Authentic 2 - Bug #35629

trace lors des logs, mauvais objet request

28 août 2019 10:01 - Frédéric Péters

Statut:	Fermé	**Début:**	28 août 2019
Priorité:	Normal	**Echéance:**	
Assigné à:		**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Oui	**Planning:**	Non

Description

Depuis 0f17a562 dans #35302, je me trouve en local (runserver) avec des :

```
Traceback (most recent call last):
  File "/usr/lib/python2.7/wsgiref/handlers.py", line 86, in run
    self.finish_response()
  File "/usr/lib/python2.7/wsgiref/handlers.py", line 131, in finish_response
    self.close()
  File "/usr/lib/python2.7/wsgiref/simple_server.py", line 33, in close
    self.status.split(' ',1)[0], self.bytes_sent
  File "/usr/lib/python2.7/BaseHTTPServer.py", line 433, in log_request
    self.requestline, str(code), str(size))
  File "/home/fred/src/eo/venv1.11/local/lib/python2.7/site-packages/django/core/servers/basehttp.
py", line 124, in log_message
    level(format, *args, extra=extra)
  File "/usr/lib/python2.7/logging/__init__.py", line 1174, in info
    self._log(INFO, msg, args, **kwargs)
  File "/usr/lib/python2.7/logging/__init__.py", line 1293, in _log
    self.handle(record)
  File "/usr/lib/python2.7/logging/__init__.py", line 1303, in handle
    self.callHandlers(record)
  File "/usr/lib/python2.7/logging/__init__.py", line 1343, in callHandlers
    hdlr.handle(record)
  File "/usr/lib/python2.7/logging/__init__.py", line 762, in handle
    rv = self.filter(record)
  File "/usr/lib/python2.7/logging/__init__.py", line 617, in filter
    if not f.filter(record):
  File "/home/fred/src/eo/authentic/src/authentic2/log_filters.py", line 43, in filter
    record.ip = request.META.get('REMOTE_ADDR', self.DEFAULT_IP)
AttributeError: '_socketobject' object has no attribute 'META'
```

J'ai dégagé ça en local via,

```
          request = record.request = getattr(record, 'request', middleware.StoreRequestMiddleware.g
et_request())
+         if request and not hasattr(request, 'META'):
+             record.request = None
```

mais il doit y avoir davantage à creuser.

Révisions associées

Révision e18a4852 - 11 septembre 2019 11:06 - Frédéric Péters

misc: ignore non-request objects in log filters (#35629)

Historique

#1 - 28 août 2019 10:18 - Éloi Rivard

Pour récupérer l'IP à partir d'une socket :

```
+                 if not hasattr(request, "META"):
```

```
+                    record.id = request.getpeername()[0]
+
+                else:
+                    record.ip = request.META.get('REMOTE_ADDR', self.DEFAULT_IP)
```

#2 - 10 septembre 2019 22:04 - Frédéric Péters

- Fichier 0001-misc-ignore-non-request-objects-in-log-filters-35629.patch ajouté

- Statut changé de Nouveau à Solution proposée

- Patch proposed changé de Non à Oui

#3 - 11 septembre 2019 10:29 - Emmanuel Cazenave

- Statut changé de Solution proposée à Solution validée

#4 - 11 septembre 2019 11:06 - Frédéric Péters

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

```
commit e18a48522f0408656b3135d2b3921b1ce23ee171
Author: Frédéric Péters <fpeters@entrouvert.com>
Date:   Tue Sep 10 22:02:45 2019 +0200

    misc: ignore non-request objects in log filters (#35629)
```

#5 - 11 septembre 2019 15:17 - Benjamin Dauvergne

Bizarre que ce filtre soit actif sur un authentic-multitenant.

#6 - 11 septembre 2019 16:14 - Frédéric Péters

C'est ma config locale, elle peut ne pas tout à fait correspondre à ce qu'on a sur les serveurs. (mais la même erreur a été rencontrée via hobo sur les serveurs).

#7 - 11 septembre 2019 18:44 - Benjamin Dauvergne

Frédéric Péters a écrit :

> C'est ma config locale, elle peut ne pas tout à fait correspondre à ce qu'on a sur les serveurs. (mais la même erreur a été rencontrée via hobo sur les serveurs).

Ça dans un sens c'est plus logique, après le patch est là depuis le 5 mars je ne vois pas ce qui a changé récemment pour que ça arrive.

#8 - 11 septembre 2019 18:46 - Benjamin Dauvergne

Benjamin Dauvergne a écrit :

> Frédéric Péters a écrit :
>
> > C'est ma config locale, elle peut ne pas tout à fait correspondre à ce qu'on a sur les serveurs. (mais la même erreur a été rencontrée via hobo sur les serveurs).
>
> Ça dans un sens c'est plus logique, après le patch est là depuis le 5 mars je ne vois pas ce qui a changé récemment pour que ça arrive.

Je vois que sur hobo c'est aussi sur du devinst par sur un vrai serveur.

#9 - 11 septembre 2019 19:18 - Frédéric Péters

Côté hobo c'est sur une installation serveur (dans le ticket rencontré à GL, rencontré aussi hier côté imio).

#10 - 11 septembre 2019 19:34 - Benjamin Dauvergne

Frédéric Péters a écrit :

> Côté hobo c'est sur une installation serveur (dans le ticket rencontré à GL, rencontré aussi hier côté imio).

La trace que je vois sur #36017 c'est un runserver pas gunicorn ou uwsgi, on est d'accord ?

#11 - 11 septembre 2019 20:03 - Frédéric Péters

Oui, tout à fait, installation serveur, mais runserver temporairement exécuté sur cette base.

#12 - 12 septembre 2019 08:09 - Benjamin Dauvergne

Frédéric Péters a écrit :

> Oui, tout à fait, installation serveur, mais runserver temporairement exécuté sur cette base.

Ok j'ai trouvé le basehttpserver de Django se trouve dépendre de SocketServer.TcpServer qui pose l'objet socket retourné par socket.accept() dans self.request, et donc il se trouve que les logs venant de basehttpserver ne sont pas compatibles avec l'usage de request dans extra par les autres logs de Django; ça veut juste dire qu'il faudrait un traitement particulier du logger django.server pour reproduire la config de base de Django au niveau formatage et filtre :

```
# django/utils/log.py
 29     'formatters': {
 30         'django.server': {
 31             '()': 'django.utils.log.ServerFormatter',
 32             'format': '[%(server_time)s] %(message)s',
 33         }
 34     },
```

On va se contenter de ta correction mais c'est bon de savoir d'où tout ça vient.

#13 - 12 septembre 2019 11:17 - Emmanuel Cazenave

Au passage, maintenant en local je fais des trucs comme ça :

```
import logging.config

LOGGING_CONFIG = None # pour dire à django ne fais rien sur la config de log, c'est moi qui la gère de A à Z

LOGGING = {
    # définir une config from scratch comme je veux
}
logging.config.dictConfig(LOGGING)
```

Ça a selon moi l'immense avantage de permettre d'avoir toute la config de log immédiatement sous les yeux, et donc de redevenir compréhensible au lieu d'être aiguille dans botte de foin.

#14 - 12 septembre 2019 11:20 - Benjamin Dauvergne

Emmanuel Cazenave a écrit :

> Au passage, maintenant en local je fais des trucs comme ça :
>
> [...]
>
> Ça a selon moi l'immense avantage de permettre d'avoir toute la config de log immédiatement sous les yeux, et donc de redevenir compréhensible au lieu d'être aiguille dans botte de foin.

Ouaip mais ça n'empêche de devoir comprendre comment les softs utilisent logging, c'est dommage mais formateurs sont intimement liés par exemple au contenu des objets record (si tu as un formateur "%(host)s" mais que record.host n'existe pas ça va planter :/ au lieu d'afficher un contenu par défaut, ou "None" ou "-"); idem ici nos filtre supposent que record.request est tooujours un objet Request comme tout ce qui est loggé par défaut sur le logger django.request mais finalement ce n'est pas le cas pour le logger django.server.

#15 - 15 septembre 2019 17:15 - Frédéric Péters

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

Fichiers

0001-misc-ignore-non-request-objects-in-log-filters-35629.patch	963 octets 10 septembre 2019	Frédéric Péters