Léo MERY Florian ERNST

How we plan to handle the authentication/security for our application

In order to secure our website, we must ensure that we manage several exploitable vulnerabilities:

Firstly, we aim to prevent the possibility of SQL injections such as 'OR''=' by escaping single quotes properly.
The best solution is to prepare each query using client input. For example, if the parameters of a GET or POST method is used in a query, we must prepare correctly the query to avoid any problem.

Secondly, in order to protect passwords, we intend to use the hash algorithms offered by PHP, in this way we will be able to check the completeness of an item of information without having to check all this information. Moreover, the hashing offers us other advantages such as: not being able to decipher a hash because when it is generated (the original data cannot be easily found); the final size of a hash will always be the same; the hashing of the same data will always return the same value.
In this way, we only need to keep the hash of the user's password in our database and not the password itself. And when the user wants to connect, just compare the hash of the password he just entered with the one in our database to see if he has the right one. Thus, only the user is the holder of the password allowing access to his account.

But hash also has its flaws, as the hash always offers the same results. If a hacker has a list of knowns hashes, he can get every password in his list in a fast an easy way. That’s called a Rainbow Table attack and is a very popular method to decipher passwords. Moreover, by finding one password’s hash, the hacker would find every identical password. This means that users having the same password (and common passwords) are very vulnerables.

In order to defend against this kind of problem we plan to use a salt, in other words a suffix or prefix that we will add to the password before hashing it. Thus, decoding hash becomes extremely complex. The system is even more secure: by giving each user a unique salt, if the hacker finds a password with a bruteforce method, he will have to repeat the whole process for other users.

In addition, both the French Data Protection Act (Law n°2018-493[1]) and the European General Data Protection Regulation (GDPR) requires that users' data have a guarantee of confidentiality. If your database is hacked and no valid protection guarantees the security and confidentiality of the data contained in it, you may be held liable. This is another reason to ensure that our site is sufficiently protected.

[1] https://privacylawblog.fieldfisher.com/2018/post-gdpr-french-data-protection-law-adopted