Security of web Application document

What is this document about? The OWASP Top 10. Are the risks applicable for my application, and why. How did the risks impact my implementation? How does this effect the risks?

Top 10 contain:

>Injection

>Broken Authentication

>Sensitive Data Exposure

>XML External Entities

>Broken Access control

>Security Configuration

>Cross-Site Scripting

>Insecure Deserialization

>Using Components with Known Vulnerabilities

>Insufficient Logging and Monitoring

1. Injection-My application uses hibernate and my backend SQL injection happens by not taking any parameters that can harm my application. The SQL statements are inside a interface using functions like “.save” , “.delete” , “.findBy” etc. Basically using ORMs.

However, **Note**: SQL structure such as table names, column names, and so on cannot be escaped, and thus user-supplied structure names are dangerous. This is a common issue in report-writing software.

I have done my best to prevent my application from injection.

2. Broken Authentication is always possible and very harmful indeed due to possible leak of private information and so on. I took this pretty seriously and implemented a very solid Authorization and Authentication as well. Starting with the fact that easy credentials for creating a user account are not allowed (example your password should contain at least 6 symbols) however it would definitely be a user’s mistake if they provide a easy to guess password. Anyway their password is hashed in the backend and not visible for anyone in the Database. Also if user forgets to log out will which will forget all of his data and cookies he will be open to exposing his data for the next user on the same device.

Additionally I did when entering the application it generates a new session and token and more giving you safety during your use of the application. Also you cannot create multiple users with the same username and more.

3. Sensitive Data Exposure

I am hashing my password data only once however it is far from enough to prevent a dangerous attack for sure because I am not encrypting any more of the important data such as ids and more. There are even data exposed in the Cookies visible and reachable for users, even though they cannot modify it themselves. Anyway I am not storing data such as Passports and more which definitely must be encrypted .

What is interesting given having my application is that I am using web sockets and a lot of messages are received between the users and I still don’t have any protection implemented which lays down the opportunity for hackers to damage it. However, I have in mind to at least to make my application enhanced using https so that it is at least a bit more secure. Then all of the data will be hard to be distinguished and read. Having read that I have exposed the data of a user in his local storage in the session , which gives him the ability to see them all even though cannot change them because it requires access to the serve. However changing them to read only cookie is going to suit just fine.

4. XML External Entities attacks are very malicious, thus

I am avoiding serialization of sensitive data but I have not implemented anything more in order to protect my XML from attacks such as changing the url to a different and so on. To be honest I am not quite aware what and hot to improve on this.

5. Broken Access Control

Restrictions on what authenticated users are allowed to do are often not properly enforced. Attackers can exploit these flaws to access unauthorized functionality and/or data, such as access other users’ accounts, view sensitive files, modify other users’ data, change access rights, etc.

Thus I have made the authorization and you have to be set either a ordinary user, admin or moderator and once you have any of those rights you are authorized and able to send requests to the server either you don’t have the access to the server side. On one hand even when logging you send a request to the server to authorize you otherwise you can not enter the application.

6. Security misconfiguration

Even though I have many unnecessary features installed I am not using them in order to prevent the impostors from harming or visiting content(pages that are not supposed to be accessible by ordinary users). Additionally I have made restrictions and redirections when one tries to access unreachable content.

7. Cross-Site Scripting

There are such risks applicable to my application because the only thing I rely on to keep me safe from this is my React Js providing valid data to my backend through @RequestBody which takes particular data and provides it to the backend . Thus I hope that if the input is not adequate a Exception will be provided. Having the Authorization should me able to handle and return at least who is responsible for the attack. To be honest I am not really sure how to prevent this in my situation.

8. Insecure Deserialization

One of the most serious attacks possible. They make my logic and application vulnerable to changes in the main code. Thus I am not sending any serialized data to the backend and relying on on json formats.

9.Using Components with Known Vulnerabilities

As long as I am concerned I have exposed only components that are fully protected and secured from crashes or bugs. However it is possible that there are some more errors that I am not aware and might occur.

10. Insufficient Logging and Monitoring

I give errors such as Error 500 or not Authorized which gives a little bit of knowledge where my protection lays thus makes my application vulnerable. I have not taken this risk in mind. And it is very likely that someone can overload my database by sending a humongous amount of data to it and this will lead to crashing the whole system. I will be digging into creating a restriction in my logic layer for a user not to be able to make more than few posts or messages for day.

By Georgi Filev