Project Horus - Security Report

Storing passwords securely

We have used an online git repository[1] for the actual hashing algorithms and password verification. It uses SHA1 iterated 128.000 times with random generated salts. We know from the lectures that SHA1 alone is not secure, but as the documentation of the repository states, using SHA1 in iterations along with salt is "secure enough" for the purpose of this project.

We hash the password when the user first registers and encode it to base64 to conceive the structure of the stored hash (at first glance, of course). When verifying the validity of an entered password, we decode the corresponding hash of the user and then use the *verifyPassword* method of the library.

Secure cookies

For storing the cookies we used an symmetric encryption. It uses AES iterated over 100000 times with a random generated salt. It encrypts the used which consists of the userID and the timestamp. When a user logs in the cookie is saved in a table containing the user and the cookie. Also if another user wants to log in to same account it can not until the user logs out.

Security against attacks

Against sql injections the system is quite secure as it uses parameterized queries and also when entering the log-in credentials the user can write only a valid email so it is quite sanitized. Against the XSS attacks we didn't really took too much precautions as we didn't find ant part of the code that could be used to start an XSS attack.

[1] https://github.com/defuse/password-hashing