w.c.s. - Bug #46674

uwsgi peut interrompre un job d'action de masse

16 septembre 2020 07:59 - Emmanuel Cazenave

Statut:	Fermé	**Début:**	16 septembre 2020
Priorité:	Normal	**Echéance:**	
Assigné à:	Frédéric Péters	**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Non	**Planning:**	Non

Description

A coup de fd.write('pid %s now %s start\n' % (os.getpid(), datetime.datetime.now())) dans backoffice/mangement.py pour savoir quel process exécute le job, exemple ici PID 31630, constater ensuite que le statut des demandes visées n'a pas l'air de bouger, puis ensuite :

```
sudo grep 31630 /var/log/syslog
.....
Sep 16 09:47:31 publik-prod uwsgi[23348]: ...The work of process 31630 is done. Seeya!
Sep 16 09:48:32 publik-prod uwsgi[23348]: Wed Sep 16 09:48:32 2020 - worker 9 (pid: 31630) is taki
ng too much time to die...NO MERCY !!!

Sep 16 09:48:33 publik-prod uwsgi[23348]: worker 9 killed successfully (pid: 31630)
```

Demandes liées:

Lié à w.c.s. - Development #48407: pouvoir qualifier de "longs" certains afte...	**Fermé**	**09 novembre 2020**

Historique

#2 - 16 septembre 2020 11:02 - Thomas Noël

Mon analyse : on lance des threads pour faire les tâches asynchrones, elles peuvent durer. Plus tard le worker associé arrive en fin de vie (parce qu'on les renouvelle avec « max-requests = 500 » et « max-worker-lifetime = 7200 ») mais uwsgi n'arrive pas à tuer le worker, parce que le thread le bloque. Et uwsgi fini par se lasser et envoyer un SIGKILL.

Il faudrait regarder si un paramètre permet de modifier ce SIGKILL qui semble être envoyé une minute après, et temporiser à 30 minutes par exempler (bien qu'un jour ça ne suffira pas).

#3 - 16 septembre 2020 11:13 - Thomas Noël

C'est « worker-reload-mercy » qui est par défaut à 60 secondes. Tu peux essayer ça, genre le passer à 1800 ?

#4 - 16 septembre 2020 11:22 - Emmanuel Cazenave

Je n'ai plus mon cas d'usage sous la main. Mais ça parait un peu casse gueule, genre les worker reloadent pas pour une toute autre raison qu'une thread de job en train de tourner et pouf tout est bloqué pendant 30 min non ?

#5 - 16 septembre 2020 11:53 - Thomas Noël

Emmanuel Cazenave a écrit :

> Je n'ai plus mon cas d'usage sous la main. Mais ça parait un peu casse gueule, genre les worker reloadent pas pour une toute autre raison qu'une thread de job en train de tourner et pouf tout est bloqué pendant 30 min non ?

Si. De toute façon avoir des threads de plusieurs minutes derrière uwsgi, c'est casse-gueule :/

#6 - 16 septembre 2020 12:08 - Frédéric Péters

- Assigné à mis à Frédéric Péters

Quand uwsgi est utilisé il y aurait à basculer sur un dispositif particulier, à créer à partir de https://uwsgi-docs.readthedocs.io/en/latest/PythonDecorators.html (et je m'assigne pour regarder, mais ça ne sera pas tout de suite).

En attendant, "pouf tout est bloqué pendant 30 min non ?" je ne vois pas ce qui serait bloqué; mais peu importe, pour le moment on peut retirer les max-requests / max-worker-lifetime sur ce serveur, ou regarder pour mettre worker-reload-mercy; l'idée du "bloqué pendant 30 minutes" ce serait "tous les workers sont occupés par des afterjobs qui durent des plombes" ? Je n'ai pas l'impression vu qu'uwsgi pourra de toute façon toujours créer

de nouveaux workers.

#7 - 16 septembre 2020 13:23 - Emmanuel Cazenave

Au final je m'en suis sorti en augmentant max-worker-lifetime à 72000 mais sans toucher à max-request. J'ai tout de même du lancer l'action de masse plusieurs fois pour que tout passe. Je n'ai pas une vue précise sur tous les échecs.

J'ai observé ce qui est dans la description du ticket trois fois d'affilée, avec alors max-worker-lifetime à la valeur qu'on lui donne dans debian/uwsgi.ini 7200.
Et à chaque fois seulement une minute max qui s'écoulait entre le lancement de l'action et l'observation que ça n'avançait plus, confirmé dans la foulé par les logs de uwsgi.

Une minute c'est pas beaucoup, mais ce serveur est bien sollicité, donc je suppose que sur ces trois tentatives le job s'est retrouvé pris par un worker qui n'était déjà pas loin de son max-requests (500) ou alors vraiment pas de chance un worker pas loin de la fin de ses 7200 secondes de vie (bref j'arrête ici les conjectures).

#8 - 16 septembre 2020 15:06 - Thomas Noël

Emmanuel, tant que tu laisses "max-request" tu cours un risque. Si ton afterjob est lancé alors que le worker répondait à sa 490ème requête, dix clics plus tard il va être tué par uwsgi (et ça va vite, quand même).

Frédéric, « l'idée du "bloqué pendant 30 minutes" » je pense qu'on peut imaginer un pépin lors d'un "restart" de wcs. Mais franchement, on s'en fout un peu. C'est quand même pas terrible ce kill-9 violent au bout de 60 secondes, à mon avis.

Je poserais quand même bien le worker-reload-mercy = 600 sur ton instance, ne serait-ce que pour voir comment ça va jouer "la prochaine fois".

Et oui, la bonne soluce c'est plutôt https://uwsgi-docs.readthedocs.io/en/latest/PythonDecorators.html

#9 - 09 novembre 2020 14:16 - Frédéric Péters

- Lié à Development #48407: pouvoir qualifier de "longs" certains afterjobs ajouté

#10 - 18 décembre 2020 16:39 - Frédéric Péters

- Statut changé de Nouveau à Fermé

Corrigé via #48407.