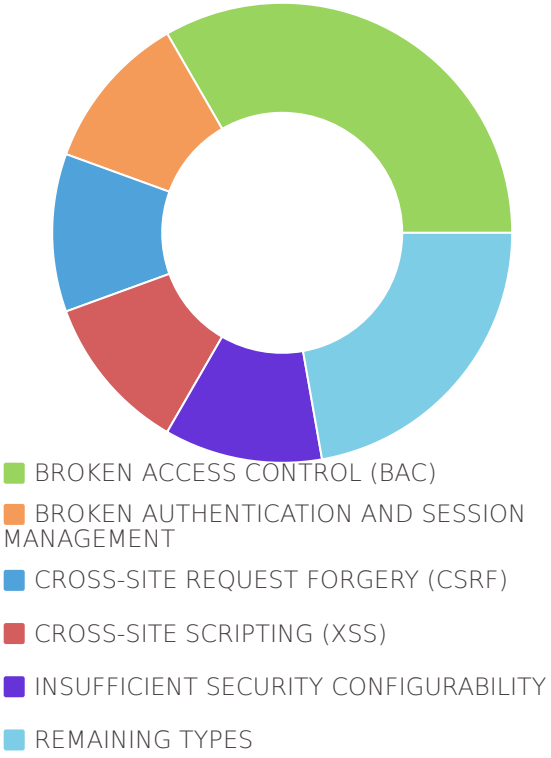
Low

Vulnerabilities that have either a very low business impact, maximum high likelihood, or very low likelihood, maximum high business impact, are considered Low severity. Also, vulnerabilities where both business impact and likelihood are low are considered Low severity. Risk score 1 through 4.

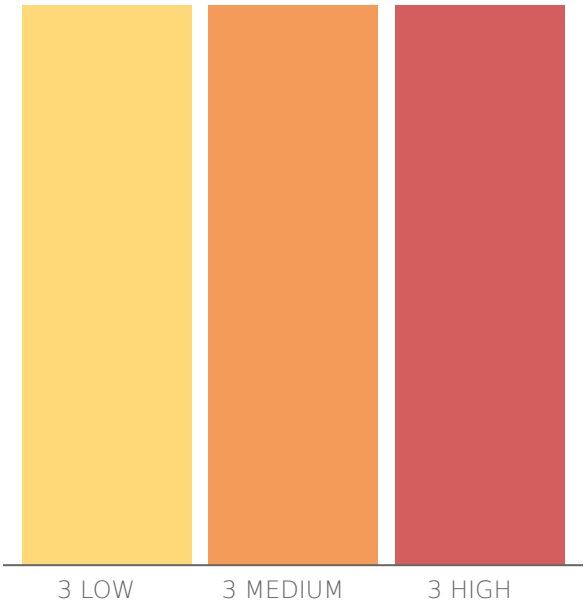
Summary of Findings

The following charts group discovered vulnerabilities by OWASP vulnerability type, and by overall estimated severity.

BY VULNERABILITY TYPE



BY CRITICALITY



PDF created May 04, 2021 18:20 UTC
Cobalt Web App 2021
by Cobalt Labs • San Francisco, CA • cobalt.io

Analysis

During the test there was a strong focus on the many dynamic inputs through various form fields and URL parameters. These inputs were observed to drive many dynamic portions of the rendered pages and fortunately there were no XSS or SQLi related issues. Unfortunately however, there was an SSRF discovered in the process creation feature which can be abused to leak the AWS credentials in use. Because of this it is recommended that stronger input validations are performed on all untrusted user supplied input.

Some less severe issues were also noted such as a lack of session termination at logout. This is a standard best practice control in session management and it is recommended that user sessions are terminated when the user logs out to prevent an attacker with access to the session token from using it to impersonate the victim. There was also an issue with the multi-tenant access controls that may allow for editing or reading another tenants data.

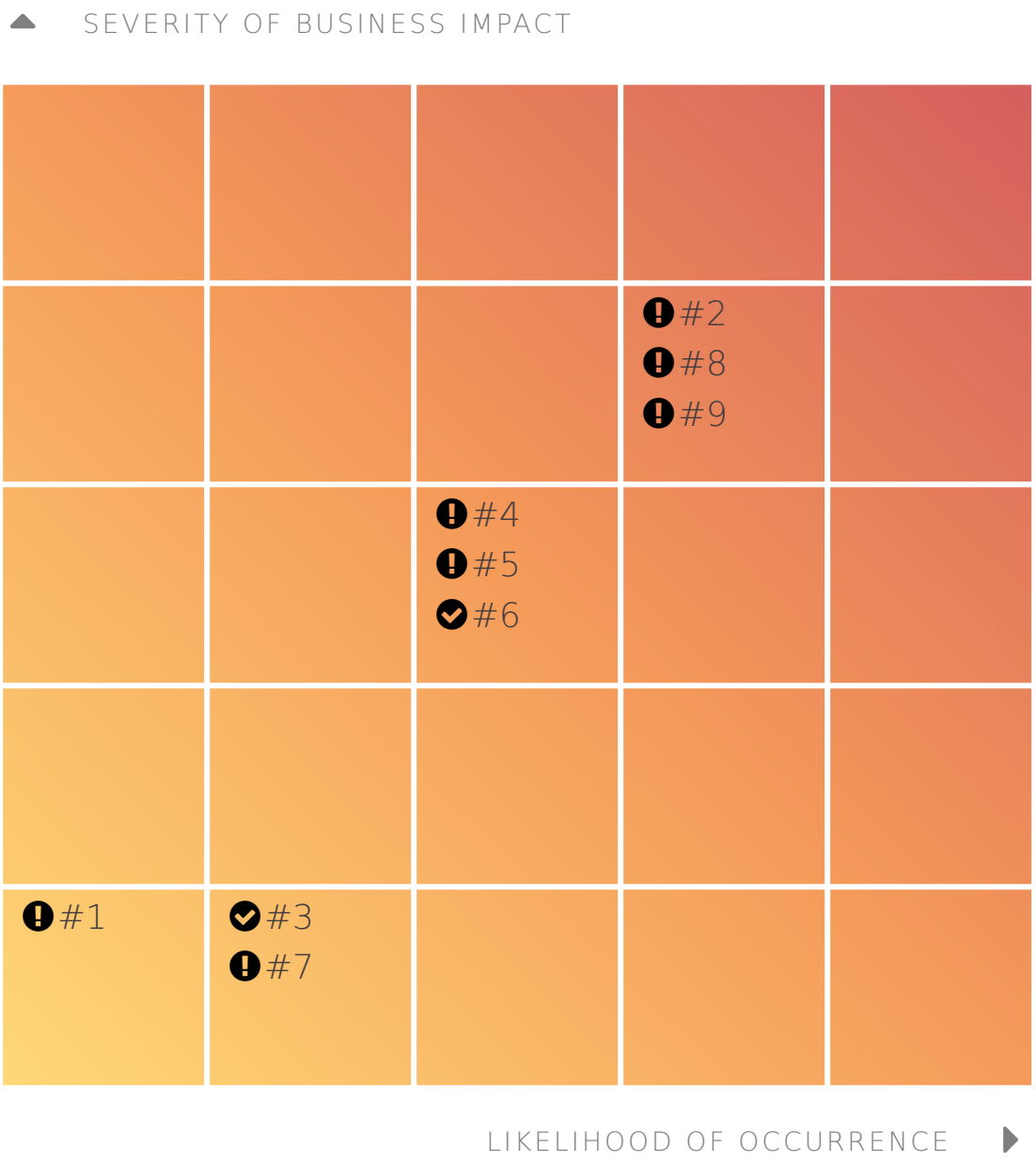
Open Ports and Services

This section documents all open ports and services identified for the target as well as any potential action needed.

Port	Service and Version	Action needed (if any)
80	http web server	none
443	https web server	none

General Risk Profile

The chart below summarizes vulnerabilities according to business impact and likelihood, increasing to the top right.



PDF created May 04, 2021 18:20 UTC
Cobalt Web App 2021
by Cobalt Labs • San Francisco, CA • cobalt.io

Recommendations

We recommend running regular security assessments to identify vulnerabilities introduced by new features or refactored code.

Based on the findings the following remediation is suggested:

1. Perform input sanitization, prefer whitelisting to blacklisting, and output encoding of all user inputs.
2. Require strong passwords such as a minimum length of 10 characters and requiring upper and lower case letters plus numbers and symbols.
3. Terminate user sessions at logout, as well as after a business acceptable amount of time to expire.
4. Do not allow sessions on one tenant to interact with another tenant that may be hosting another organizations data.
5. Lockout accounts after multiple failed login attempts and limit login attempts with a captcha or temporary IP bans.
6. Do not disclose internal IP addresses in public DNS records.
7. Provide generic responses to requests to the forgot password feature.

Post-Test Remediation

As of the conclusion of this document, the following mitigations have been implemented for the identified vulnerabilities.

Finding	Type	Criticality	State	Resolved
#PT3524_2	Broken Access Control (BAC)	High	Ready for Re-Test	
#PT3524_3	Sensitive Data Exposure	Low	*Accepted Risk	17 Nov 2020
#PT3524_4	Broken Authentication and Session Management	Medium	Ready for Re-Test	
#PT3524_5	Broken Access Control (BAC)	Medium	Ready for Re-Test	
#PT3524_6	Server Security Misconfiguration	Medium	Accepted Risk	13 Nov 2020
#PT3524_8	Cross-Site Scripting (XSS)	High	Ready for Re-Test	
#PT3524_9	Cross-Site Request Forgery (CSRF)	High	Ready for Re-Test	
#PT3524_7	Broken Access Control (BAC)	Low	Ready for Re-Test	

Finding	Type	Criticality	State	Resolved
#PT3524_1	Insufficient Security Configurability	Low	Ready for Re-Test	

*Accepted Risk Reasons

Finding	Accepted Risk Reason
#PT3524_3	Risk mitigated by Web Application Firewall (WAF).

PDF created May 04, 2021 18:20 UTC
Cobalt Web App 2021
by Cobalt Labs • San Francisco, CA • cobalt.io

Terms

Please note that it is impossible to test networks, information systems and people for every potential security vulnerability. This report does not form a guarantee that your assets are secure from all threats. The tests performed and their resulting issues are only from the point of view of Cobalt Labs. Cobalt Labs is unable to ensure or guarantee that your assets are completely safe from every form of attack. With the ever-changing environment of information technology, tests performed will exclude vulnerabilities in software or systems that are unknown at the time of the penetration test.