Authentic 2 - Development #21870

oidc : configuration pour les attributs liés aux scopes

15 février 2018 11:43 - Frédéric Péters

Statut:	Fermé	**Début:**	15 février 2018
Priorité:	Normal	**Echéance:**	
Assigné à:	Josué Kouka	**% réalisé:**	100%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Oui	**Planning:**	

Description

create_user_info() dans src/authentic2_idp_oidc/utils.py hardcode quelques attributs pour les scopes oidc profile et email; côté CUT c'est étendu via le hook idp_oidc_modify_user_info mais peut-être qu'il pourrait quand même déjà exister une mécanique de base pour étendre ça sans code; en ayant les scopes sous forme d'objets qui pourraient ainsi être modifiés (comme on peut configurer côté SAML les attributs envoyés), ou en ayant les scopes dans les settings.

Demandes liées:

Lié à Authentic 2 - Bug #23544: Ajouter la possibilité de demander un CLAIM v...	**Nouveau**	**02 mai 2018**

Révisions associées

Révision f06900ea - 02 mai 2018 17:37 - Josué Kouka

idp oidc: add extra attributes configuration (#21870)

Historique

#4 - 09 avril 2018 17:42 - Josué Kouka

- Fichier 0001-idp-oidc-allow-extension-of-user-info-21870.patch ajouté

- Statut changé de Nouveau à En cours

- Assigné à mis à Josué Kouka

- Patch proposed changé de Non à Oui

C'est un patch draft. Pour l'instant il ne permet l'extension des user info via un settings A2_IDP_OIDC_USER_INFO_EXTRA pour le domaine profile. Pour les scopes email et roles je ne vois pas trop de cas d'utilisation ou l'on voudrait modifier les infos retournées (il y'en a peut etre).
Je me dis que l'on pourrait vouloir supprimer des attributs retournés, du coup je me demande si une config comme celle ci dessous conviendrait ?:

```
{
  "<scope>": {
      "add": [...],
      "remove": [...]
  }
}
```

#5 - 09 avril 2018 17:49 - Benjamin Dauvergne

Je suis contre ce ticket, c'est soit on fait les choses bien, et donc stocker avec le reste de la configuration OIDC, soit on reste sur le hook pour l'instant.

#6 - 09 avril 2018 17:56 - Josué Kouka

Benjamin Dauvergne a écrit :

> Je suis contre ce ticket, c'est soit on fait les choses bien, et donc stocker avec le reste de la configuration OIDC

Ok, par config oidc tu entends config du client ?

#7 - 09 avril 2018 21:55 - Benjamin Dauvergne

Oui, un modèle du genre:

```
class OIDCAttribute
   client = FK(OIDCClient)
   name = CharField() # "adress.street" <- sera déplié
```

```
   value = CharField() # le nom d'un attribut comme coté SAML
   scope = CharField() # une liste de scopes dont il fait partie
```

idéalement ça devrait gérer le cas où un attribut est vérifié en servant le même nom avec le suffixe _verified comme le prévoit la norme.

Une migration de donnée devrait ajouter les attributs existants comme OIDCAttribute et on pourrait ensuite supprimer le code en dur.

#8 - 13 avril 2018 09:36 - Josué Kouka

- Fichier 0001-idp-oidc-add-extra-attributes-configuration-21870.patch ajouté

Benjamin Dauvergne a écrit :

> Oui, un modèle du genre:
>
> [...]
>
> idéalement ça devrait gérer le cas où un attribut est vérifié en servant le même nom avec le suffixe _verified comme le prévoit la norme.
>
> Une migration de donnée devrait ajouter les attributs existants comme OIDCAttribute et on pourrait ensuite supprimer le code en dur.

Ok.
Un patch un peu brouillon de ce qui est expliqué plus haut. Je rencontre un souci lors de la récupération des valeurs de certains attributs. J'ai un test qui échoue parce que authentic2.attributes_ng.engine.get_attributes ne me renvoie pas la bonne valeur. Dans le cas ou je demande à renvoyer django_user_ou je suis censé avoir en retour une valeur sérialisée, mais ce n'est pas la cas ou quand je demande a2_role_names je reçois une QuerySet à la place des noms de roles.
Je creuse pour essayer de comprendre le pourquoi.

#9 - 13 avril 2018 10:41 - Benjamin Dauvergne

Oui le code est un peu brut:

```
def get_attributes(instance, ctx):
    user = ctx.get('user')
    User = get_user_model()
    if not user or not isinstance(user, User):
        return ctx
    for field in User._meta.fields:
        value = getattr(user, field.name)
        if value is None:
            continue
        ctx['django_user_' + str(field.name)] = getattr(user, field.name) # pour django_user_ou on va obtenir
le modèle directement
    for av in AttributeValue.objects.with_owner(user):
        ctx['django_user_' + str(av.attribute.name)] = av.to_python()
        ctx['django_user_' + str(av.attribute.name) + ':verified'] = av.verified
    ctx['django_user_groups'] = [group for group in user.groups.all()]
    ctx['django_user_group_names'] = [unicode(group) for group in user.groups.all()]
    if user.username:
        splitted = user.username.rsplit('@', 1)
        ctx['django_user_domain'] = splitted[1] if '@' in user.username else ''
        ctx['django_user_identifier'] = splitted[0] if '@' in user.username else ''
    ctx['django_user_full_name'] = user.get_full_name()
    Role = get_role_model()
    roles = Role.objects.for_user(user)
    ctx['a2_role_slugs'] = roles.values_list('slug', flat=True)
    ctx['a2_role_names'] = roles.values_list('name', flat=True)
    ctx['a2_role_uuids'] = roles.values_list('uuid', flat=True)
    if 'service' in ctx and getattr(ctx['service'], 'ou', None):
        ou = ctx['service'].ou
        ctx['a2_service_ou_role_slugs'] = roles.filter(ou=ou).values_list('slug', flat=True)
        ctx['a2_service_ou_role_names'] = roles.filter(ou=ou).values_list('name', flat=True) <-- ValuesQuerySe
t
        ctx['a2_service_ou_role_uuids'] = roles.filter(ou=ou).values_list('uuid', flat=True)
    return ctx
```

Coté SAML je suppose que ça applique unicode() là dessus ou que ça génère plusieurs valeurs si ça détecte une liste, et oui c'est ça (ctx est le résultat de l'appel à get_attributes()).

```
    tuples = [tuple(t) for definition in qs for t in definition.to_tuples(ctx)]
    seen = set()
    for name, name_format, friendly_name, value in tuples:
```

```
    def to_tuples(self, ctx):
        if not self.attribute_name in ctx:
            return
        name_format = self.name_format_uri()
        name = self.name
        friendly_name = self.friendly_name or None
        values = ctx.get(self.attribute_name)
        for text_value in normalize_attribute_values(values):
            yield (name, name_format, friendly_name, text_value)

def normalize_attribute_values(values):
    '''Take a list of values or a single one and normalize it'''
    values_set = set()
    if isinstance(values, basestring) or not hasattr(values, '__iter__'):
        values = [values]
    for value in values:
        if isinstance(value, bool):
            value = str(value).lower()
        values_set.add(unicode(value))
    return values_set
```

#10 - 13 avril 2018 10:59 - Benjamin Dauvergne

Pour moi tu es sur la bonne piste:

- en OIDC on appelle les attributs des "claim"s, renomme OIDCAttribut en OIDCClaim pour rester dans le jargon OIDC
- modifie normalize_attribute_values() pour ne pas systématiquement convertir les valeurs uniques en liste (c'est normal en SAML mais en OIDC qui fait la différence entre liste et valeur unique puisque le format de transfert c'est du JSON, ce n'est pas nécessaire), tu peux en faire un flag ou mieux deux fonctions différentes, idem sur la normalisation vers unicode, je ne suis pas sûr que ce soit toujours nécessaire, il vaut peut-être mieux le réserver pour les modèles, en fait il faut peut-être mieux un normalize_attribute_values() spécifique à OIDC
- applique simplement normalize_attribute_values() à tes valeurs avant d'utiliser la valeur dans user_info
- manque la migration de donnée pour créer les OIDCClaim pour la configuration de base
- dans l'admin si on ne définit pas d'attributs (il y a un InlineFormSet, on doit pouvoir détecter qu'il est vide, et/ou l'initialiser dans la vue de création d'un OIDCClient) créer aussi la configuration de base

Points pas importants à ce stade mais je les écris ici, tu pourras en faire des tickets supplémentaires pour plus tard:

- j'aurai plutôt vu l'association scope<->claim sur un objet OIDCScope mais restons comme cela pour l'instant
- d'après la spécification OIDC on peut demander un claim via son scope mais aussi directement par son nom (chaque nom de claim est lui même un scope normalement)

#11 - 17 avril 2018 17:03 - Josué Kouka

- Fichier 0002-idp-oidc-add-extra-attributes-configuration-21870.patch ajouté

- Fichier 0001-make-attribute-engine-properly-return-user-ou-data.patch ajouté

Ok j'ai pris en compte les différents commentaires.

- renommage de OIDCAttribute en OIDCClaim
- pré-définition des différents claim lors de la création (dans l'admin)
- migration des données pour la création des OIDCClaim pour les client déja existant + test.
  Par contre, j'ai un test qui échoue car l'email de l'utilisateur n'est pas vérifié. Je suis en train de chercher le pourquoi.

#12 - 18 avril 2018 10:43 - Josué Kouka

- Fichier 0002-idp-oidc-add-extra-attributes-configuration-21870.patch ajouté

- Fichier 0001-make-attribute-engine-properly-return-user-ou-data.patch ajouté

> Par contre, j'ai un test qui échoue car l'email de l'utilisateur n'est pas vérifié. Je suis en train de chercher le pourquoi.
Dans l'ancienne implémentation, email_verified était forcé à True.

Patch avec tous les tests qui passent.

#13 - 19 avril 2018 09:33 - Josué Kouka

- Fichier 0002-idp-oidc-add-extra-attributes-configuration-21870.patch ajouté

- Fichier 0001-make-attribute-engine-properly-return-user-ou-data.patch ajouté

J'avais intorduit un bug (les valeurs des attributs des OU étaient fausses) dans get_attributes dans mon précédent patch. C'est corrigé dans ceux-ci.

#14 - 20 avril 2018 10:36 - Benjamin Dauvergne

Tu ne devrais pas avoir à poser oidcclient explicitement dans une ModelForm, l'objet en cours est disponible dans self.instance.

L'initialisation de initial peut se faire même en dehors d'un POST et juste avant cette ligne:

```
        formset.__init__ = curry(formset.__init__, initial=initial)
```

Je te propose de définir un get_service_variables(service) à partager avec le code SAML équivalent dans authentic2/saml/admin.py.

Au passage un besoin de dernière minute, ce serait bien de regarder aussi la présence d'une clé claim.value + ':verified' dans attributes et si c'est présent et vrai de poser un claim claim.name + '_verified' dans user_info avec la valeur True. C'est la façon standard en OIDC de passer le statut vérifié d'un attribut.

#15 - 20 avril 2018 18:44 - Josué Kouka

- Fichier 0002-idp-oidc-add-extra-attributes-configuration-21870.patch ajouté

- Fichier 0001-make-attribute-engine-properly-return-user-ou-data.patch ajouté

J'ai pris en compte des remarques, par contre en ce qui concerne

> L'initialisation de initial peut se faire même en dehors d'un POST et juste avant cette ligne

cela ne semble pas fonctionner. D'apres les tests, on a l'impréssion que les attributs sont définis alors qu'en vrai ils ne le sont pas.

#16 - 02 mai 2018 17:32 - Benjamin Dauvergne

Ajoute une commentaire au dessus de

```
# formsets are only saved if formset.has_changed() is True, so only set initial
# values on the GET (display of the creation form)
if request.method == 'GET' and not obj:
```

et ack.

#17 - 02 mai 2018 17:40 - Josué Kouka

- Lié à Bug #23544: Ajouter la possibilité de demander un CLAIM via son nom ajouté

#18 - 02 mai 2018 17:58 - Josué Kouka

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

- % réalisé changé de 0 à 100

```
commit f06900ead4e8ebd1b0b6d759aa0547344d152a64 (HEAD -> master, origin/master, origin/HEAD)
Author: Josue Kouka <jkouka@entrouvert.com>
Date:   Fri Apr 13 09:17:19 2018 +0200

    idp oidc: add extra attributes configuration (#21870)
```

#19 - 04 décembre 2018 20:08 - Frédéric Péters

- Statut changé de Résolu (à déployer) à Fermé

Fichiers

0001-idp-oidc-allow-extension-of-user-info-21870.patch	3,14 ko	09 avril 2018	Josué Kouka
0001-idp-oidc-add-extra-attributes-configuration-21870.patch	9,15 ko	13 avril 2018	Josué Kouka
0002-idp-oidc-add-extra-attributes-configuration-21870.patch	12,8 ko	17 avril 2018	Josué Kouka
0001-make-attribute-engine-properly-return-user-ou-data.patch	2,06 ko	17 avril 2018	Josué Kouka
0001-make-attribute-engine-properly-return-user-ou-data.patch	2,06 ko	18 avril 2018	Josué Kouka
0002-idp-oidc-add-extra-attributes-configuration-21870.patch	13,4 ko	18 avril 2018	Josué Kouka
0002-idp-oidc-add-extra-attributes-configuration-21870.patch	13,3 ko	19 avril 2018	Josué Kouka
0001-make-attribute-engine-properly-return-user-ou-data.patch	2,04 ko	19 avril 2018	Josué Kouka
0002-idp-oidc-add-extra-attributes-configuration-21870.patch	15,9 ko	20 avril 2018	Josué Kouka

0001-make-attribute-engine-properly-return-user-ou-data.patch	2,04 ko	20 avril 2018	Josué Kouka