Combo - Bug #30723

Recherche d'usagers sur une multicollectivité

18 février 2019 10:03 - Marie Kuntz

Statut:	Fermé	**Début:**	18 février 2019
Priorité:	Normal	**Echéance:**	
Assigné à:	Frédéric Péters	**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Oui	**Planning:**	

Description

Sur l'agglo de Clisson (multicollectivités), sur le portail agents de l'agglo (collectivité par défaut), la recherche d'usager me mène bien à sa fiche, en revanche sur le portail agents d'une ville, la même recherche m'amène sur ma propre fiche.

Pour reproduire :

- aller sur https://agents-csma.test.entrouvert.org/
- chercher test1 : on arrive sur le profil de test1
- aller sur https://agents-clisson.test.entrouvert.org/
- chercher test1 : on arrive sur sa propre fiche

Pour info je n'ai pas réussi à reproduire sur Lille (multicollectivité également)

Révisions associées

Révision b5a8e0cd - 04 mars 2019 13:42 - Frédéric Péters

misc: monkeypatch user model with a method to get name id (#30723)

Révision cfbf0383 - 04 mars 2019 13:42 - Frédéric Péters

general: use a fake proxy object for unknown local NameIDs (#30723)

Historique

#1 - 18 février 2019 10:17 - Frédéric Péters

(je complète pour moi)

> aller sur https://agents-clisson.test.entrouvert.org/
> chercher test1 : ...

on voit bien les infos du compte test1, mais cliquer dessus et ...

> ... on arrive sur sa propre fiche

URLs :

- https://agents-csma.test.entrouvert.org/fiche-usager/089666e8d39548f79e107e92099bee12/ (affiche "test1")
- https://agents-clisson.test.entrouvert.org/fiche-usager/089666e8d39548f79e107e92099bee12/ (affiche, dans mon cas, "fred")

(on note le même uuid)

#2 - 18 février 2019 15:50 - Frédéric Péters

- Statut changé de Nouveau à En cours

Ça arrive parce que l'usager en question, attaché à l'interco, n'a pas été provisionné sur le combo de la commune (attaché à la commune); parce qu'on n'attache comme utilisateur que les comptes connus :

```
    if 'name_id' in ctx and UserSAMLIdentifier:
        try:
            ctx['selected_user'] = UserSAMLIdentifier.objects.get(name_id=ctx['name_id']).user
        except UserSAMLIdentifier.DoesNotExist:
            pass
```

De manière peut-être inappropriée, en tout cas très locale, si on ne cherche pas à réfléchir sur le provisionning,

```
--- a/combo/profile/models.py
+++ b/combo/profile/models.py
@@ -43,7 +43,7 @@ class ProfileCell(JsonCellBase):
     @property
     def url(self):
         idp = settings.KNOWN_SERVICES.get('authentic').values()[0]
-        return '{%% load combo %%}%sapi/users/{{ concerned_user|name_id }}/' % idp.get('url')
+        return '{%% load combo %%}%sapi/users/{%% firstof name_id concerned_user|name_id %%}/' % idp.get('url
')

     def is_visible(self, user=None):
         if not user or user.is_anonymous():
```

#3 - 21 février 2019 08:21 - Frédéric Péters

- Fichier 0001-profile-look-at-explicit-nameid-context-to-get-user-.patch ajouté

- Projet changé de Publik à Combo

- Statut changé de En cours à Solution proposée

- Patch proposed changé de Non à Oui

Sans autre idée, voilà ce diff sous forme de patch.

#4 - 21 février 2019 10:32 - Thomas Noël

Mais de fait, c'est le principe du "concerned_user" qui foire et ne permet pas d'avoir les cellules "données de cet usager" pour les autres cellules liées à un usager (combo/apps/family, combo/apps/wcs, combo/apps/fargo)... non ?

#5 - 21 février 2019 10:40 - Frédéric Péters

Le "soucis" c'est que les cellules concernant un usager demandent un usager connu localement, ne peuvent pas se fier à un uuid lancé en l'air.

```
    if 'name_id' in ctx and UserSAMLIdentifier:
        try:
            ctx['selected_user'] = UserSAMLIdentifier.objects.get(name_id=ctx['name_id']).user
        except UserSAMLIdentifier.DoesNotExist:
            pass
```

pourrait être modifié pour avoir à la place du "pass" un ctx['selected_user'] = FakeTransientUser(name_id=...); en s'assurant que cet objet contienne le nécessaire pour être accepté à différents endroits.

#6 - 21 février 2019 10:49 - Frédéric Péters

- Statut changé de Solution proposée à En cours

- Patch proposed changé de Oui à Non

Et ça fait un bel exemple de point où ce serait mille fois plus joli/confortable de pouvoir profiter de Django 1.11...
https://docs.djangoproject.com/en/1.11/topics/db/models/#proxy-models

#7 - 21 février 2019 10:53 - Frédéric Péters

- Assigné à mis à Frédéric Péters

#8 - 21 février 2019 12:31 - Frédéric Péters

- Fichier 0002-general-use-a-fake-proxy-object-for-unknown-local-Na.patch ajouté

- Fichier 0001-misc-monkeypatch-user-model-with-a-method-to-get-nam.patch ajouté

- Statut changé de En cours à Solution proposée

- Patch proposed changé de Non à Oui

Deux patchs, monkeypatch pour ajouter un get_name_id et utilisation un peu partout, puis ajout du fallback sur objet bidon.

#9 - 21 février 2019 16:20 - Frédéric Péters

Thomas me pointe qu'en fait les modèles proxy existaient déjà en 1.8, je ne sais plus d'où j'avais l'idée inverse.

Mais en passant, comme j'ai trouvé get_user_model().add_to_class() qui est du monkeypatch pas trop dégueu, ça me va de rester ainsi.

#10 - 21 février 2019 16:34 - Frédéric Péters

> Mais en passant, comme j'ai trouvé get_user_model().add_to_class() qui est du monkeypatch pas trop dégueu, ça me va de rester ainsi.

Surtout qu'en fait, en essayant quand même, ça échoue, ça ne semble pas bien se mixer le proxy let modèle User (User cannot proxy the swapped model 'profile.User').

#11 - 01 mars 2019 15:35 - Emmanuel Cazenave

Je n'arrive pas à voir si c'est un cas réaliste mais get_user_from_name_id ne renvoie None que si mellon n'est pas là, et dans lingo :

```
user = get_user_from_name_id(request.GET.get('NameId'))
if user is None:
   raise User.DoesNotExist()
.....
item = BasketItem.objects.get(id=request_body.get('basket_item_id'),
                user=user, cancellation_date__isnull=True)
```

Et donc erreur 500 si le nameId passé correspond à un utilisateur qui ne s'est jamais authentifié dans combo ?

#12 - 01 mars 2019 15:49 - Benjamin Dauvergne

Je vous lis, mais je ne comprend pas pourquoi on cherche localement un utilisateur dont toutes les données vont être prises à distance, ça a du être une facilité de porter des cellules faites initialement pour le portail usager je suppose.

#13 - 01 mars 2019 15:55 - Frédéric Péters

- Fichier 0002-general-use-a-fake-proxy-object-for-unknown-local-Na.patch ajouté

- Fichier 0001-misc-monkeypatch-user-model-with-a-method-to-get-nam.patch ajouté

Peut-être que la situation peut arriver, d'un site interco liant directement une démarche demandant authentification sur un site communal, ça y créera l'utilisateur via le SSO, donc uniquement sur wcs, et si derrière il y a une action d'ajout au panier, ça échouera. (comme aujourd'hui, sauf si ma lecture du code de provisionning est erronée). (il faudrait qu'un usager qui ait fait un SSO vers le service d'une OU soit provisionné sur tous les services de cette OU, à vérifier avec Benj)

Cela étant, quand même, parce que les API de panier ont besoin in fine d'un vrai utilisateur, version modifiée pour avoir une version de get_user_from_name_id qui raise plutôt que retourner un faux utilisateur proxy.

#14 - 01 mars 2019 17:22 - Emmanuel Cazenave

Le truc ci-dessous pourrait se rajouter dans tests_profile.py (c'est un peu tordu mais ça passe dans du code de tes patchs), et pourquoi pas dans d'autre cellules concernées ?

```
ctx = {'synchronous': True, 'name_id': '6666'}
modify_global_context(None, ctx)
context = cell.get_cell_extra_context(ctx)
assert context['profile_fields']['first_name']['value'] == 'Foo'
assert context['profile_fields']['birthdate']['value'] == datetime.date(2018, 8, 10)
assert requests_get.call_args[0][0] == 'http://example.org/api/users/6666/'
```

#15 - 01 mars 2019 19:13 - Emmanuel Cazenave

Et à vrai dire, pour rebondir sur la remarque de Benjamin et sur l'idée de départ return '{%% load combo %%}%sapi/users/{%% firstof name_id concerned_user|name_id %%}/' % idp.get('url'), je trouverais ça plus lisible qu'on touche aux cellules concernées pour qu'elles puissent utiliser tel quel un name_id qui serait dans leur contexte.

Les contorsions pour maintenir l'abstraction du concerned_user sont un peu dure à suivre.

#16 - 01 mars 2019 20:28 - Frédéric Péters

Benjamin :

> ça a du être une facilité de porter des cellules faites initialement pour le portail usager je suppose.

L'objectif était bien de pouvoir utiliser dans le portail agent les cellules déjà développées pour le côté front, en utilisant ce qui existait déjà, c'est-à-dire get_concerned_user().

Emmanuel :

> (...) je trouverais ça plus lisible qu'on touche aux cellules concernées pour qu'elles puissent utiliser tel quel un name_id qui serait dans leur

> contexte.

J'essaie de comprendre ce que tu vois, sans être bien sûr. J'imagine, tu peux voir l'objet User comme une enveloppe plus ou moins remplie autour du NameID, et logique alors, on pourrait se dire quelle nécessité à l'enveloppe, plutôt porter directement le NameID. C'est assez raisonnable, et sans doute que mon patch encourage cette lecture, en réunissant les différents endroits qui dupliquaient le code permettant d'obtenir un NameID à partir d'un User.

Mais avoir dans le contexte le user, ça reste de toute façon nécessaire pour les cellules fonctionnant avec les données de combo, les cellules notification, tableau de bord, panier, etc. Et ça veut donc dire qu'une cellule qui a besoin d'appeler un service tiers, elle doit le faire avec get_name_id(context['user']) plutôt que juste context['user_name_id'], ça ne me semble pas si problématique.

Aussi, même si ça devrait pouvoir être supprimé, il reste une cellule comme la cellule d'abonnement aux newsletters, qui appelle un service tiers, mais en utilisant l'email comme référence, pas le NameID.

On peut prendre du temps à l'occasion pour passer sur tout ça, au moins mettre par écrit le fonctionnement actuel, réfléchir à comment ça pourrait évoluer, mais pour le moment, ces patchs 1/ corrigent un vrai bug, signalé, 2/ refactorisent avec succès le code, vu que la part de duplication diminue.

#17 - 04 mars 2019 11:08 - Emmanuel Cazenave

Frédéric Péters a écrit :

> J'essaie de comprendre ce que tu vois, sans être bien sûr.

J'ai juste l'impression que pour les cellules qui vont chercher leur données auprès d'un service externe sur la base du name_id, on pourrait s'en sortir en gérant de la façon suivante, que ça couvrirait les cas portail usager/agent, mono/multicollectivité, utilisateur provisionné ou pas :

- si il y a un name_id dans mon contexte prendre celui là
- sinon prendre le name_id de l'utilisateur authentifié

Et de là j'imagine un patch simplissime, genre une méthode :

```
def get_name_id(context, request):
    if 'name_id' in context:
        return context['name_id']
    return request.user.name_id
```

que les cellules concernées n'auraient qu'appeler, et voilà, comme disent les américains.

Je loupe peut-être quelque chose, etc, etc.

#18 - 04 mars 2019 11:27 - Frédéric Péters

> Je loupe peut-être quelque chose, etc, etc.

L'ajout de NameID aux appels se fait déjà de manière "automatique" à un seul endroit, dans combo/utils/request_wrapper.py, sur base d'un objet User.

#19 - 04 mars 2019 12:48 - Emmanuel Cazenave

Non mais dis donc, dans des endroits au on a accès au contexte :

```
return '/api/users/%s/drafts' % user_name_id
....
return '/api/users/%s/forms' % user_name_id
......
return '{%% load combo %%}%sapi/users/{{ concerned_user|name_id }}/' % idp.get('url')
```

Bref c'est quand même plus compliqué que ce que j'imaginais donc je retiens ton argument de non duplication.

(je trouve que ça mérite quand même largement des tests unitaires)

#20 - 04 mars 2019 12:48 - Emmanuel Cazenave

- Statut changé de Solution proposée à Solution validée

#21 - 04 mars 2019 13:44 - Frédéric Péters

- Statut changé de Solution validée à Résolu (à déployer)

Non mais dis donc, dans des endroits au on a accès au contexte :

Ces points sont particuliers dans w.c.s., parce qu'on a double besoin d'une info utilisateur, à la fois l'info sur l'agent qui fait l'appel et sur l'usager concerné.

Pour une cellule / un connecteur normal, il doit juste y avoir un appel à requests.get(url, user=...), avec tout le temps la même URL.

```
commit cfbf038356b625acf923b71bab311ef63d217b98
Author: Frédéric Péters <fpeters@entrouvert.com>
Date:   Thu Feb 21 11:20:09 2019 +0100

    general: use a fake proxy object for unknown local NameIDs (#30723)

commit b5a8e0cda4b9882f9aa82f28122c9fa9f838a5fa
Author: Frédéric Péters <fpeters@entrouvert.com>
Date:   Thu Feb 21 11:18:26 2019 +0100

    misc: monkeypatch user model with a method to get name id (#30723)
```

#22 - 04 mars 2019 16:16 - Frédéric Péters

- Statut changé de Résolu (à déployer) à Solution déployée

Fichiers

0001-profile-look-at-explicit-nameid-context-to-get-user-.patch	963 octets	21 février 2019	Frédéric Péters
0002-general-use-a-fake-proxy-object-for-unknown-local-Na.patch	1,03 ko	21 février 2019	Frédéric Péters
0001-misc-monkeypatch-user-model-with-a-method-to-get-nam.patch	19,6 ko	21 février 2019	Frédéric Péters
0002-general-use-a-fake-proxy-object-for-unknown-local-Na.patch	2,54 ko	01 mars 2019	Frédéric Péters
0001-misc-monkeypatch-user-model-with-a-method-to-get-nam.patch	19,6 ko	01 mars 2019	Frédéric Péters