16.06.2022

Security analysis Report

By Erick Divin, Serhii Lysin, Alexey Kovzel, Andreea Goga,
Marijke van Iperen, Tymur Astashov

Database Security

The security of the Dao classes.

First of all the dao classes are java classes that take care of all communication between the database and the backend in java. Those dao classes are used as a service for the resource classes. No other classes should have the option to use the database. At the end it is the goal to use only SQL related code in the main class. That class should be extended by other dao classes. In those classes all input would be sanitized by the prepared statement. An XSS or SQL injection should not be possible because of that.

Further the classes in the DAO are private/protected such that other classes are not able to enter the database. So, the dao class should not cause any security issues.

Authentication

The authentication is done by either logging into an existing account or creating a new one on sign up. By credentials we chose the user's email and password that are used only once to enter a new session with the server. After that, the user acquires a token that is randomly generated in the UUID format. This token is stored in the user's browser as a cookie and is used to access server resources. Also, this token points to the user's account, so that the server can understand the user's permissions and their role on the server (e.g. Client, Admin, Crew member). Each token has an expiration date, after which the token is deleted on the user's browser and he/she should be authenticated again.

Input Validation

When a user submits a form (e.g. when logging in, booking an event, etc.), their input is validated on both the client's and the server's sides using regex patterns. Validation on the client's side is necessary to notify the user about the invalid input before actually sending the data to the server. And on the server's side for preventing such data from being stored (for security, consistency and reliability). More about the validation:

Fullname - validates that a given fullname is not empty, does not contain numbers or special characters.

Password - validates that a password contains at least 3 characters, one letter and one number.

Phone - validates that a phone number contains ten digits, probably parentheses or an international prefix.

Password - validates that a password contains at least 3 characters, one letter and one number, so that it cannot be easily brute forced.

Email - validates that an email adheres to the regex pattern provided by RFC 5322.

XSS Prevention

At this stage of development of the project, we didn’t use much techniques of preventing unwanted XSS attacks, as the front end part is not yet fully completed. However, with the development of the project we’re planning to implement some of the most common prevention techniques:

- filtering input on arrival: at the point where the user input is received, we filter based on what is expected as an input/valid input
- output encoding: both for HTML(for example URLs, Attribut Contexts) and CSS
- using HTML safe attributes
- using Cookie Attributes
- rewriting URLs