Redmine - Feature #11542

Improve password security and use some standards.

2012-07-31 00:18 - Mikołaj Milej

Status:	New	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:	Accounts / authentication	**Estimated time:**	0.00 hour
Target version:			
Resolution:			

Description

There is salt in the password, but it's implemented in non standard way. Developers self developed pattern like hash(salt + hash(password))

There is no protection against brute force attack (it attacker get your passwords from database), because SHA is designed/implemented to be as fast as possible, but with brute force we want attacker to be as slow as possible.

There are some information about good practice in password storing:
https://www.owasp.org/index.php/Password_Storage_Cheat_Sheet#Password_Storage_Rules
http://www.openwall.com/articles/PHP-Users-Passwords

History

#1 - 2012-07-31 03:37 - William Roush

If Redmine dev's want to do that, they'd want to look into doing something like Bcrypt (http://en.wikipedia.org/wiki/Bcrypt), because of its built in salt support and it's adaptive. I'm not sure if Ruby/Rails has any tools immediately available to it but it would be nice.

Implementation/migration could be tricky though.

As for me, I run LDAP auth so not really a huge deal.

#2 - 2012-07-31 09:32 - Mikołaj Milej

As far as I know automatic migration is impossible, all users have to reset their passwords. If are talking about password storing the point is as good security as possible.

I want to use LDAP, but now it confuses me and I don't know how to set it up.

#3 - 2012-08-05 10:15 - William Roush

> There is no protection against brute force attack (it attacker get your passwords from database), because SHA is designed/implemented to be as fast as possible, but with brute force we want attacker to be as slow as possible.

I'd still point out that mathematically to find out a specific password for an account via brute force: SHA is fine (it's still computationally unreasonable to brute force a salted SHA password). To run all salts in a DB against a known *password list* however will yield me a large number of hits relatively quickly, however using password lists isn't brute forcing.

Algorithms like bcrypt and PBKDF2 combat the list of common passwords issue, by making it computationally unreasonable to run anything but the smallest common password lists against a salted DB.

#4 - 2012-10-28 21:06 - Bernd May

While you might not call it brute forcing, when using a large dictionary it essentially is. Every second thus gained via the use of bcrypt or scrypt is a real pia for the attacker.
I wonder if this can be made into a plugin, but considering the changes it needs to make in the User model I guess it is more a candidate for a patch :-/

#5 - 2016-03-03 13:46 - @ go2null

I have a plugin that implements some password policy rules from my ISIT dept.
One is adding a timeout after X failed login attempts.
https://github.com/go2null/redmine_account_policy

Would love to see some improvements in this area though.