Chamilo LMS - Feature #7646

New password encryption method

21/04/2015 18:02 - Yannick Warnier

<table>
<thead>
<tr>
<th>Status:</th>
<th>Feature implemented</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority:</td>
<td>Normal</td>
</tr>
<tr>
<td>Assignee:</td>
<td>Julio Montoya</td>
</tr>
<tr>
<td>Category:</td>
<td>Global / Others / Misc</td>
</tr>
<tr>
<td>Target version:</td>
<td>1.10.0</td>
</tr>
<tr>
<td>Complexity:</td>
<td>Normal</td>
</tr>
<tr>
<td>Start date:</td>
<td>21/04/2015</td>
</tr>
<tr>
<td>Due date:</td>
<td></td>
</tr>
<tr>
<td>% Done:</td>
<td>100%</td>
</tr>
<tr>
<td>Estimated time:</td>
<td>10.00 hours</td>
</tr>
<tr>
<td>Spent time:</td>
<td>0.00 hour</td>
</tr>
<tr>
<td>SCRUM pts -</td>
<td>?</td>
</tr>
</tbody>
</table>

Description

Currently, the strongest password encryption we have is SHA1, which is already considered insecure.

We need a new encryption method, or a new set of encryption methods. Most complex (and secure) encryption methods nowadays use 2 parameters:

- one salt
- one number of encryption processes

This means that we need to add those two parameters to configuration.dist.php and then update all the code that deals with the current encryption method:

- search for $configuration['password_encryption']
- check all remaining uses of "sha1" in the Chamilo code, just in case

Then modify (mainly) the api_get_encrypted_password() function in api.lib.php to be able to use other stuff than sha1.

An interesting place to look for algorithms is in main/inc/lib/phpseclib/Net and /Crypt

This might be added to 1.10.0 if we get the time to do so. Otherwise we’ll just pass it to 2.0.

Julio, is there any existing library that we could use for that, integrating the current hashing algorithms we have (none, md5 and sha1) without too much difficulty?

Revision dd947882 - 12/05/2015 15:48 - Julio Montoya

Add symfony2 security encoders see #7646

- Add bcrypt encoder.
- Add password-compat if bcrypt is used in php < 5.5.
- New db fields: user.salt, user.username_canonical.
- Remove use of api_get_encrypted_password().
- During installation the UserManager::create_user() is now used.
- Add Repository and Manager classes for the user entity.
- Remove function encryptPass in cm_webservice.php
- Fix registration.soap.php

Revision 0f517ebf - 07/07/2015 05:00 - Yannick Warnier

Fix missing password validation in profile edition - refs #7646

History

#1 - 04/05/2015 19:43 - Julio Montoya

- Assignee changed from Julio Montoya to Yannick Warnier

The security component of Symfony2 does the job.

It supports custom encoders and already has support for bcrypt, sha1.

I could integrate only that component.
#2 - 05/05/2015 05:54 - Yannick Warnier
- Assignee changed from Yannick Warnier to Julio Montoya

How many hours do you estimate it might take you (approx)?

#3 - 05/05/2015 08:04 - Julio Montoya
- Assignee changed from Julio Montoya to Yannick Warnier

I think 1 or 2 hours. Is more easy now because we have entities already added in the base code.

#4 - 05/05/2015 15:36 - Yannick Warnier
- Status changed from Needs more info to Assigned
- Assignee changed from Yannick Warnier to Julio Montoya

OK, go ahead with this change then. Please make sure this integrates with the current encryption mechanisms we have in Chamilo (even if it takes a little longer): none, MD5 and SHA1 (yeah, I know, the first two are not encryption mechanisms), so a user can easily migrate to 1.10.x maintaining his current encryption method, but new installations can support at least one strong encryption method.

Please note that bcrypt requires PHP 5.5 or the composer installation of a compatibility module, so either use another method or install the compat module (I prefer avoiding the compat module until we move to a strict requirement of PHP 5.5, but I'll let you take the decision).

#5 - 05/05/2015 15:44 - Julio Montoya

Yannick Warnier wrote:

OK, go ahead with this change then. Please make sure this integrates with the current encryption mechanisms we have in Chamilo (even if it takes a little longer): none, MD5 and SHA1 (yeah, I know, the first two are not encryption mechanisms), so a user can easily migrate to 1.10.x maintaining his current encryption method, but new installations can support at least one strong encryption method.

Please note that bcrypt requires PHP 5.5 or the composer installation of a compatibility module, so either use another method or install the compat module (I prefer avoiding the compat module until we move to a strict requirement of PHP 5.5, but I'll let you take the decision).

No need to do that.
I found this:

In Symfony\Component\Security\Core\Encoder\BCryptPasswordEncoder

```php
if (!function_exists('password_hash')) {
    throw new \RuntimeException("To use the BCrypt encoder, you need to upgrade to PHP 5.5 or install the "ircmaxell/password-compat" via Composer.");
}
```

This package "ircmaxell/password-compat" seem to resolve that problem.

#6 - 05/05/2015 16:07 - Yannick Warnier

That's what I meant by a "compatibility module" :-p

#7 - 12/05/2015 14:56 - Julio Montoya
- Status changed from Assigned to Needs more info
- Assignee changed from Julio Montoya to Yannick Warnier

In main/admin/user_list.php there's a code that detects "easy passwords".

This code will not work with the new changes because we use a salt to check the password. It means that to validate a password 2 fields are needed the user salt and the user password.

#8 - 12/05/2015 17:44 - Yannick Warnier
- Status changed from Needs more info to Assigned
- Assignee changed from Yannick Warnier to Julio Montoya
- % Done changed from 0 to 30
Just check the password field in this case (the salt is not important in the sense that it is used to hash the rest and it will not be asked to the user trying to enter the platform).
Make sure there is a mechanism to offer a default random hash (otherwise we'll confuse the user). Actually, the hash itself could simply be generated automatically. You don't need to ask the user at all (don't even show it in the form). It's internal.

Yannick Warnier wrote:

Just check the password field in this case (the salt is not important in the sense that it is used to hash the rest and it will not be asked to the user trying to enter the platform).
Make sure there is a mechanism to offer a default random hash (otherwise we'll confuse the user). Actually, the hash itself could simply be generated automatically. You don't need to ask the user at all (don't even show it in the form). It's internal.

As far as I read, I need the salt to validate/compare passwords. So I have to ask for every user if their password is easy or not (in admin/user_list.php) If the salt is empty it will work but not for users with salt.

OK, the main/admin/user_list.php feature can be forgotten in case of bcrypt (just disable this feature there). I was talking specifically about the password recommendation on the profile edition page (which was broken). I just fixed that with https://github.com/chamilo/chamilo-lms/commit/0f517ebf1e7d67470b604dcb7aee903a38ac30e. Closing the task. Good job over the new authentication mechanism!