Redmine - Feature #4640

Change password hash to be compatible with Apache

2010-01-23 20:04 - Jerry Van Baren

Status:	New	**Start date:**	2010-01-23
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:	SCM extra	**Estimated time:**	0.00 hour
Target version:			
Resolution:			

Description

By changing the password hash to be Base64 encoded and prefixing it with '{SHA}', Apache can authenticate directly using the database using mod_authn_dbd. This eliminates the need for the Redmine.pm perl module, at least for Apache-supported databases.

http://httpd.apache.org/docs/2.2/mod/mod_dbd.html

In the general Apache configuration (note the host=127.0.0.1 is probably necessary, the port=5432 should be optional):

```
        # mod_dbd configuration
        DBDriver pgsql
        DBDParams "host=127.0.0.1 port=5432 dbname=redmine user=redmine password=[secret]"
        DBDMin  4
        DBDKeep 8
        DBDMax  20
        DBDExptime 300
```

In the Location section that you wish to require authentication for access:

```
                AuthType Basic
                Authname "Enter your Redmine user name and password"
                AuthBasicProvider dbd
                Require valid-user
                # mod_authn_dbd SQL query to authenticate a user
                AuthDBDUserPWQuery "SELECT hashed_password FROM users WHERE login = %s"
```

History

#1 - 2010-01-23 20:08 - Jerry Van Baren

In the patch, lines 96-108 of app/models/user.rb (backwards compatibility) are not strictly needed, assuming the migration is done to convert the password encoding.

#2 - 2010-01-23 20:31 - Jerry Van Baren

For reference, this improves on
http://www.redmine.org/wiki/1/Repositories_access_control_with_apache_mod_dav_svn_and_mod_perl

#3 - 2010-01-23 22:34 - Holger Just

Note that Apache can already authenticate against MySQL using the auth-mysql module. See
http://maff.ailoo.net/2009/03/authenticate-apache-against-redmine-with-authmysql/

There exists a similar module for postgres (mod_auth_pgsql2 or libapache2-mod-auth-pgsql in Debian) which unfortunately is not able to generate SHA1 hashes by itself. One could however send the original (unhashed) password in the SQL query string and use the SHA1 function from pgcrypto

I wanted to test this, but had not yet time to do so. But it should be possible.

#4 - 2010-01-23 23:04 - Jerry Van Baren

mod_authn_dbd is a generic way to authenticate against a SQL database. It appears to be a better solution than database-specific mods like mod-auth-mysql, mod_auth_pgsql2, or libapache2-mod-auth-pgsql.

Based on the fact that mod_authn_dbd is listed on the Apache 2.2 site and the others are not, I would conclude it is a better choice.

Note that my patch is really a one line change to simply put the hash in an Apache-compatible format, the rest of the code is a migration to the Apache-compatible format and some backwards compatibility code that probably is unnecessary (it was helpful for my testing).

#5 - 2010-01-23 23:45 - Holger Just

Well at least you would also have to patch source:/trunk/extra/svn/Redmine.pm#L277 to use your new password scheme.

Another alternative would be to use the following statement as AuthDBDUserPWQuery which should generate the needed format on the fly. It looks a bit ugly but the encode(decode()) is needed to please postgres' type system.

```
# mod_authn_dbd SQL query to authenticate a user
AuthDBDUserPWQuery "SELECT '{SHA}' || encode(decode(hashed_password, 'escape'), 'base64') FROM users WHERE login = %s"
```

Using this SQL string your patch would not be needed at the cost of a bit of database overhead.

#6 - 2010-01-24 00:53 - Jerry Van Baren

> Well at least you would also have to patch source:/trunk/extra/svn/Redmine.pm#L277 to use your new password scheme.

No, the patch makes Redmine.pm *unnecessary.* It isn't *my* password scheme, it is the "native" Apache password encoding and thus Apache can use the Redmine database *directly* for authentication.

> AuthDBDUserPWQuery "SELECT '{SHA}' || encode(decode(hashed_password, 'escape'), 'base64') FROM users WHERE login = %s"

That is sort of where I started. The above SQL is engine-specific: the above is Postgresql-syntax, the other dB engines use a different syntax to concatenate strings. Ugly, annoying, and error-prone.

> Using this SQL string your patch would not be needed at the cost of a bit of database overhead.

But it is silly to save a hexadecimal encoded hash only to convert it to base64 while extracting it. Redmine "encapsulates" the password handling in the User class, so a simple change to User and a simple migration is transparent to the users of User.

Nobody in Redmine other than User cares if it is hexadecimal encoded or base64 encoded. There is no reason that the Redmine database should not store the password hash in the Apache format vs. the hexadecimal format. In addition, the patch changes makes it work with all dB engines.

#7 - 2010-01-24 01:28 - Holger Just

Just tried it on my server and noticed an error in the SQL string. It should be

```
# mod_authn_dbd SQL query to authenticate a user
AuthDBDUserPWQuery "SELECT '{SHA}' || encode(decode(hashed_password, 'hex'), 'base64') FROM users WHERE login = %s"
```

While this technically works, it still lacks a very important factor! Yes, it will go to the database and look for a given user. But is does not check if the user has the correct right on the specific repository. You would have to inject the current project name (as retrieved from the URL) into the query to generate a query similar to:

```
SELECT
  '{SHA}' || encode(decode(users.hashed_password, 'hex'), 'base64')
FROM members, projects, users, roles, member_roles
WHERE
  projects.id = members.project_id
  AND member_roles.member_id = members.id
  AND users.id = members.user_id
  AND roles.id = member_roles.role_id
  AND users.status=1
  AND users.login = %s
  AND projects.identifier = <project_identifier>
  AND roles.permissions LIKE '%- :browse_repository%;
```

And I have no idea how to do that. From what I've read, it is not possible, is it?

#8 - 2010-01-24 01:50 - Holger Just

First I really like the idea of getting rid of Redmine.pm as it is the only reason for a preforked apache and mod_perl in my installation. So in the end, I'm on your side ;)

Jerry Van Baren wrote:

> No, the patch makes Redmine.pm unnecessary. It isn't my password scheme, it is the "native" Apache password encoding and thus Apache can use the Redmine database directly for authentication.

I think, we should be able to support Redmine.pm for a while. Some people might not be able to run mod_dbd. Also, Redmine.pm allows to transparently authenticate against LDAP.

> > AuthDBDUserPWQuery "SELECT '{SHA}' || encode(decode(hashed_password, 'escape'), 'base64') FROM users WHERE login = %s"
>
> That is sort of where I started. The above SQL is engine-specific: the above is Postgresql-syntax, the other dB engines use a different syntax to concatenate strings. Ugly, annoying, and error-prone.

I am sure every database (except maybe for sqlite) would allow such a concept.

> > Using this SQL string your patch would not be needed at the cost of a bit of database overhead.
>
> Nobody in Redmine other than User cares if it is hexadecimal encoded or base64 encoded. There is no reason that the Redmine database should not store the password hash in the Apache format vs. the hexadecimal format. In addition, the patch changes makes it work with all dB engines.

I see that there is actually some overhead in my approach. But I fear that there might be issues with other authentication schemes or plugins. But this is just a gut feeling...

The main problem (aka. show stopper) is the desire to authenticate for a specific repository/project not just *any* user.

#9 - 2010-03-08 22:01 - Jean-Philippe Lang

Holger Just wrote:

> While this technically works, it still lacks a very important factor! Yes, it will go to the database and look for a given user. But is does not check if the user has the correct right on the specific repository.

Indeed, it doesn't eliminate the need for the Redmine.pm.

#10 - 2010-12-16 02:47 - Jerry Van Baren

- File base64_hash.patch added

Updated the patch to apply against version 1.0.4

To authenticate from Apache against the Redmine database, use the mod_dbd and a sql query (no perl wedge necessary):

```
        # mod_dbd configuration to authenticate against Redmine
        DBDriver pgsql
        DBDParams "host=127.0.0.1 port=5432 dbname=[dbname] user=[user] password=[seekrit]"
        DBDMin  4
        DBDKeep 8
        DBDMax  20
        DBDExptime 300

        <Location /share>
                DAV svn
                SVNParentPath "/srv/svn"
                SVNAutoversioning on    # do auto checkout/replacement
                ModMimeUsePathInfo on   # guess the mime type

                AuthType Basic
                Authname "Subversion File Share"
                AuthBasicProvider dbd
                Require valid-user
                # mod_authn_dbd SQL query to authenticate a user
                # group_id can be determined by looking in the
                #   Administration groups page.
                AuthDBDUserPWQuery "SELECT hashed_password FROM users WHERE login = %s AND id IN (SELECT user_
id FROM groups_users WHERE group_id = 4)"
```

```
        </Location>
```

#11 - 2010-12-16 02:53 - Jerry Van Baren

P.S. I hardcoded the group ID "WHERE group_id = 4" in the SQL statement because the group ID will generally be a "figure out once, never changes" behavior. I didn't feel it was worth more looking up to make the group ID more user friendly.

#12 - 2011-03-24 08:41 - Toshi MARUYAMA

- Category set to SCM extra

Files

base64_hash.patch	2.42 KB	2010-01-23	Jerry Van Baren
base64_hash.patch	2.43 KB	2010-12-16	Jerry Van Baren