Fargo - Development #16842

Déléguer l'authentification des clients à authentic.

12 juin 2017 15:52 - Mikaël Ates

Statut:	Fermé	**Début:**	12 juin 2017
Priorité:	Normal	**Echéance:**	
Assigné à:	Josué Kouka	**% réalisé:**	100%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Oui	**Planning:**	

Description

Permettre que fargo puisse s'appuyer sur authentic pour la vérification des identifiants des clients, par exemple les couples client_id et client_secret des clients OAUTH2.

Demandes liées:

Lié à Authentic 2 - Development #16580: Ajouter une authentification pour DRF...	**Fermé**	**29 mai 2017**
Lié à Authentic 2 - Development #16583: Ajouter une API check-password	**Fermé**	**29 mai 2017**
Lié à Fargo - Development #16192: ajouter dépôt de document en oauth2	**Fermé**	**05 mai 2017**

Révisions associées

Révision 94dab06b - 06 mars 2018 12:05 - Josué Kouka

misc: move some util functions in a utils.py file (#16842)

Révision 85ebba83 - 06 mars 2018 12:08 - Josué Kouka

api: use DRF for OAUTH2 APIs (#16842)

Révision e22648dd - 06 mars 2018 12:08 - Josué Kouka

api: authenticate OAUTH2 clients through Authentic (fixes #16842)

Historique

#2 - 12 juin 2017 15:53 - Mikaël Ates

- Lié à Development #16580: Ajouter une authentification pour DRF basée sur les client_id/client_secret des clients OIDC ajouté

#3 - 12 juin 2017 15:53 - Mikaël Ates

- Lié à Development #16583: Ajouter une API check-password ajouté

#4 - 05 juillet 2017 17:46 - Josué Kouka

- Lié à Development #16192: ajouter dépôt de document en oauth2 ajouté

#5 - 18 décembre 2017 14:54 - Paul Marillonnet

Est-ce qu'on encore ici des cas d'usages qui sortent du support OAuth2 dans Fargo ?

#6 - 19 janvier 2018 19:22 - Josué Kouka

- Fichier 0001-use-drf-for-oauth2-APIs-16842.patch ajouté

- Fichier 0002-add-rp-remote-idp-authentication-16842.patch ajouté

- Patch proposed changé de Non à Oui

Voila, un patch.
Je suis parti de l'idée d'avoir une seule classe drf pour l'authentification locale et distante. Du coup dans:

- le 0001 je change les vues get_document_token et put_document en des vues drf.
- le 002 je rajoute l'authentication distante.

#7 - 19 janvier 2018 19:22 - Josué Kouka

Paul Marillonnet a écrit :

> Est-ce qu'on encore ici des cas d'usages qui sortent du support OAuth2 dans Fargo ?

Non, je ne pense pas.

#8 - 19 janvier 2018 19:22 - Josué Kouka

- Statut changé de Nouveau à En cours

#9 - 30 janvier 2018 16:29 - Benjamin Dauvergne

Ça ne ressemble pas à ce qui est fait dans petale, il faut faire pareil, voir petale/authentication.py

#10 - 30 janvier 2018 16:30 - Benjamin Dauvergne

Notamment hériter de BasicAuthentication.

#11 - 30 janvier 2018 16:33 - Benjamin Dauvergne

Et je découperai un peu plus, d'abord on bouge tout dans fargo/utils.py, ensuite on remplace les vues par des vues à base de classes, ensuite on vire authenticate_client() pour une classe d'authentification basée sur BasicAuthentication, ensuite on ajoute la délégation à authentic.

#12 - 31 janvier 2018 17:28 - Josué Kouka

- Fichier 0002-use-drf-for-oauth2-APIs-16842.patch ajouté

- Fichier 0001-misc-move-some-util-functions-in-a-utils.py-file-168.patch ajouté

- Fichier 0003-add-rp-remote-idp-authentication-16842.patch ajouté

> Ça ne ressemble pas à ce qui est fait dans petale, il faut faire pareil, voir petale/authentication.py
> Notamment hériter de BasicAuthentication.

OK c'est ce qui est fait **en parti** car le backend d'authentification diffère ici (OAUTH2Client)

> Et je découperai un peu plus, d'abord on bouge tout dans fargo/utils.py, ensuite on remplace les vues par des vues à base de classes, ensuite on vire authenticate_client() pour une classe d'authentification basée sur BasicAuthentication, ensuite on ajoute la délégation à authentic.

Ok, dans le:

- 0001: fonctions dans utils.py
- 0002: passage à drf avec utilisation d'une classe d'authentication basée sur BasicAuthentication et suppression du de authenticate_client. Je n'ai pas splitter plus que ça parce que je me suis dit qu'avec l'introduction de l'APIView autant définir la classe d'authentification en meme temps.
- 0003: ajout de l'authentification via l'idp.

#13 - 31 janvier 2018 18:30 - Benjamin Dauvergne

Je vais relire ça ce soir.

#14 - 31 janvier 2018 18:34 - Frédéric Péters

> 0003: ajout de l'authentification via l'idp.

J'étais curieux sur cette API utilisée côté authentic j'ai regardé et il me semble qu'on a une exigence sur une permission particulière :

```
class CheckPasswordAPI(BaseRpcView):
    permission_classes = (DjangoPermission('custom_user.search_user'),)
    serializer_class = CheckPasswordSerializer
```

mais ici :

```
            response = requests.post(url, json={
                'username': client_id,
                'password': client_secret}, auth=(client_id, client_secret), verify=False)
```

l'appel se fait en http basic auth avec les identifiants de l'utilisateur qu'on veut checker, qui n'a sans doute pas custom_user.search_user.

(et verify=False…)

Ça a été testé en local avec un vrai authentic et autre chose qu'un compte admin ?

#15 - 31 janvier 2018 18:42 - Josué Kouka

Frédéric Péters a écrit :

> 0003: ajout de l'authentification via l'idp.

J'étais curieux sur cette API utilisée côté authentic j'ai regardé et il me semble qu'on a une exigence sur une permission particulière :

[...]

mais ici :

[...]

l'appel se fait en http basic auth avec les identifiants de l'utilisateur qu'on veut checker, qui n'a sans doute pas custom_user.search_user.

(et verify=False…)

Ça a été testé en local avec un vrai authentic et autre chose qu'un compte admin ?

Le droit d'accès à l'API pour les clients OIDC est défini au niveau du model avec has_api_access.
Ensuite pour les permissions un (Fake User) OIDCUser est renvoyé avec toutes les permissions accordées (tout est dans src/authentic/authentication)

#16 - 31 janvier 2018 18:50 - Frédéric Péters

> Ça a été testé en local avec un vrai authentic et autre chose qu'un compte admin ?

#17 - 31 janvier 2018 18:51 - Benjamin Dauvergne

C'est moi qui ait dit des bêtises à Josué, il faut effectivement un compte particulier pour accéder à cette API, sur CUT/pétale j'utilise des credentials OIDC juste pour cela (sachant qu'on a une authentification pour les APIs basée dessus qui file toutes les permissions).

Donc Josué première remarque, faudrait un setting FARGO_AUTHENTIC_AUTH en plus du FARGO_AUTHENTIC_URL, idem voir petale.

#18 - 31 janvier 2018 18:52 - Benjamin Dauvergne

En fait je suis con, ça va fonctionner ce que fait Josué, d'ailleurs on s'est fait chier pour rien sur petale.

#19 - 31 janvier 2018 18:54 - Frédéric Péters

L'API check_password, elle est appelée ici comme pourrait être appelée une API ping qui ne ferait rien, la question du check des credentials étant déjà assurée avant, c'est bien ça ?

#20 - 31 janvier 2018 19:27 - Benjamin Dauvergne

Ici effectivement les mêmes credentials servent aux deux, mais l'API a bien un usage si je veux tester les identifiants d'un autre utilisateur qui lui n'a pas le droit d'accéder à cette API. Pour l'instant elle ne fait rien de très intéressant cette API à part dire oui ou non, mais on pourrait imaginer renvoyer un cookie de session à utiliser par une appli mobile.

#21 - 31 janvier 2018 20:24 - Josué Kouka

Frédéric Péters a écrit :

> Ça a été testé en local avec un vrai authentic et autre chose qu'un compte admin ?

Oui oui.

#22 - 06 mars 2018 12:10 - Josué Kouka

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

- % réalisé changé de 0 à 100

Appliqué par commit e22648dd3f34de04d1ace5e7928c00fbf646000d.

#23 - 06 mars 2018 12:13 - Benjamin Dauvergne

- Statut changé de Résolu (à déployer) à Fermé

Fichiers

0001-use-drf-for-oauth2-APIs-16842.patch	13,7 ko	19 janvier 2018	Josué Kouka
0002-add-rp-remote-idp-authentication-16842.patch	5,71 ko	19 janvier 2018	Josué Kouka
0002-use-drf-for-oauth2-APIs-16842.patch	10,1 ko	31 janvier 2018	Josué Kouka
0003-add-rp-remote-idp-authentication-16842.patch	5,67 ko	31 janvier 2018	Josué Kouka
0001-misc-move-some-util-functions-in-a-utils.py-file-168.patch	6,59 ko	31 janvier 2018	Josué Kouka