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.

A solution can be to use this statement:

```
sql = "SELECT name FROM users WHERE name='" & replace(name,"'","''") & _
      "' AND password='" & replace(password," ' ", " ' ' " )
```

Instead of

```
sql = "SELECT name FROM users WHERE name=''  AND password= ''
```

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, hashage offers us other advantages such as not being able to decipher a hash because when it is generated, the data is lost, the final size of a hash will always be the same, the hashage 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 result, if the hacker can get the password hash, he can try to decrypt it using algorithms to compare it with the words in the dictionary.

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. It is possible to secure the system even more by giving each user a unique salt, so if the hacker miraculously arrives at bruteforce a password, he will have to repeat the whole process for other users.

In addition, the French Data Protection Act 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.