w.c.s. - Development #3201

création d'un type de champ "mot de passe"

02 juillet 2013 16:02 - Thomas Noël

Statut:	Fermé	**Début:**	02 juillet 2013
Priorité:	Haut	**Echéance:**	10 octobre 2014
Assigné à:	Frédéric Péters	**% réalisé:**	0%
Catégorie:		**Temps estimé:**	8:00 heures
Version cible:			
Patch proposed:	Oui	**Planning:**	

Description

Idée générale : permettre d'avoir des formulaires de demande de comptes pour des logiciels externes (mais aussi, pour nous, pour créer des comptes dans Authentic2).

Il nous manque pour cela un champ de type "password", qui affiche deux champs (le second pour vérifier). On ajoute un indicateur de force (en js, cf #2503) et un validateur de force (refus du mot de passe s'il est jugé en dessous d'un niveau => donc avoir le même algo de détection en Python que celui du JS).

Une option du champ permet de définir le stockage dans le formdata : chiffré (selon une liste de clés nommées et fournies dans site-options ?), hashé (shaX..., imaginer pouvoir générer plusieurs types de hash d'un coup ?), voire en clair.

Note : c'est une demande du CG14 dans le cadre du projet "accès cadastre"

Demandes liées:

Lié à w.c.s. - Development #5805: Utiliser le nouveau PasswordEntryWidget dan...	**Fermé**	**24 octobre 2014**

Historique

#1 - 02 juillet 2013 16:15 - Benjamin Dauvergne

Stocker directement la version hashée (voir plusieur comme tu dis) ça me parait le mieux; pour vous simplifiez la vie vous pouvez vous reposer sur les classes fournies par Django et les deux autres qu'on a nous même (w.c.s et Drupal); ça vous fait déjà une belle liste.

#2 - 02 juillet 2013 17:10 - Thomas Noël

Yep. Mais il faut aussi permettre de stocker une version cryptée au cas où on doive ensuite l'envoyer à un webservice (ou autre) qui nécessite d'avoir le mot de passe en clair pour créer le compte. Brefffllle...

Pour la liste Django :

('django.contrib.auth.hashers.PBKDF2PasswordHasher',
'django.contrib.auth.hashers.PBKDF2SHA1PasswordHasher',
'django.contrib.auth.hashers.BCryptPasswordHasher',
'django.contrib.auth.hashers.SHA1PasswordHasher',
'django.contrib.auth.hashers.MD5PasswordHasher',
'django.contrib.auth.hashers.UnsaltedMD5PasswordHasher',
'django.contrib.auth.hashers.CryptPasswordHasher',)

avec plein de code dans https://github.com/django/django/blob/master/django/contrib/auth/hashers.py

#3 - 02 juillet 2013 17:39 - Thomas Noël

En option du champ, prévoir un champ regex de validation, où on pourra mettre :

```
^(?=.*[A-Z].*[A-Z])(?=.*[!@#$&*])(?=.*[0-9].*[0-9])(?=.*[a-z].*[a-z].*[a-z]).{6,}$
```

(de http://stackoverflow.com/questions/5142103/regex-for-password-strength)

#4 - 04 juillet 2013 00:51 - Benjamin Dauvergne

redmine@entrouvert.com écrivait:

> La demande #3201 a été mise à jour par Thomas Noël.
>
> Yep. Mais il faut aussi permettre de stocker une version cryptée au cas où on doive ensuite l'envoyer à un webservice (ou autre) qui nécessite d'avoir le mot de passe en clair pour créer le compte.

> Brefflle...

Si la clé de chiffrement n'est définissable que via site-options.cfg et pas via l'admin web, ça protège de l'admin w.c.s qui modifie le workflow pour récupérer un mot de passe mais qui n'a pas d'accès au serveur d'hébergement par contre ça ne protège pas d'un admin unix indélicat.

Donc il faudrait que cet option ne soit utilisable que si cet clé est défini. À ce propose je me demande si ce ne serait pas utile de générer une 'secret_key' globale à l'installation du paquet w.c.s. un peu comme la secret-key dans les projets Django ou alors à l'initialisation d'un virtualhost par qommon on pourrait générer un site-options.cfg générique contenant la secret-key.

#5 - 04 juillet 2013 00:51 - Benjamin Dauvergne

redmine@entrouvert.com écrivait:

> En option du champ, prévoir un champ regex de validation, où on pourra mettre :
>
> ^(?=.*[A-Z].*[A-Z])(?=.*[@#$&*])(?=.*[0-9].*[0-9])(?=.*[a-z].*[a-z].*[a-z]).{6,}$
>
> (de http://stackoverflow.com/questions/5142103/regex-for-password-strength)

Think of the users please. C'est carrément obscure et je suis sûr que ça ne couvre pas des tas de politiques débiles sur la force des mots de passe (« ne contient pas un élément du login » vu explicitement sur le projet Gabriel de Formiris). Au pire je préfèrerai encore un champ expression python, c'est tout aussi obscure mais au moins on peut tout faire.

#6 - 04 juillet 2013 11:15 - Thomas Noël

Benjamin Dauvergne a écrit :

> Think of the users please. C'est carrément obscure et je suis sûr que ça ne couvre pas des tas de politiques débiles sur la force des mots de passe (« ne contient pas un élément du login » vu explicitement sur le projet Gabriel de Formiris). Au pire je préfèrerai encore un champ expression python, c'est tout aussi obscure mais au moins on peut tout faire.

Comme ta proposition se contredit elle-même, je reste sur mon idée d'une bête regex comme dans les champs textes. Et tant pis si c'est limité et obscur. Na.

(blague à part : la notion de contrainte "générique" sur les champs d'un formulaire ou d'une page, c'est une chose à travailler ailleurs, de façon globale, et pas seulement sur ce champ là).

#7 - 04 juillet 2013 12:15 - Thomas Noël

Benjamin Dauvergne a écrit :

> redmine@entrouvert.com écrivait:
> (...) Donc il faudrait que cet option ne soit utilisable que si cet clé est défini.

Yep, bonne idée.

> À ce propose je me demande si ce ne serait pas utile de générer une 'secret_key' globale à l'installation du paquet w.c.s. un peu comme la secret-key dans les projets Django ou alors à l'initialisation d'un virtualhost par qommon on pourrait générer un site-options.cfg générique contenant la secret-key.

Zou #3208 un peu pour mémoire et pour discussions

#8 - 20 mai 2014 14:52 - Thomas Noël

- Description mis à jour

#9 - 20 mai 2014 14:55 - Thomas Noël

Note dans le cadre CG14 : les mots de passe seront à fournir en hash "Blowfish-based, variant 2a" (dans leur base actuelle ils utilisent dans la requête postgresql : crypt('motdepasse', gen_salt('bf', 8))).

Petit soucis : je ne trouve pas d'algo de hash "blowfish" dans Debian Wheezy.

#10 - 03 octobre 2014 09:26 - Thomas Noël

- Echéance mis à 10 octobre 2014

- Priorité changé de Normal à Haut

- Patch proposed mis à Non

#11 - 03 octobre 2014 16:56 - Frédéric Péters

- Fichier 0001-password-field-draft.patch ajouté

Un patch brouillon (sommairement testé avec postgresql en stockage), qui met le strict minimum en place, sur lequel ajouter les options qu'on voudrait.

#12 - 24 octobre 2014 15:32 - Thomas Noël

Dans les options :

- les options de force de mot de passe wcs : min_length, max_length, count_uppercase (minimum de majuscules), count_lowercase, count_digit, count_special
- possibilité de dire une liste des hashs sous lesquels on veut stocker le mot de passe (aucun, sha1, sha256, md5, etc) dans l'optique de pouvoir les envoyer à des systèmes externes ensuite
  - la variable "password_raw" serait un dico : {'md5': ..., 'sha1': ...}
  - l'affichage restant toujours "*******"

(pour le coup de password_raw, je dis ça sans avoir regardé si c'est possible)

#13 - 24 octobre 2014 16:12 - Frédéric Péters

- Assigné à changé de Thomas Noël à Frédéric Péters

Je reprends un peu la main.

#14 - 24 octobre 2014 16:54 - Frédéric Péters

- Fichier 0001-fields-password-field-3201.patch ajouté

- Statut changé de Nouveau à En cours

- Patch proposed changé de Non à Oui

Voilà avec quand même deux vérifications pour tester (longueur min et max).

Au niveau des variables de substitutions, le dict est mis à plat, ça donne par exemple :

- 'form_var_pwd_sha1': 'e0c9035898dd52fc65c41454cec9c4d2611bfb37'
- 'form_var_pwd_plain': 'aa'

Dans la sortie json, le dict est visible, exemple :


```
{"display_id": 2, "fields": {"pwd": {"plain": "aa", "sha1": "e0c9035898dd52fc65c41454cec9c4d2611bfb37"}},
  "receipt_time": "2014-10-24T16:35:30", "id": "password/2", "workflow": {"status": {"id": "new", "name": "Nou
veau"}}}
```


Dans PostgreSQL c'est stocké dans une colonne text[][], ça donne :

```
wcs=> select f2 from formdata_103_password;
                              f2
-----------------------------------------------------------------
 {{plain,aa},{sha1,e0c9035898dd52fc65c41454cec9c4d2611bfb37}}
```

Un truc que j'ai hésité à faire c'est chiffrer via une clé secrète ce qui est poussé vers le navigateur pour la page de confirmation, j'ai laissé ça pour plus tard (pour le moment c'est base64(json(x))).

Pour le moment j'ai juste mis plain, md5 et sha1.

On pourrait aussi se dire que si on a "plain", un changement au champ pourrait mettre tous les hashs à jour, je passe.

Ce n'est pas testé en stockage conventionnel.

#15 - 24 octobre 2014 18:10 - Thomas Noël

Frédéric Péters a écrit :

> Pour le moment j'ai juste mis plain, md5 et sha1.

Ca suffira pour la première release.

> On pourrait aussi se dire que si on a "plain", un changement au champ pourrait mettre tous les hashs à jour, je passe.

Pas important (on n'a pas d'effet rétro-actif dans wcs pour l'instant).

> Ce n'est pas testé en stockage conventionnel.

Testé, marche sans problème.

En revanche pépin d'export, il faudrait ajouter :

```
    def get_csv_value(self, value, hint=None):
        return ['*'*8]
```

#16 - 24 octobre 2014 19:09 - Frédéric Péters

Ok, je continue avec les autres paramètres et ajoute le get_csv_value.

#17 - 24 octobre 2014 19:47 - Frédéric Péters

- Fichier 0001-fields-password-field-3201.patch ajouté

Voilà une nouvelle version du patch avec tous les paramètres ainsi que le widget javascript d'évaluation de force de mot de passe, celui-ci est configuré pour prendre en compte la longueur minimale mais au-delà de ça son algorithme est tout différent, je ne pense pas que ça soit bien grave mais je modifierais un peu la traduction pour transformer "Trop faible" en "Très faible".

J'ai aussi un patch pour utiliser ce new PasswordEntryWidget dans les vues d'enregistrement et de mot de passe oublié, mais j'en ferai un ticket séparé.

#18 - 25 octobre 2014 16:36 - Thomas Noël

- Fichier 0001-fields-password-field-3201.patch ajouté

Testé, ça me parait très bien, ack !

Frédéric Péters a écrit :

> Voilà une nouvelle version du patch avec tous les paramètres ainsi que le widget javascript d'évaluation de force de mot de passe, celui-ci est configuré pour prendre en compte la longueur minimale mais au-delà de ça son algorithme est tout différent, je ne pense pas que ça soit bien grave mais je modifierais un peu la traduction pour transformer "Trop faible" en "Très faible".

D'accord pour "très faible" (parce que "ton mot de passe il est trop faible", les djeunes vont mal capter).

Puis il faudra effectivement voir comment ajouter à ce javascript les conditions wcs (nombre de de majuscules, etc). Mais en attendant c'est déjà "pushable" tel quel puisque les conditions sont de toute façon imposées (et elles peuvent être précisées dans le "hint"). Bref, à voir + tard dans le cadre d'un autre ticket.

J'ai joué sur ce patch, voici ma version attachée avec :

- faire en sorte que le "hint" s'affiche lié au pwd1 (c'est un peu bidouille mais j'ai pas trouvé autrement)
- permettre un choix de titre pour le pwd2 (au lieu du simple "Confirmation" par défaut).

#19 - 25 octobre 2014 16:42 - Frédéric Péters

Pour le hint, tu peux juste faire hint = kwargs.pop('hint', None). La ligne self.count_special = kwargs.get('count_special', 0) est dupliquée. C'était déjà dans mon mon patch mais le readonly=self.attrs.get('readonly'), est inutile vu la condition posée juste au-dessus.

#20 - 25 octobre 2014 16:59 - Thomas Noël

- Fichier 0001-fields-password-field-3201.patch ajouté

Frédéric Péters a écrit :

> Pour le hint, tu peux juste faire hint = kwargs.pop('hint', None).

On n'avait pas dit qu'on restait en Python 1.5 ? ;)

Voici le patch avec ces corrections. Je te laisse pousser si c'est bon pour toi.

#21 - 27 octobre 2014 10:34 - Frédéric Péters

- Statut changé de En cours à Résolu (à déployer)

Il a fallu que j'installe une woody pour trouver un Python 1.5 et constater que tu avais raison; damned.

Pour ma peine j'ai ajouté un peu de tests.

```
commit 989d5f8a64c37b7ae6cdbdc31a2b016428e3befb
Author: Frédéric Péters <fpeters@entrouvert.com>
Date:   Fri Oct 3 16:53:35 2014 +0200

    fields: password field (#3201)
```

#22 - 22 décembre 2014 14:07 - Thomas Noël

- Statut changé de Résolu (à déployer) à Fermé

Fichiers

0001-password-field-draft.patch	2,85 ko	03 octobre 2014	Frédéric Péters
0001-fields-password-field-3201.patch	6,62 ko	24 octobre 2014	Frédéric Péters
0001-fields-password-field-3201.patch	10,3 ko	24 octobre 2014	Frédéric Péters
0001-fields-password-field-3201.patch	10,9 ko	25 octobre 2014	Thomas Noël
0001-fields-password-field-3201.patch	10,7 ko	25 octobre 2014	Thomas Noël