*OWASP*

*REPORT*

By Tim Verhees  
 386482

V1.0

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Likelihood** | **Impact** | **Risk** | **Actions** | **Planned** |
| A1 Broken Access Control | Moderate | Low | Low | Further limit functionality for regular users | No |
| A2 Cryptographic Failures | Low | Low | Low | N/A | No |
| A3 Injection | Low | Moderate | Low | N/A | No |
| A4 Insecure Design | Moderate | Moderate | Moderate | Restructure data architecture | No |
| A5 Security Misconfiguration | Low | Moderate | Low | Rechecking of all configurations regarding security | No |
| A6 Vulnerable and Outdated Components | High | Moderate | High | Keeping up to date with packages and making sure no vulnerable packages are used | Yes |
| A7 Identification and Authentication Failures | High | High | High | Limiting log in attempts, enforce stricter password ruling. | No |
| A8 Software and Data Integrity Failures | Moderate | Moderate | Moderate | Making sure that only trusted packages are used and pipeline optimization | Yes |
| A9 Security Logging and Monitoring Failures | Moderate | Low | Low | Making sure that all failed attempts at access are logged sufficiently | No |
| A10 Server-Side Request Forgery (SSRF) | Low | Low | Low | N/A | No |

Reasons

A1

When looking at the actual application of my web application and the data that is used, access control is of low consequence. This is because there is no sensitive data used in my application. Now it would be quite a hindrance regarding losing the data in case of such a breach. However, for these instances authorization is implemented.   
A2

Once again, no sensitive data is being used in my application. This means that at the end of the day the only real data in need of hashing would be passwords. Which is already functional. The application will not actually use any credit card or payment info, no personal records or anything like that.  
A3  
The already implemented method of data processing with requests is already secure. All the user input is not taken as literal and are processed in a secure method where common methods of injection would have no effect. In addition to this, any and all data from the application is non-sensitive. And for actual release of the application, back ups would be used.  
A4

By design my application was made with minimal security necessity, this is because it was always intended to be simple. This meant that it would have no need for sensitive data. However certain parts if the data's structural design could see improvement, mostly regarding ban lists and the data they hold. This then also extends to the requests they have in place and how they are then prone to getting data not relevant to the call made.  
A5

It is important to have your security configured properly, whilst your back-end application is running, no other front-end application should be able to interact with it besides your own for instance. Luckily this is currently the case, my application is configured in such a way where only my back-end, front-end and database are allowed to communicate with one another. Outside influence is prevented.  
A6

Throughout the development of my web application several efforts regarding adding features led to the installation of various package that have been used for a couple of different applications. These applications are, at times, visible. Every one of these packages could have issues regarding security with known exploits into those packages. One way this could be solved is through keeping these packages as up to date as possible and keeping watch of old/abandoned packages.  
A7

Authentication is a tough concept to get right in any application, and how someone goes about it can vary greatly from application to application. Currently in my application there is a great risk of brute force attacks, seeing as there is currently no limit on login attempts on my application. Another weak point of my application is that passwords can be of any length, with any character. Of course, to ensure secure passwords across the board, I should implement both a minimum character amount and ruling to ensure stronger passwords such as mandatory special characters and capital letters. Currently this is not planned for shipping in its entirety for the deadline of the application, however limiting login attempts could be implemented.  
A8

It is important that the used packages in an application are verified and trusted for usage in one's application. You don't want to have a package that can make reading the data easier, but then also shares that data with the author of the package. Therefore, I will only be using trusted package that do not have a history of these kinds of things. Another part of this is a good pipeline implementation, this will help ensure quality of security, sufficient coverage of the application to make sure all its important features are tested and verified.  
A9

It is important to be able to recognise breaches and deal with them in a timely manner. Therefore, login attempts should be logged in a secure manner where they can be accessed at any time to respond to any ongoing issues. This will not be implemented in my application because of its lack of sensitive data usage. This also extends to testing however, and such edge cases should be tested in a way first and foremost they are prevented and secondly, logged to help identify the where the threat is coming from.  
A10  
As URL usage in requests become more and more prevalent, the risk of SSRF attacks becomes greater, this means that extra care should be used when using URLs for such requests. Luckily the implementation in my application sanitizes the user's input and therefore securely adds data to the database. There is also no usage of regular HTTP, all these requests are made using HTTPS.

Conclusion

All in all, my application is not the most secure and does need some significant work before it can really be put out into the World Wide Web, luckily it does not use sensitive data for any of its features, meaning that even if a security breach were to occur, the impact would be negligible.

This, however, does not excuse bad security design and decisions, which will be reflected upon and improved in future applications.