From: **Müller, Normen** n.mueller@safeplace.de
Subject: Authentication via Challenge-Response
Date: 13. August 2018 at 21:19
To: Shantyr Sergei sergei.shantyr@dnetwo.de, Bobojonov Bunyod bunyodreal@gmail.com

Hi guys,

I don't feel comfortable in sending the user's password in clear text (protected by Https only) in the registration as well as in the login process.

As far as I know, regarding the registration process there is no other option then sending the password in clear text protected by Https only.

But I'd like to recommend/ discuss an *improvement regarding the login process*. Eventually I want the registration process to be the only point in time where the password is sent in clear text.

Please read carefully my proposal and give me your feedback:

 # /account/signup

Client app sends user's `e-mail` address and retrieves a *registration token*,
which is signed with a generated OTP.

The `username` and the OTP is saved in the `accounts` table.

This has already been implemented but in this proposal I changed the API path from just `/signup` to `/account/signup`.

 # /account/register

Client app sends user's `e-mail` and `password` together with the *registration token*.

If the backend successfully validates the *registration token*, the user's
password is set (overrides the previous OTP) and the user is eventually
registered.

Note, this is the only point in time where the password is sent to the backend
in clear text protected by Https only.

This has already been implemented but in this proposal I changed the API path from just `/register` to `/account/register`.

 # /account/challenge

Client app request a challenge to log in by sending his `username`.

Request body: `{ "username" : "<e-mail>" }`
Response body: `{ "challenge" : "<challenge>" }`

 # /account/login

Client app sends `username` and

```
response := md5(challenge + password)
```

to the backend.

Request body: `{ "username" : "<e-mail>", "response" : "<response>" }`
Response body: `{ "token" : "<access-token>" }`

If the backend successfully evaluates the combination of
`username` and `response` it sends back an *access token*, which has an

user name and `response`, it sends back an access token, which has an expiration time and is signed with the backend master JWT secret.

Note, that the `password` is not sent in clear text to the backend. This is an improvement of security. Again, the only point in time the `password` is sent in clear text (protected by Https only), is in the registration phase (`/account/register`).

BR, /nm