Hobo - Development #55043

Quelques optimisations sur le provisionning des roles

21 juin 2021 17:13 - Emmanuel Cazenave

Statut:	Fermé	**Début:**	21 juin 2021
Priorité:	Normal	**Echéance:**	
Assigné à:	Emmanuel Cazenave	**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Oui	**Planning:**	Non

Description

Suite à des tests sur un déploiement à 17 000 rôles.

Révisions associées

Révision 438b79dc - 28 septembre 2021 15:15 - Emmanuel Cazenave

provisionning: count db queries (#55043)

Révision 32ce0768 - 28 septembre 2021 15:15 - Emmanuel Cazenave

provisionning: grab existing roles by uuid in one query (#55043)

Révision 5dc88fc9 - 28 septembre 2021 15:15 - Emmanuel Cazenave

provisionning: delete superfluous roles in one query (#55043)

Révision 6e896dbc - 28 septembre 2021 15:19 - Emmanuel Cazenave

provisionning: delete roles in one query when deprovisionning (#55043)

Historique

#1 - 21 juin 2021 18:52 - Emmanuel Cazenave

- Fichier 0001-provisionning-count-db-queries-55043.patch ajouté

- Fichier 0002-provisionning-grab-existing-roles-by-uuid-in-one-que.patch ajouté

- Fichier 0003-provisionning-delete-superfluous-roles-in-one-query-.patch ajouté

- Fichier 0004-provisionning-delete-roles-in-one-query-when-deprovi.patch ajouté

- Statut changé de Nouveau à Solution proposée

- Patch proposed changé de Non à Oui

Sans surprise le temps est passé essentiellement sur des requêtes SQL, pour optimiser en réduire le nombre.

Sur un message de provisionning full à 17 00 rôles les options ci-jointes font gagner une cinquantaine de secondes (de 170 à 120 secondes).

Ça reste donc encore bien long, pour poursuivre il faudrait s'attaquer à Role.objects.get_or_create qui est appelé rôle par rôle mais c'est plus compliqué parce qu'on ne peut pas utiliser le bulk_create de django qui ne fonctionne pas si le modèle repose sur plusieurs tables, ce qui est le cas ici (auth_group et common_role).

Du coup je ne pousse pas plus loin sur ce ticket, pas trop d'idées.

#2 - 21 juin 2021 18:57 - Frédéric Péters

Ok mais 120 secondes ça va en pratique continuer à péter, ce qu'il faut à mon sens c'est passer le traitement en asynchrone (i.e. on reçoit le message, on le met de côté, on répond ok à authentic, on traite le message); on a des spooler uwsgi presque partout désormais pour gérer ça.

#3 - 21 juin 2021 23:09 - Thomas Noël

Frédéric Péters a écrit :

> Ok mais 120 secondes ça va en pratique continuer à péter, ce qu'il faut à mon sens c'est passer le traitement en asynchrone (i.e. on reçoit le message, on le met de côté, on répond ok à authentic, on traite le message); on a des spooler uwsgi presque partout désormais pour gérer ça.

Sur cette affaire : ne risque-t-on pas de se manger des conflits ? Il me semblait (mais je peux me tromper) que le passage par RabbitMQ assurait que les messages étaient traités un par un, et dans l'ordre d'arrivée. Avec un système asynchrone on va perdre ça, parce que les spoolers sont multiples, et que je suis pas sûr qu'ils gèrent un ordre. Et donc un message n°1 qui ajoute un rôle pourrait être joué après le n°2 qui le supprime, et au final le rôle sera toujours là.

Mais peut-être que je me trompe par rapport à RabbitMQ et dans ce cas, oui, l'asynchrone ça serait chouette.

#4 - 21 juin 2021 23:17 - Emmanuel Cazenave

A vrai dire j'avais pensé au spooler uwsgi je ne m'y suis pas lancé parce qu'incapable de répondre à la question "qu'est ce qu'on gagne par rapport à rabbitmq en passant par uwsgi ?".

C'est une vraie question que je pose, pas une fin de non recevoir sur l'approche.

#5 - 21 juin 2021 23:45 - Thomas Noël

Emmanuel Cazenave a écrit :

> A vrai dire j'avais pensé au spooler uwsgi je ne m'y suis pas lancé parce qu'incapable de répondre à la question "qu'est ce qu'on gagne par rapport à rabbitmq en passant par uwsgi ?".
> C'est une vraie question que je pose, pas une fin de non recevoir sur l'approche.

Je dirais, dans un avenir plus ou moins proche, ne plus utiliser RabbitMQ ? C'est encore illusoire mais bon, je sais que t'aime aussi les compètes d'endurance.

En attendant, passer par HTTP simplifie quand même beaucoup le débogue voir la supervision (mesure des temps de réponse). Dans authentic on pourra recevoir le nom du job créé et aller le "voir" tourner sur le logiciel cible.

Note que pour moi on ne ferait pas systématiquement de l'async. Imaginer que l'async ne se déclenche que si le volume reçu dans le notify est jugé important, avec un settings.HOBO_AGENT_NOTIFY_ASYNC_THRESHOLD=<nombre d'actions à partir duquel on répond en async>.

#6 - 22 juin 2021 08:08 - Frédéric Péters

> > "qu'est ce qu'on gagne par rapport à rabbitmq en passant par uwsgi ?".
>
> ne plus utiliser RabbitMQ

Oui c'est totalement l'objectif; pour reprendre des points fournis dans #43245 qui a introduit le provisionning HTTP : c'est trop compliqué de débugguer les problèmes RabbitMQ, on n'a aucun outil d'analyse; comme on tourne rabbitmq avec un unique worker ça fait un goulot d'étranglement global, partagé par tous les tenants. Aussi, ça assure (dans le cas commun d'ajout d'un utilisateur, ou d'un rôle) un provisionning synchrone, ce qui est d'une grande aide dans les workflows qui font ça.

~~

> Sur cette affaire : ne risque-t-on pas de se manger des conflits ?

Je ne suis pas si sûr de ce que rabbitmq assurait réellement. (et vu le nombre de fois où il a fallu retirer/remettre un rôle pour voir le provisionning se faire, parce que message expiré/perdu, l'éventuel conflit devait passer inaperçu).

Si on en arrive aux conflits, je ne me suis pas remis de vrais messages de provisionning sous les yeux mais je pense qu'on arriverait à les faire évoluer pour inclure des timestamps pour éviter ça.

> Imaginer que l'async ne se déclenche que si le volume reçu dans le notify est jugé important, avec un settings.HOBO_AGENT_NOTIFY_ASYNC_THRESHOLD=<nombre d'actions à partir duquel on répond en async>.

Volume important ou message précédemment passé en asynchrone et pas encore traité.

#7 - 22 juin 2021 10:15 - Benjamin Dauvergne

Emmanuel Cazenave a écrit :

> C'est une vraie question que je pose, pas une fin de non recevoir sur l'approche.

Ça n'a pas de rapport; mais oui l'implémentation d'une queue est un peu lourde avec RabbitMQ et trop simple dans certains cas dans uwsgi mais RabbitMQ n'est pas un gestionnaire de jobs.

Ce que je pense c'est qu'on doit passer d'un modèle push à un modèle pull, et là pas de souci pour jeter ça dans un spooler. Mais il faudrait pouvoir faire des mise à jour incrémentales et pour ça il nous manque un timestamp sur le modèle "through" de Role.members pour la création et un soft

delete. Une fois qu'on a ça on peut avoir un web-service qui nous sert les rôles récents ainsi que les changements d'appartenance depuis une date donnée et authentic n'aura plus qu'à envoyer un message en HTTP "un truc à changer sur les rôles, mets toi à jour" et coté appli on peut faire des mises à jour régulière au cas où un message se soit perdu à un moment.

À noter qu'on pourra faire pareil pour les utilisateurs (il manque sur les utilisateurs un changement de timestamp quand on l'ajoute/retire d'un rôle aussi, ça nous le donnera).

PS: en donc oui celery et rabbitmq sont bien trop compliqués pour nos usages.

#8 - 22 juin 2021 10:19 - Paul Marillonnet

Au passage j'ai l'impression que dans 0002, quitte à optimiser on doit pouvoir taper au moins une compréhension de dictionnaire, genre à la place de :

```
            for role in Role.objects.filter(uuid__in=target_uuids):
                roles_by_uuid[role.uuid] = role
```

lirc même en python>=3.7 ça reste mieux.

#9 - 22 juin 2021 17:16 - Emmanuel Cazenave

- Fichier 0001-provisionning-count-db-queries-55043.patch ajouté

- Fichier 0002-provisionning-grab-existing-roles-by-uuid-in-one-que.patch ajouté

- Fichier 0003-provisionning-delete-superfluous-roles-in-one-query-.patch ajouté

- Fichier 0004-provisionning-delete-roles-in-one-query-when-deprovi.patch ajouté

Pour le spooler uwsgi il y a maintenant #55092.

Asynchrone ou pas ça reste intéressant de faire ça plus vite et de pas taper dans la DB comme des bourrins. Nouvelle version, j'avais fait une petite coquille dans le découpage des patchs.

Remarque de Paul non pris en compte, ce serait comme nettoyer la coque d'un bateau qui a plus de voiles.

#10 - 19 août 2021 11:12 - Benjamin Dauvergne

- Statut changé de Solution proposée à Solution validée

#11 - 28 septembre 2021 15:29 - Emmanuel Cazenave

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

```
commit 6e896dbcdd6f2efa4a3f26dc2237b3bb75437b7d
Author: Emmanuel Cazenave <ecazenave@entrouvert.com>
Date:   Mon Jun 21 17:57:36 2021 +0200

    provisionning: delete roles in one query when deprovisionning (#55043)

commit 5dc88fc9a71203bf9f59055c0a96143e01741ce9
Author: Emmanuel Cazenave <ecazenave@entrouvert.com>
Date:   Mon Jun 21 17:55:03 2021 +0200

    provisionning: delete superfluous roles in one query (#55043)

commit 32ce07683438d1ca966d8f3cd1de9a9461e755aa
Author: Emmanuel Cazenave <ecazenave@entrouvert.com>
Date:   Mon Jun 21 17:49:28 2021 +0200

    provisionning: grab existing roles by uuid in one query (#55043)

commit 438b79dc7e136d7412c146dae815938de3af7807
Author: Emmanuel Cazenave <ecazenave@entrouvert.com>
Date:   Mon Jun 21 17:16:28 2021 +0200

    provisionning: count db queries (#55043)
```

#12 - 05 octobre 2021 10:17 - Frédéric Péters

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

Fichiers

0001-provisionning-count-db-queries-55043.patch	6,8 ko	21 juin 2021	Emmanuel Cazenave
0002-provisionning-grab-existing-roles-by-uuid-in-one-que.patch	1,7 ko	21 juin 2021	Emmanuel Cazenave
0003-provisionning-delete-superfluous-roles-in-one-query-.patch	1,69 ko	21 juin 2021	Emmanuel Cazenave
0004-provisionning-delete-roles-in-one-query-when-deprovi.patch	1,52 ko	21 juin 2021	Emmanuel Cazenave
0001-provisionning-count-db-queries-55043.patch	6,8 ko	22 juin 2021	Emmanuel Cazenave
0002-provisionning-grab-existing-roles-by-uuid-in-one-que.patch	2,37 ko	22 juin 2021	Emmanuel Cazenave
0003-provisionning-delete-superfluous-roles-in-one-query-.patch	1,69 ko	22 juin 2021	Emmanuel Cazenave
0004-provisionning-delete-roles-in-one-query-when-deprovi.patch	1,52 ko	22 juin 2021	Emmanuel Cazenave