Hobo - Development #53059

ajouter une API de provisionning synchrone à authentic

13 avril 2021 14:16 - Frédéric Péters

Statut:	Fermé	**Début:**	13 avril 2021
Priorité:	Normal	**Echéance:**	
Assigné à:	Frédéric Péters	**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Oui	**Planning:**	Non

Description

Il faudrait ajouter une API pour demander un provisionning synchro explicite, qui rendrait uniquement la main une fois le provisionning terminé. (ça suppose/forcerait un provisionning HTTP); ce serait surtout pour les utilisateurs (parce qu'ils se trouvent parfois créés via des appels depuis wcs et puis pas encore là) mais ça pourrait aussi être utile pour les rôles (depuis un workflow wcs on a une action explicite, qui gère ça, mais dans le code de déploiement global publik-imio-industrialisation je me suis trouvé ajouter une boucle vérifiant que le rôle était arrivé sur chacun des modules).

hobo/agent/authentic2/ est chargé dans INSTALLED_APPS d'authentic, il y a moyen de déclarer des URL par là.

Demandes liées:

Lié à Hobo - Development #56920: ajouter un paramètre pour garantir la synchr...	**Fermé**	**14 septembre 2021**

Révisions associées

Révision 13c83ff3 - 15 avril 2021 12:48 - Frédéric Péters

authentic: add API to force user provisionning (#53059)

Historique

#2 - 14 avril 2021 09:59 - Frédéric Péters

- Statut changé de Nouveau à En cours

- Assigné à mis à Frédéric Péters

#3 - 14 avril 2021 10:42 - Frédéric Péters

- Fichier 0001-authentic-add-API-to-force-user-provisionning-53059.patch ajouté

- Statut changé de En cours à Solution proposée

- Patch proposed changé de Non à Oui

Voilà, /api/provision/ (je me suis demandé si on voulait /api/ réservé aux API natives authentic mais j'ai vu qu'authentic2-cut installait aussi une API sous /api/, donc voilà); ça utilise djangorestframework comme les autres API d'authentic, ça peut recevoir en entrée user_uuid, role_uuid, service_type et service_url.

- user_uuid et role_uuid sont exclusifs, correspondent à la demande de provisionning de tel utilisateur ou tel rôle,
- service_type permet de limiter le provisionning aux services du type donné (ex: wcs, pour ne pas bloquer sur les appels à tous les autres modules),
- service_url permet de manière différente de limiter le provisionning, cette fois sur base de l'URL du service (à noter ici que c'est une comparaison "in" qui est réalisée, c'était au début un raccourci moche pour permettre à l'appelant d'oublier le slash final mais ça pourrait aussi être utile en multicollectivités, avec le nom de la collectivité dans l'url, oui ça fait substitut moche à l'utilisation d'une OU) (si ce paramètre est questionné, je peux totaltement le retirer).

Ça retourne dans un attribut "leftover_audience" les endroits où le provisionning a échoué, je me suis demandé si on voulait mettre 'err': 1 au cas où il y a quelque chose dedans, je ne l'ai pas fait.

#4 - 14 avril 2021 11:30 - Thomas Noël

Frédéric Péters a écrit :

> Ça retourne dans un attribut "leftover_audience" les endroits où le provisionning a échoué, je me suis demandé si on voulait mettre 'err': 1 au cas où il y a quelque chose dedans, je ne l'ai pas fait.

Tu as une raison pour ça ? Pour ma part je préfère un err:1 au moindre pépin, histoire qu'on ne se retrouve pas avec des conditions bizarres en Django dans les workflows pour gérer les pépins.

Au sujet des pépins potentiels je vois les timeout : faire en sorte que authentic réponde dans un délai maxi de 20 secondes est nécessaire à mon avis. J'imagine bêtement un truc genre :

```
  ...
  total_timeout = 20  # pourrait venir dans la signature de la fonction, genre total_timeout=120 par défaut en
 mode de notif classique
  for audience in rest_audience:
      start_time = time.time()
      try:
          requests.put(..., timeout=min(total_timeout, 5))
      except requests.RequestException as e:
          logger.error(u'error provisionning to %s (%s)', audience, e)
      else:
          leftover_audience.remove(audience)
      total_timeout = total_timeout - (time.time() - start_time)
      if total_timeout <= 0:
          logger.error(u'global notify_agents_http timeout'
          break
    ....
```

#5 - 14 avril 2021 11:45 - Frédéric Péters

> Tu as une raison pour ça ?

La même que pour le fallback amqp qui existe pour le provisionning HTTP, parfois un service est down, genre redémarrage, mais oui pour mettre err: 1.

> Au sujet des pépins potentiels je vois les timeout : faire en sorte que authentic réponde dans un délai maxi de 20 secondes est nécessaire à mon avis. J'imagine bêtement un truc genre :

S'il y a timeout l'appelant verra bien qu'il y a un soucis à l'appel; je n'ajouterais rien de particulier ici.

#6 - 14 avril 2021 13:50 - Thomas Noël

Frédéric Péters a écrit :

> oui pour mettre err: 1.

Impec.

> > Au sujet des pépins potentiels je vois les timeout : faire en sorte que authentic réponde dans un délai maxi de 20 secondes est nécessaire à mon avis. J'imagine bêtement un truc genre :
>
> S'il y a timeout l'appelant verra bien qu'il y a un soucis à l'appel; je n'ajouterais rien de particulier ici.

Ça me va, je ne sais pas pourquoi je cherche toujours à me compliquer la vie.

#7 - 14 avril 2021 15:54 - Frédéric Péters

- Fichier 0001-authentic-add-API-to-force-user-provisionning-53059.patch ajouté

Voilà err: 1 quand il y a au moins un échec.

#8 - 14 avril 2021 20:46 - Benjamin Dauvergne

Si jamais ça a une importance tout me va. Mais ça aurait pu passer peut-être par un paramétrage des APIs :

- ajouter ?sync lors d'un POST sur l'API des rôles
- détecter ça dans le provisionning 'sync' in StoreRequestMiddleware.get_request().GET
- activer alors un provisioning synchrone en priorité pour les services combo (si la requête vient de w.c.s. c'est via une action ajouter un rôle, et elle est faite d'abord localement) et ensuite les autres services de façon classique

Là il me semble que ça va demander pas mal de paramétrage dans w.c.s. pour être utilisé alors que l'ajout de sync pourrait être une simple case à cocher dans les actions ajouter/retirer un rôle.

Est-ce qu'il a été identifié un besoin de provisionning synchrone pour autre chose que combo ?

#9 - 14 avril 2021 21:34 - Frédéric Péters

C'est écrit dans la description du ticket le problème immédiat c'est le provisionning des utilisateurs, pas des rôles; il y a un lien vers l'origine dans mon premier commentaire mais c'est en fait assez courant qu'authentic soit contourné. Sur l'option d'ajouter un paramètre inconnu d'authentic sur une de ses API, ça ne m'a pas effleuré l'esprit mais je n'y serais pas allé par peur de venir avec des contraintes extérieures sur ces API (déjà placer cette API sous /api/ j'ai hésité…).

#10 - 14 avril 2021 21:45 - Benjamin Dauvergne

Je sais que l'air du temps est à multiplier les appels de web-service plutôt que les actions de workflow; mais alors dans ce cas précis de lier une objet de w.c.s. à un utilisateur dont w.c.s. vient de demander la création, je ferai comme pour les actions ajouter/retirer rôle, je ferai un minimum de boulot localement dans une action "custom" de création d'utilisateur. Comme on connaît l'uuid et 2/3 attributs on pourrait créér l'utilisateur immédiatement au retour de l'appel à authentic. Il faudrait revoir le code de provisionning qui n'est pas top sur les création concurrentes.

Je ne vois un intérêt à la complication de ce genre de billard à 3 bandes que quand il y a plus de 2 briques en jeu. Mais je valide le patch de mon coté et je laisse le dernier mot à Thomas.

#11 - 14 avril 2021 21:49 - Frédéric Péters

cf le commentaire derrière le lien qui reprend cette idée et dit pourquoi non : "alternativement dans w.c.s. il y aurait une action native de création d'usager, comme on a l'ajout/retrait de rôle à un usager, (mais on dira que c'est encombrer l'interface d'une action bien rarement utile)".

#12 - 15 avril 2021 01:00 - Thomas Noël

C'est vrai qu'on est à 3 ou 4 bandes mais j'aime relativement bien l'idée que pour la gestion des utilisateurs, c'est Authentic qui pilote et donc on doit lui dire quoi faire (y compris s'il faut créer un utilisateur dans wcs alors qu'on est dans wcs...). Finalement c'est du billard, c'est quand même mieux que du cross-country (je créé l'utilisateur dans authentic et je récupère l'uuid et je crée un utilisateur localement en lui ajoutant un identifiant saml pour faire croire qu'il vient d'authentic).

Idée de ce soir : pour aider au support et à la compréhension des résultats, je pense que dans le retour du webservice on devrait aussi envoyer l'audience qui a réussi à être jointe, dans une clé "audience". Je sais que ça aidera/rassurera beaucoup de mes collègues.

#13 - 15 avril 2021 03:47 - Benjamin Dauvergne

Ou alors prenons le problème à l'envers, quand on affecte un uuid inconnu à une fiche, w.c.s. pourrait simplement requêter /api/users/<uuid>/ voir si l'utilisateur n'existerait pas et le provisionnerait à la volée. Un appel de WS en moins.

#14 - 15 avril 2021 09:13 - Frédéric Péters

- Fichier 0001-authentic-add-API-to-force-user-provisionning-53059.patch ajouté

Patch actualisé pour reprendre une clé "reached_audience".

#15 - 15 avril 2021 12:25 - Thomas Noël

- Statut changé de Solution proposée à Solution validée

Ca me va bien comme ça.

#16 - 15 avril 2021 12:48 - Frédéric Péters

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

```
commit 13c83ff3cec916ac67b9e41cb810598c79594cc5
Author: Frédéric Péters <fpeters@entrouvert.com>
Date:   Wed Apr 14 00:13:15 2021 +0200

    authentic: add API to force user provisionning (#53059)
```

#17 - 15 avril 2021 20:16 - Frédéric Péters

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

#18 - 14 septembre 2021 07:02 - Benjamin Dauvergne

- Lié à Development #56920: ajouter un paramètre pour garantir la synchronicité dans /api/provision ajouté

Fichiers

0001-authentic-add-API-to-force-user-provisionning-53059.patch	15,3 ko	14 avril 2021	Frédéric Péters
0001-authentic-add-API-to-force-user-provisionning-53059.patch	15,9 ko	14 avril 2021	Frédéric Péters
0001-authentic-add-API-to-force-user-provisionning-53059.patch	16,4 ko	15 avril 2021	Frédéric Péters