w.c.s. - Development #14207

Modifier le provisionning pour retrouver les rôles par uuid, slug ou nom

05 décembre 2016 11:53 - Benjamin Dauvergne

Statut:	Fermé	**Début:**	05 décembre 2016
Priorité:	Normal	**Echéance:**	
Assigné à:	Benjamin Dauvergne	**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Oui	**Planning:**	

Description

On actuellement deux cas:

- le rôle a été crée par le provisionning: id = uuid, slug = uuid, name = name
- le rôle existait avant le provisionning: id = <id séquentiel généré par w.c.s.>, slug = uuid, name = name

Le deuxième cas existe principalement à Montpellier (mais aussi toute plateforme qui a migré des plate-formes publik initiales basées sur portail-citoyen).

Je souhaiterai ajouter un champ explicite uuid aux rôle pour y stocker l'uuid, et ensuite affecter réellement le slug venant d'authentic dans le champ slug. Ce nouveau champ devraêtre indexé lui aussi. Le but étant de pouvoir vraiment utiliser le slug pour se référer à un rôle de manière non-équivoque coté w.c.s, lors de migration de plateforme ou d'importation de workflows ou formulaires.

Demandes liées:

Lié à Publik - Development #9934: import/export utilisateurs et rôles d'authe...	**Fermé**	**10 février 2016**
Lié à w.c.s. - Bug #22826: global name 'slug' is not defined	**Fermé**	**27 mars 2018**

Révisions associées

Révision 03123e97 - 27 mars 2018 13:08 - Benjamin Dauvergne

store IdP role's uuid in w.c.s. Role objects (#14207)

- added Role.uuid = None, make it indexed
- modify hobo_notify to store provisionnined roles' uuid in Role.uuid,
and roles' slug in Role.slug
- factorize role lookup in hobo_notify and implement those rules for
lookup:
- Role.uuid matches uuid OR
- Role.id matches uuid (retro-compatibility) OR
- Role.slug matches uuid (retro-compatibility) OR
- Role.slug matches slug (NEW) OR
- Role.name matches name (NEW)
- search users's roles by name too
- update roles provisionning through SAML to use the new .uuid field
- update workflow actions to add/remove roles to use the new .uuid field

Historique

#1 - 05 décembre 2016 11:53 - Benjamin Dauvergne

- Tracker changé de Bug à Development

#2 - 05 décembre 2016 11:59 - Frédéric Péters

Je suis plutôt pour évoluer vers id == uuid partout, plutôt qu'un nouveau champ. (en faisant cette évolution "manuellement" sur les sites où ce ne serait pas encore le cas (i.e. un script qui passe bien partout formdefs/workflows où il y a des références aux rôles et modifie celles-ci)).

#3 - 05 décembre 2016 12:05 - Benjamin Dauvergne

C'est 'achement plus lourd à faire, il faudrait modifier les références partout.

#4 - 08 mars 2017 17:57 - Benjamin Dauvergne

Je remonte ce ticket, problème survenu pour les déploiement automatiques sictiam:
- création d'une instance de w.c.s. à partir d'un export complet (soir un zip du répertoire data) : les rôles sont des pickles, id == role-authentic.uuid, name = role-authentic.name, slug role-authentic.uuid

- création d'un rôle du même nom coté authentic

Souci: contrairement à ce que je pensais le matching des rôles se fait uniquement via l'UUID ou le slug et jamais par nom, ça ne marche que si le slug actuel du rôle w.c.s. correspond au rôle authentic, ça ne marche donc que dans le cas de la migration d'un w.c.s. legacy vers une plate-forme Publik avec IdP.

Actuellement JB a du forcer l'uuid lors de la création du rôle dans authentic pour que ça se recolle.

Alors deux choses:
- on ajoute un match par name en plus de ceux par slug existants, ça résout le problème immédiatement,
- on fait aussi ce qui est décrit plus haut et qui met en ordre le format des données coté w.c.s et a2 tout en continuant à supporter l'existant (i,e role.id != a2_role.uuid)

#5 - 08 mars 2017 18:02 - Frédéric Péters

> - création d'une instance de w.c.s. à partir d'un export complet (soir un zip du répertoire data) : les rôles sont des pickles, id == role-authentic.uuid, name = role-authentic.name, slug role-authentic.uuid

Je dirais qu'il y a erreur là, sur du déploiement où les rôles sont gérés par authentic, il ne faut pas les taper dans l'export wcs.

#6 - 08 mars 2017 18:16 - Benjamin Dauvergne

Ok mais comment recoller les rôles assignés aux formulaires/workflow avec les rôles d'authentic ?

#7 - 08 mars 2017 21:00 - Thomas Noël

Benjamin Dauvergne a écrit :

> Ok mais comment recoller les rôles assignés aux formulaires/workflow avec les rôles d'authentic ?

Comme je l'avais signalé à Jibé, je pense que l'import des workflows puis formulaires depuis des export XML (et non pas des pickles) peut recoller avec des rôles existants, selon leur nom.

J'avais proposé (sans insister parce que c'est un peu relou) à JB le process suivant :

- déploiement "cook" avec wcs vide (ni role ni workflow ni formulaires)
- ajout des x rôles nécessaires via authentic (et waitpoint/check de leur bon provisionning dans wcs) -- avec éventuellement programmation des permissions d'admission wcs
- import de workflows en xml un par un dans wcs (y'a pas de commande pour, mais on peut/devrait la créer)
- import de formdefs en xml un par un dans wcs (y'a pas de commande pour, mais on peut/devrait la créer)

C'est donc un peu relou car il y a pas mal de code à jouer, mais ce sont des outils qui pourraient ensuite être intégrés dans le cook ; et on avance alors à grand pas vers la matrikisation.

#8 - 08 mars 2017 22:04 - Frédéric Péters

- Sujet changé de Modfier le provisionning des rôles pour conserver le slug en tout circonstance à Modifier le provisionning des rôles pour conserver le slug en tout circonstance

> Ok mais comment recoller les rôles assignés aux formulaires/workflow avec les rôles d'authentic ?

J'imaginais aussi, l'export/import des rôles dans Authentic il contient et restaure l'uuid, du coup il n'y a rien à recoller. (et donc, même, par rapport à ce que j'écrivais ils pourraient se trouver dans l'export wcs ça ne changerait rien).

Mais je viens de regarder le code du hobo_deploy et du chargement initial des rôles, et il a du code explicite pour changer les uuid. Et c'est fait ainsi parce que ces rôles sont créés selon les instances d'autres applications (un wcs est déployé avec le template xxx, les rôles du fichier xxx sont créés dans authentic), que ces rôles peuvent être importés plusieurs fois (une fois par collectivité).

(tout ça est quand même un peu éloigné de ce ticket)

#9 - 09 mars 2017 01:23 - Benjamin Dauvergne

J'aimerai expliquer pourquoi je n'aime pas utiliser les UUIDs au niveau des imports/exports. Les imports/exports concernent des plate-formes indépendantes, les rapports entre elles pour moi devrait dépendre de données sémantique un slug ou un nom pour moi ça a un sens, un uuid pas. L'uuid ça sert uniquement à recoller les morceaux dans un système distribué donné étant donné des renommages (et aussi réconcilier des bases diverses en évitant des collisions mais ici ce n'est pas le problème qui nous occupe).

Pour moi que ce soit au niveau de l'import des workflows/formulaires ou de l'import de rôles dans authentic, la clé devrait être le slug et

éventuellement le nom. Si on exporte un Agent SVE d'un site x ou d'un site y ça ne devrait fondamentalement pas faire de différence.

#10 - 20 février 2018 14:19 - Benjamin Dauvergne

- Lié à Development #9934: import/export utilisateurs et rôles d'authentic ajouté

#11 - 20 février 2018 15:19 - Benjamin Dauvergne

- Fichier 0001-store-IdP-role-s-uuid-in-w.c.s.-Role-objects-14207.patch ajouté

À tester beaucoup.

#12 - 20 février 2018 15:21 - Benjamin Dauvergne

- Sujet changé de Modifier le provisionning des rôles pour conserver le slug en tout circonstance à Modifier le provisionning pour retrouver les rôles par uuid, slug ou nom

#13 - 20 février 2018 15:31 - Benjamin Dauvergne

- Fichier 0003-modifie-le-provisionning-des-r-les-via-le-SSO-rebase.patch ajouté

- Fichier 0001-store-IdP-role-s-uuid-in-w.c.s.-Role-objects-14207.patch ajouté

- Fichier 0004-modifie-les-actions-de-workflows-d-ajout-r-traction-.patch ajouté

- Fichier 0002-recherche-les-r-les-des-utilisateurs-aussi-par-nom-r.patch ajouté

- recherche aussi les rôles des utilisateurs par leur nom (si un utilisateur arriver avant que l'UUID ait été corrigé)
- réutilise CmdHoboNotify.find_role() lors du SSO pour trouver les rôles des utilisateurs
- modifie l'action de workflow Add/RemoveRole pour utiliser role.uuid ou à défaut role.slug

#14 - 20 février 2018 15:31 - Benjamin Dauvergne

- Patch proposed changé de Non à Oui

#15 - 20 février 2018 16:03 - Thomas Noël

Au moins déplacer find_role dans la classe Role, pour ne pas se retrouver avec des import hobo_notify dans saml2.py

Ensuite j'ai à cet instant du mal à mesurer l'impact de l'ajout de Role.uuid, notamment au niveau des import depuis de l'XML, qui ont aussi leur logique de recherche d'un rôle existant. Sans doute faudrait-il chercher convergence ici.

#16 - 20 février 2018 16:32 - Benjamin Dauvergne

Au niveau des imports XML concernant les rôles je trouve ça (j'ai pas regardé coté status item mais je crois que c'est le même code):

```
wcs/formdef.py-         roles_elements = [
wcs/formdef.py-             ('roles', 'user-roles'),
wcs/formdef.py-             ('backoffice_submission_roles', 'backoffice-submission-roles')
wcs/formdef.py-         ]
wcs/formdef.py-         for attr_name, node_name in roles_elements:
wcs/formdef.py-             if not getattr(self, attr_name, None):
wcs/formdef.py-                 continue
wcs/formdef.py-             roles = ET.SubElement(root, node_name)
wcs/formdef.py-             for role_id in getattr(self, attr_name):
wcs/formdef.py-                 if role_id is None:
wcs/formdef.py-                     continue
wcs/formdef.py-                 role_id = str(role_id)
wcs/formdef.py-                 if role_id.startswith('_') or role_id == 'logged-users':
wcs/formdef.py-                     role = unicode(role_id, charset)
wcs/formdef.py-                 else:
wcs/formdef.py-                     try:
wcs/formdef.py:                         role = unicode(Role.get(role_id).name, charset)
wcs/formdef.py-                     except KeyError:
wcs/formdef.py-                         role = unicode(role_id, charset)
wcs/formdef.py-                 sub = ET.SubElement(roles, 'role')
wcs/formdef.py-                 if include_id:
wcs/formdef.py-                     sub.attrib['role_id'] = role_id
wcs/formdef.py-                 sub.text = role
wcs/formdef.py-
wcs/formdef.py-         if self.workflow_roles:
wcs/formdef.py-             roles = ET.SubElement(root, 'roles')
wcs/formdef.py-             for role_key, role_id in self.workflow_roles.items():
```

```
wcs/formdef.py-                if role_id is None:
wcs/formdef.py-                    continue
wcs/formdef.py-                role_id = str(role_id)
wcs/formdef.py-                if role_id.startswith('_') or role_id == 'logged-users':
wcs/formdef.py-                    role = unicode(role_id, charset)
wcs/formdef.py-                else:
wcs/formdef.py-                    try:
wcs/formdef.py:                        role = unicode(Role.get(role_id).name, charset)
wcs/formdef.py-                    except KeyError:
wcs/formdef.py-                        role = unicode(role_id, charset)
wcs/formdef.py-                sub = ET.SubElement(roles, 'role')
wcs/formdef.py-                sub.attrib['role_key'] = role_key
wcs/formdef.py-                if include_id:
wcs/formdef.py-                    sub.attrib['role_id'] = role_id
wcs/formdef.py-                sub.text = role
wcs/formdef.py-        roles_elements = [
wcs/formdef.py-            ('roles', 'user-roles'),
wcs/formdef.py-            ('backoffice_submission_roles', 'backoffice-submission-roles')
wcs/formdef.py-        ]
wcs/formdef.py-        for attr_name, node_name in roles_elements:
wcs/formdef.py-            if tree.find(node_name) is None:
wcs/formdef.py-                continue
wcs/formdef.py-            roles_node = tree.find(node_name)
wcs/formdef.py-            roles = []
wcs/formdef.py-            setattr(formdef, attr_name, roles)
wcs/formdef.py-            for child in roles_node.getchildren():
wcs/formdef.py-                role_id = None
wcs/formdef.py-                value = child.text.encode(charset)
wcs/formdef.py-                if value.startswith('_') or value == 'logged-users':
wcs/formdef.py-                    role_id = value
wcs/formdef.py-                elif include_id:
wcs/formdef.py-                    role_id = child.attrib.get('role_id')
wcs/formdef.py:                    if role_id and not Role.has_key(role_id):
wcs/formdef.py-                        role_id = None
wcs/formdef.py-
wcs/formdef.py-                if not role_id:
wcs/formdef.py:                    for role in Role.select(ignore_errors=True):
wcs/formdef.py-                        if role.name == value:
wcs/formdef.py-                            role_id = role.id
wcs/formdef.py-                            break
wcs/formdef.py-                if role_id:
wcs/formdef.py-                    roles.append(role_id)
wcs/formdef.py-
wcs/formdef.py-        if tree.find('roles') is not None:
wcs/formdef.py-            roles_node = tree.find('roles')
wcs/formdef.py-            formdef.workflow_roles = {}
wcs/formdef.py-            for child in roles_node.getchildren():
wcs/formdef.py-                role_key = child.attrib['role_key']
wcs/formdef.py-                role_id = None
wcs/formdef.py-                value = child.text.encode(charset)
wcs/formdef.py-                if value.startswith('_') or value == 'logged-users':
wcs/formdef.py-                    role_id = value
wcs/formdef.py-                elif include_id:
wcs/formdef.py-                    role_id = child.attrib.get('role_id')
wcs/formdef.py-                else:
wcs/formdef.py:                    for role in Role.select(ignore_errors=True):
wcs/formdef.py-                        if role.name == value:
wcs/formdef.py-                            role_id = role.id
wcs/formdef.py-                            break
wcs/formdef.py-
wcs/formdef.py:                if role_id and not Role.has_key(role_id):
wcs/formdef.py-                    role_id = None
wcs/formdef.py-
wcs/formdef.py-                formdef.workflow_roles[role_key] = role_id
wcs/formdef.py-
wcs/workflows.py-    def _roles_export_to_xml(self, attribute, item, charset, include_id=False):
wcs/workflows.py-        if not hasattr(self, attribute) or not getattr(self, attribute):
wcs/workflows.py-            return
wcs/workflows.py-        el = ET.SubElement(item, attribute)
wcs/workflows.py-        for role_id in getattr(self, attribute):
wcs/workflows.py-            if role_id is None:
wcs/workflows.py-                continue
wcs/workflows.py-            role_id = str(role_id)
wcs/workflows.py-            if role_id.startswith('_') or role_id == 'logged-users':
wcs/workflows.py-                role = unicode(role_id, charset)
```

```
wcs/workflows.py-            else:
wcs/workflows.py-                try:
wcs/workflows.py:                    role = unicode(Role.get(role_id).name, charset)
wcs/workflows.py-                except KeyError:
wcs/workflows.py-                    role = unicode(role_id, charset)
wcs/workflows.py-            sub = ET.SubElement(el, 'item')
wcs/workflows.py-            sub.attrib['role_id'] = role_id
wcs/workflows.py-            sub.text = role
wcs/workflows.py-    def _get_role_id_from_xml(self, elem, charset, include_id=False):
wcs/workflows.py-        if elem is None:
wcs/workflows.py-            return None
wcs/workflows.py-
wcs/workflows.py-        value = elem.text.encode(charset)
wcs/workflows.py-
wcs/workflows.py-        # look for known static values
wcs/workflows.py-        if value.startswith('_') or value == 'logged-users':
wcs/workflows.py-            return value
wcs/workflows.py-
wcs/workflows.py-        # if we import using id, only look at the role_id attribute
wcs/workflows.py-        if include_id:
wcs/workflows.py-            if not 'role_id' in elem.attrib:
wcs/workflows.py-                return None
wcs/workflows.py-            role_id = str(elem.attrib['role_id'])
wcs/workflows.py:            if Role.has_key(role_id):
wcs/workflows.py-                return role_id
wcs/workflows.py-            else:
wcs/workflows.py-                return None
wcs/workflows.py-
wcs/workflows.py-        # if not using id, look up on the name
wcs/workflows.py:        for role in Role.select(ignore_errors=True):
wcs/workflows.py-            if role.name == value:
wcs/workflows.py-                return role.id
wcs/workflows.py-
wcs/workflows.py-        # if a computed value is possible and value looks like
wcs/workflows.py-        # an expression, use it
wcs/workflows.py-        if value.startswith('=') or Template.is_template_string(value):
wcs/workflows.py-            return value
wcs/workflows.py-
wcs/workflows.py-        # if the roles are managed by the idp, don't try further.
wcs/workflows.py-        if get_publisher() and get_cfg('sp', {}).get('idp-manage-roles') is True:
wcs/workflows.py-            raise WorkflowImportError(N_('Unknown referenced role (%s)'), (value,))
wcs/workflows.py-
wcs/workflows.py-        # and if there's no match, create a new role
wcs/workflows.py-        role = Role()
wcs/workflows.py-        role.name = value
wcs/workflows.py-        role.store()
wcs/workflows.py-        return role.id
```

c'est basé uniquement sur l'id ou le nom, à l'occasion je remarque que l'import de workflow bloque un rôle inconnu quand c'est managé par l'IdP mais pas l'import des formdef, ça vient du #13933.

Et donc à la lecture de ce #13933 je me rends compte qu'on est toujours sur deux positions différentes toi et Fred et moi, pous vous l'uuid est partout et sert de référence, d'où le fait que ce soit grave de créer un rôle automatiquement alors que pour moi on s'en fout, c'est la procédure de provisionning qui était foireuse (et plutôt que le #13933 c'est ce ticket ici qu'il aurait fallu traiter).

Donc déjà il faudrait le même comportement sur les formdef que sur les workflows, qu'elle qu'il soit. Ensuite si on commit ce ticket là alors je pense qu'on peut défaire le #13933 et ajouter la création automatique des rôles à l'import des formdef (comme pour les workflows), mais c'est discutable.

#17 - 16 mars 2018 01:25 - Benjamin Dauvergne

Up, nécessaire pour l'import des rôles.

#18 - 16 mars 2018 07:57 - Frédéric Péters

Je n'ai toujours pas changé de position, j'aurais vraiment préféré id=uuid plutôt que poser ça dans un nouvel attribut. Mais ok j'oublie.

D'un coup d'œil :

- l'intitulé des patchs est en français.
- vu qu'il se trouve utilisé ailleurs le find_role ne devrait pas être dans une commande de management, devrait être en classmethod de Role.
- ça reste à tester un certain temps.

#19 - 16 mars 2018 14:20 - Benjamin Dauvergne

Frédéric Péters a écrit :

Je n'ai toujours pas changé de position, j'aurais vraiment préféré id=uuid plutôt que poser ça dans un nouvel attribut. Mais ok j'oublie.

Rétro-compatibilité impose de mettre uuid aussi ailleurs que dans id (montpellier), cohérence impose de ne plus le mettre dans slug, mais j'ai remis Role.id == uuid en plus. J'ai en tête aussi le fait de ne plus bloquer l'import d'un workflow ou formulaire sur absence d'un rôle mais de le créer via son nom, cela impose de pouvoir remettre l'uuid ensuite ailleurs, mais j'entends bien la facilité de debug pour toi de pouvoir voir directement via l'id en conditions normales qu'un rôle est là ou pas.

D'un coup d'œil :

- l'intitulé des patchs est en français.

J'ai rebasé et j'ai intégré des commentaires en anglais.

- vu qu'il se trouve utilisé ailleurs le find_role ne devrait pas être dans une commande de management, devrait être en classmethod de Role.

Ok.

- ça reste à tester un certain temps.

D'ici que que l'import des rôles soit poussé sûr, si il y a des idées de tests obscures qui ne seraient pas déjà fait je veux bien en ajouter.

#20 - 16 mars 2018 14:20 - Benjamin Dauvergne

- Fichier 0001-store-IdP-role-s-uuid-in-w.c.s.-Role-objects-14207.patch ajouté

Rebasage et prise en compte des remarques de Fred.

#21 - 16 mars 2018 14:20 - Benjamin Dauvergne

- Assigné à mis à Benjamin Dauvergne

#22 - 26 mars 2018 17:28 - Frédéric Péters

Je l'ai pris en local avec l'idée de l'envoyer mais tous les tests échouent avec :

```
    def register_ident_methods(self):
        try:
            import lasso
        except ImportError:
            lasso = None
        classes = []
        if lasso:
            import qommon.ident.idp
            classes.append(qommon.ident.idp.IdPAuthMethod)
        import qommon.ident.franceconnect
>       classes.append(qommon.ident.franceconnect.FCAuthMethod)
E       AttributeError: 'module' object has no attribute 'franceconnect'

wcs/qommon/publisher.py:635: AttributeError
```

#23 - 26 mars 2018 17:43 - Frédéric Péters

(poussé dans une branche pour l'avoir dans jenkins : https://jenkins.entrouvert.org/job/wcs-wip-branches/142/console)

#24 - 26 mars 2018 21:07 - Benjamin Dauvergne

- Fichier 0001-store-IdP-role-s-uuid-in-w.c.s.-Role-objects-14207.patch ajouté

Import devenu inutile de CmdHoboNotify retiré.

#25 - 26 mars 2018 22:12 - Frédéric Péters

```
>               role = Role(id=uuid)
E               TypeError: __init__() got an unexpected keyword argument 'id'
```

— https://jenkins.entrouvert.org/job/wcs-wip-branches/143/console

#26 - 26 mars 2018 23:22 - Benjamin Dauvergne

La branche est à jour avec des tests qui passent.

#27 - 27 mars 2018 13:28 - Frédéric Péters

- Statut changé de Nouveau à Résolu (à déployer)

```
commit 03123e9742ce1b4b292a967f22609998effd0660
Author: Benjamin Dauvergne <bdauvergne@entrouvert.com>
Date:   Tue Feb 20 15:14:15 2018 +0100

    store IdP role's uuid in w.c.s. Role objects (#14207)
(...)
```

Laborieux. À peine testé en vrai en local, voyons rapidement comment ça tourne sur la recette.

#28 - 27 mars 2018 14:48 - Frédéric Péters

- Lié à Bug #22826: global name 'slug' is not defined ajouté

#29 - 23 décembre 2018 14:52 - Frédéric Péters

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

Fichiers

0001-store-IdP-role-s-uuid-in-w.c.s.-Role-objects-14207.patch	9,59 ko	20 février 2018	Benjamin Dauvergne
0003-modifie-le-provisionning-des-r-les-via-le-SSO-rebase.patch	1,19 ko	20 février 2018	Benjamin Dauvergne
0001-store-IdP-role-s-uuid-in-w.c.s.-Role-objects-14207.patch	9,59 ko	20 février 2018	Benjamin Dauvergne
0004-modifie-les-actions-de-workflows-d-ajout-r-traction-.patch	1,29 ko	20 février 2018	Benjamin Dauvergne
0002-recherche-les-r-les-des-utilisateurs-aussi-par-nom-r.patch	1,47 ko	20 février 2018	Benjamin Dauvergne
0001-store-IdP-role-s-uuid-in-w.c.s.-Role-objects-14207.patch	12 ko	16 mars 2018	Benjamin Dauvergne
0001-store-IdP-role-s-uuid-in-w.c.s.-Role-objects-14207.patch	11,8 ko	26 mars 2018	Benjamin Dauvergne