OWASP Security Report

HeardIT

Mihail Vasilev

Date: 25.05.2024

# Introduction

The purpose of this report is to analyze and outline any potential security risks that HeardIT web-application may be vulnerable to. In order to accurately examine the possible security issues, I am going to take the OWASP top 10 and based on them I am going to determine whether the application is exposed to any possible threats. I am also going to state what actions I am going to take in order to address the found vulnerabilities.

# OWASP Top 10

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Impact** | **Risk** | **Possible actions** | **Planned** |
| **A01** –  Broken Access Control | Severe | High | * Prevent unauthorized user access * Different levels of permission depending on the type of user * Allow access to the services only from trusted origins | Yes |
| **A02** – Cryptographic Failures | Severe | Moderate | * Auth0 used for user authentication and authorization | Yes |
| **A03** –  Injection | Severe | Moderate | * Make the application and the database protected from direct SQL injections with the use of Object Relational Mapping Tools (ORM). * External data storage providers like Auth0 and Google Cloud Storage | Yes |
| **A04** –  Insecure Design | Severe | Moderate | * The application follows the established best practices and conventions for architecture design. | Yes |
| **A05** –  Security Misconfiguration | Moderate | High | * The application is kept regularly updated and configurations are reviewed on a timely basis * Any unnecessary features and test accounts are to be removed once the project goes live | Yes |
| **A06** –  Vulnerable and Outdated Components | Moderate | Moderate | * Services are regularly updated and use well maintained versions of software components * Utilization of professional products and services like Auth0 and Google Cloud which have continuous development cycles | Yes |
| **A07** – Identification and Authentication Failures | High | High | * The application validates and prevents user from creating unsafe credentials * External authentication like Google, Facebook and more is implemented * Two-factor authentication is integrated using Auth0 | Yes |
| **A08** –  Software and Data Integrity Failures | Moderate | Low | * The application uses trusted and well- established libraries, components and service providers like Google and Auth0 * Application is updated on a regular basis * Security checks for files that might contain malicious contents | Yes |
| **A09** –  Security Logging and Monitoring Failures | Moderate | Low | * Monitoring through the use of Google Cloud metrics and monitoring capabilities | Yes |
| **A10** –  Server-Side Request Forgery (SSRF) | Low | Low | * The services send data only to trusted URLs | Yes |

# A01 – Broken Access Control

This policy ensures that users are not allowed to read and manipulate information outside of their intended permissions. This is a simple, yet effective method to prevent unauthorized users from having access to the application’s sensitive data and functionalities. In case of failure, data may be leaked, modified, destroyed or functions that otherwise are outside of the user’s range of permissions may be used in order to cause damage.

To prevent from such attacks, the first thing that needs to be established is the user’s permissions and available functionalities. For example, a regular user should not have the rights to perform actions that are reserved for the web-application’s admins. Another point is that the gateways and API endpoints needs to be protected from unauthorized users and untrusted origins (Cross-Origin Resource Sharing). One should not be able to access the applications APIs and other services from outside the intended use-cases (in my case the HeardIT front-end).

Proper user authentication and role segregation will be put into practice to minimize the risk of broken access control. HeardIT’ s services will be properly protected from outside and untrusted origins. This safety measures will be taken in order to minimize the risks and protect both the users and the data on HeardIT.

# A02 – Cryptographic Failures

This security measure focuses on issues related to incorrect implementation of cryptographic functions, leading to data exposure and integrity problems. Depending on the information handled and stored, different protections methods and measures must be implemented in order to comply with regulations such as EU's General Data Protection Regulation (GDPR). Sensitive information such as passwords, personal private information and more need to be stored securely. The best general direction to follow when working with sensitive data is that, the less sensitive information you store the better. Any data that is unnecessary must not be stored. Nowadays user information can be seen as tool and as such it must be protected and regulated measures need to be put in practice.

Since the majority of my sensitive data is related to the user information that HeardIT handles, I am going to use Auth0 as my authentication, authorization and user data storage provider. This was determined as the most suitable solutions for the user data protection for my application since Auth0 is a trusted and highly specialized in the field service that focuses on providing secure and reliable security measures. I have examined Auth0 and explained why I have chosen to use them for my authentication, authorization and user data storage service in the *Research-DataDistribution-HeardIT.docx* document.

For the rest of my services, I am going to keep my technologies up to date and avoid using outdated or deprecated technologies, since they can pose a big risk for the applications’ security.

# A03 – Injection

HeardIT’ s services can become vulnerable if malicious commands and queries to the database are used, object-relational mapping is not properly implemented and the data received from outside is not properly validated, filtered and sanitized by the different services. Some of the most common injections are SQL, NoSQL and OS commands. Since I am using MySQL databases for my data storage, I need to be aware of such dangers and place sufficient prevention mechanisms.

To prevent from such attacks, each service will use a safe and secure API that allows access only from trusted origins. All outside data should be validated before it is used and stored. The access to the database will be strictly controlled and protected from outside threats. Another protection measure is that the data will be split into multiple storages across the services, depending on the use case.

# A04 – Insecure Design

The application is created to be secure by design from potential attacks. A well-designed application with good integrity, proper user story developed workflow, failure states and well defined and established architecture ensures that the information is kept safe and data is not lost nor misused. These factors are incredibly important for the application since they are related to the general structure and design of the whole application which might not be feasible to be changed on a later stage of the development. For these reasons, this is a core principle that is a part of the entire development process from the very beginning.

Proper microservice architecture design is being followed for all of the services. Each service is specialized in its own functionality. Each service has access only to its needed components and services. This way it is harder for an attacker to gain access to the entire application.

# A05 – Security Misconfiguration

Misconfigurations can occur in cases where appropriate security measures are not implemented or where test and default accounts are kept enabled or unchanged. Sometimes inappropriate error handling may reveal overly informative information to users who are not supposed to know such information. Frameworks and software that is not up to date can cause potential flaws in the security. Any unnecessary features that have been utilized during the development process are to be removed in the final product.

To prevent such issues, it is advised to regularly review and update security configurations for Kubernetes, Auth0, and all other components. Software will be kept up to date with the latest packages and configurations to avoid vulnerabilities. The application should be kept clean from any unnecessary features and components and should be properly configured for its actual users.

# A06 – Vulnerable and Outdated Components

Vulnerabilities can occur due to outdated components, packages, libraries or is unsupported by the developers. Regular updating and monitoring are to be implemented to ensure that the application is up to date and follows the modern standards for security. Part of this regular updating is done by using professional services like Auth0 and Google Cloud which are widely used and have active development and support cycles. Components are be obtained from official and trusted sources. Any unused and unnecessary dependencies are be removed.

# A07 – Identification and Authentication Failures

Services and applications with poor user authentication procedures are prone to identification and authentication failures and attacks. Potential weaknesses such as permitting automated and brute force attacks, weak or default passwords, ineffective or weak credential recovery processes or improper usage session identifiers can lead to breaches and misuse of information.

To prevent from such potential faults, HeardIT will ensure that its users are not allowed to create simple and easy to crack authentication credentials and implement multi-factor authentication with the use of Auth0’s services. Auth0 has the functionality to enforce strong passwords and can integrate two-factor authentication into the authentication process. Furthermore, authentication and authorization using external services like Google, Facebook and other will also be implemented. The application will not be deployed with any default superuser credentials.

# A08 – Software and Data Integrity Failures

Applications that rely on many outside plugins, libraries and modules especially from untrusted sources are prone to integrity vulnerabilities. Since HeardIT uses third party provided tools and services for some vital parts of its functionalities, it is required that proper research into the credibility and validity of the providers is conducted. Both Auth0 and Google Cloud are well trusted by the development world and have proven that they are at the forefront of the modern software industry. Sufficient verification must be implemented when auto-updated and use only trusted sources. The code for the services is always reviewed in order to ensure that any potential for introducing malicious software is kept to a minimum. Since HeardIT is working with a lot of different types of information such as files, integrity checks for files in Google Cloud Storage are to be implemented to prevent tampering.

# A09 – Security Logging and Monitoring Failures

Failure monitoring and logging of the application’s services is an essential requirement in every modern enterprise application. This is done to ensure that any security breaches are caught as soon as possible and to have a proper overview of the performance and execution of the operations by each service. In the HeardIT this is will be done by implementing sufficient logging capabilities in each service and also by utilizing the deployment environment monitoring and logging capabilities. Google Cloud allows for a live monitoring of the all aspects of the application including the traffic, load, CPU and memory usage, and many other parameters to monitor the application services.

# A10 – Server-Side Request Forgery (SSRF)

Access to the HeardIT services in the cluster is sanitized and validated so that it is harder for a malicious user to create an opportunity for an SSRF breach. Such an attacker will attempt to make the application services to send information to an unexpected destination which otherwise would not be permitted. To prevent such vulnerabilities, HeardIT will validates and send only to trusted and destinations on the allowed list. HTTP redirections are disabled and raw responses are never sent directly to the clients. Kubernetes network policies can be considered to restrict the outbound traffic and firewalls can be implemented at a later stage to block malicious requests.

# Conclusion

To conclude the OWASP Top 10 security report, HeardIT is a well maintained, well designed and well developed web-application that utilizes the most modern and relevant security by design principles for an enterprise we-application. This is done since the protection of the data that HeardIT handles is of utmost importance in this new digital age where data can be used for malicious purposes in the hands of the wrong person or organization. I strive for keeping the vulnerabilities to a minimum in order to allow my users to have a pleasant and safe experience while enjoying their favorite music.

Security is an ever-changing topic that needs to be addressed regularly. As you have seen, most of the main issues arise once the application stops being actively developed and maintained. This means that even at this moment, there are new issues that might cause problems for the security of data. For this reason, I want to outline that there is always something more that can be done and as the developer of HeardIT I will continue to do my best in order to keep the security of the application to a proper level.