Passerelle - Development #55516

logging des exceptions, avec 2.2 plus de traceback

12 juillet 2021 09:28 - Frédéric Péters

Statut:	Fermé	**Début:**	12 juillet 2021
Priorité:	Normal	**Echéance:**	
Assigné à:	Valentin Deniaud	**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Oui	**Planning:**	Non

Description

Je pose ça ici parce qu'on a dans hobo la configuration commune du logging.

Avec le passage à django 2.2 il me semble qu'on a perdu, dans les mails d'erreurs, la section "Traceback".

exemple avant :

```
Date: Sat, 26 Jun 2021 14:10:34 +0200
From: admin+passerelle.node1.prod.saas.entrouvert.org@entrouvert.org
Subject: [passerelle-vincennes.entrouvert.com] ERROR (EXTERNAL IP): Internal Server Error:
        /clicrdv/clicrdv-passeport/interventions/[form_var_clicrdv_intervention_raw]/[form_var_cli
crdv_date_raw]/times
To: admin+passerelle.prod@entrouvert.com

Internal Server Error: /clicrdv/clicrdv-passeport/interventions/[form_var_clicrdv_intervention_raw
]/[form_var_clicrdv_date_raw]/times

Exception at /clicrdv/clicrdv-passeport/interventions/[form_var_clicrdv_intervention_raw]/[form_va
r_clicrdv_date_raw]/times
Failed to retrieve theme

Request Method: GET
Request URL:
https://passerelle-vincennes.entrouvert.com/clicrdv/clicrdv-passeport/interventions/%5Bform_var_cl
icrdv_intervention_raw%5D/%5Bform_var_clicrdv_
date_raw%5D/times?apikey=vinceo&orig=demarches.vincennes.fr&algo=sha256&timestamp=2021-06-26T12%3A
10%3A33Z&nonce=f42f9d87a5fdf70e37b10859390bd41
7&signature=N3GsQJjYuUYSno0Mn6vxYYnyKHGmrxOVcmaaHnGk0H4%3D
Django Version: 1.11.29
Python Executable: /usr/bin/uwsgi-core
Python Version: 3.7.3
Python Path: ['.', '', '/usr/lib/python37.zip', '/usr/lib/python3.7', '/usr/lib/python3.7/lib-dynl
oad',
'/usr/local/lib/python3.7/dist-packages', '/usr/lib/python3/dist-packages']
Server time: sam, 26 Jui 2021 14:10:34 +0200
Installed Applications:
''
Installed Middleware:
''

Traceback:

File "/usr/lib/python3/dist-packages/django/core/handlers/exception.py" in inner
...
```

Et 1er juillet passage 2.2 :

```
Date: Thu, 01 Jul 2021 13:59:08 +0200
From: admin+passerelle.node1.prod.saas.entrouvert.org@entrouvert.org
Subject: [passerelle-vincennes.entrouvert.com] ERROR (EXTERNAL IP): Internal Server Error:
        /clicrdv/clicrdv-passeport/interventions/[form_var_clicrdv_intervention_raw]/[form_var_cli
crdv_date_raw]/times
```

```
To: admin+passerelle.prod@entrouvert.com

Internal Server Error: /clicrdv/clicrdv-passeport/interventions/[form_var_clicrdv_intervention_raw
]/[form_var_clicrdv_date_raw]/times

Report at /clicrdv/clicrdv-passeport/interventions/[form_var_clicrdv_intervention_raw]/[form_var_c
licrdv_date_raw]/times
Internal Server Error: /clicrdv/clicrdv-passeport/interventions/[form_var_clicrdv_intervention_raw
]/[form_var_clicrdv_date_raw]/times

Request Method: GET
Request URL:
https://passerelle-vincennes.entrouvert.com/clicrdv/clicrdv-passeport/interventions/%5Bform_var_cl
icrdv_intervention_raw%5D/%5Bform_var_clicrdv_
date_raw%5D/times?apikey=vinceo&orig=demarches.vincennes.fr&algo=sha256&timestamp=2021-07-01T11%3A
59%3A08Z&nonce=eff054410f707186388f737590021d9
d&signature=0cg%2B8sVSoYt/c8jSF2apxEXNBKsUvjdi5e7T35JWfNw%3D
Django Version: 2.2.24
Python Executable: /usr/bin/uwsgi-core
Python Version: 3.7.3
Python Path: ['.', '', '/usr/lib/python37.zip', '/usr/lib/python3.7', '/usr/lib/python3.7/lib-dynl
oad',
'/usr/local/lib/python3.7/dist-packages', '/usr/lib/python3/dist-packages']
Server time: jeu, 1 Jul 2021 13:59:08 +0200
Installed Applications:
''
Installed Middleware:
''

Request information:
USER: AnonymousUser

GET:
...
```

Demandes liées:

Lié à Passerelle - Development #56139: Appeler directement le log Django des ...	**Fermé**	**13 août 2021**

Révisions associées

Révision f6aafa93 - 20 juillet 2021 14:04 - Valentin Deniaud

utils: include exception logging in log_http_request (#55516)

Historique

#1 - 12 juillet 2021 16:22 - Frédéric Péters

À un moment il y a ça qui apparait (8f8c54f70bfa3aa8e311514297f1eeded2c32593, ticket django 25099)

```
        # Since we add a nicely formatted traceback on our own, create a copy
        # of the log record without the exception data.
        no_exc_record = copy(record)
        no_exc_record.exc_info = None
        no_exc_record.exc_text = None
```

et ça pourrait venir de là mais d'une manière ou d'une autre sur le SaaS de dev il y a de toute façon des tracebacks dans les emails d'erreurs (j'y ai fait des modifs locales dans passerelle pour tester); je ne pige absolument pas comment tout tourne et je ne vais pas planter la recette pour creuser ça.

#2 - 12 juillet 2021 16:28 - Benjamin Dauvergne

~~Je ne trouve pas le commit 8f8c54f70bfa3aa8e311514297f1eeded2c32593; c'est bien un commit du dépôt hobo ?~~ Ok dépôt Django.

#3 - 12 juillet 2021 16:29 - Frédéric Péters

Non, je fais référence au commit django correspondant au numéro de ticket django cité.

#4 - 19 juillet 2021 21:56 - Frédéric Péters

- Projet changé de Hobo à Passerelle

Ça ne semble concerner que passerelle, des chances que ça soit lié à ce qui se fait dans ProxyLogger::_log.

#5 - 19 juillet 2021 21:57 - Frédéric Péters

- Tracker changé de Support à Bug

- Sujet changé de logging des exceptions, avec 2.2 plus de traceback ? à logging des exceptions, avec 2.2 plus de traceback

#6 - 20 juillet 2021 10:47 - Valentin Deniaud

- Assigné à mis à Valentin Deniaud

#7 - 20 juillet 2021 12:24 - Valentin Deniaud

- Fichier 0001-utils-include-exception-logging-in-log_http_request-.patch ajouté

- Tracker changé de Bug à Development

- Statut changé de Nouveau à Solution proposée

- Patch proposed changé de Non à Oui

J'ai trempé mon pdb dans le système de logs de Django et j'arrive à ça.

En fait il faut distinguer les traces « normales », tu vas sur une vue et ça plante, des traces qui résultent d'appel à un connecteur passerelle. Je pense que sur passerelle on a toujours bien actuellement les traces normales, j'ose pas torturer la recette pour confirmer.

Donc en se concentrant sur les appels aux connecteurs, caractérisés par une grosse mécanique commune à base de décorateur to_json et de ProxyLogger, on remarque que l'objet exception n'est jamais loggé, on logge sa représentation textuelle. Quand le tout est dispatché dans les handlers, ben on a pas accès à l'exception donc à la traceback. Corollaire, le handler AdminEmailHandler n'a pas accès à la traceback.

Mystère sur pourquoi ça marchait en 1.11, peut-être un signal quelque part, je n'ai pas d'anciennes traces pour pousser l'investigation (est-ce que d'autres traces que celles de clicrdv avaient une traceback ?).

#8 - 20 juillet 2021 12:29 - Frédéric Péters

> Mystère sur pourquoi ça marchait en 1.11, peut-être un signal quelque part, je n'ai pas d'anciennes traces pour pousser l'investigation (est-ce que d'autres traces que celles de clicrdv avaient une traceback ?).

Oui, par exemple :

```
Date: Wed, 17 Mar 2021 09:01:32 +0100
From: admin+passerelle.node1.prod.saas.entrouvert.org@entrouvert.org
Subject: [passerelle.icitoyen.fr] ERROR (EXTERNAL IP): Error occurred while processing request
To: admin+passerelle.prod@entrouvert.com

Error occurred while processing request

APIError at /api-particulier/api_particulier/situation-familiale/
API-particulier platform "prod" returned a non 200 status 500: {'error': 'internal_server_error', 'reason': 'E
rreur serveur', 'message': 'Erreur serveur'}

Request Method: GET
Request URL:
https://passerelle.icitoyen.fr/api-particulier/api_particulier/situation-familiale/?code_postal=42300&numero_a
llocataire=None&orig=demarches.icitoyen.fr&algo=sha256&timestamp=2021-03-17T08%3A01%3A31Z&nonce=4567c09f1d6991
6dc4349f7756fff475&signature=ILz2jA06ZK7dIz1
tBbU6aYxIa4fUYCV6xOLVtuFzGCM%3D
Django Version: 1.11.29
Python Executable: /usr/bin/uwsgi-core
Python Version: 3.7.3
Python Path: ['.', '', '/usr/lib/python37.zip', '/usr/lib/python3.7', '/usr/lib/python3.7/lib-dynload', '/usr/
local/lib/python3.7/dist-packages', '/usr/lib/python3/dist-packages']
Server time: mer, 17 Mar 2021 09:01:32 +0100
Installed Applications:
''
Installed Middleware:
''

Traceback:

File "/usr/lib/python3/dist-packages/passerelle/utils/jsonresponse.py" in api
  131.             resp = f(*args, **kwargs)
```

```
File "/usr/lib/python3/dist-packages/passerelle/views.py" in perform
  511.         result = self.endpoint(request, **params)

File "/usr/lib/python3/dist-packages/passerelle/apps/api_particulier/models.py" in v2_situation_familiale
  285.             user=user,

File "/usr/lib/python3/dist-packages/passerelle/apps/api_particulier/models.py" in get
  124.                     'content': data,

Exception Type: APIError at /api-particulier/api_particulier/situation-familiale/
Exception Value: API-particulier platform "prod" returned a non 200 status 500: {'error': 'internal_server_err
or', 'reason': 'Erreur serveur', 'message': 'Erreur serveur'}
```

#9 - 20 juillet 2021 12:30 - Frédéric Péters

Ou encore, (mais celle-ci est avant le système d'endpoint),

```
Date: Wed, 07 Apr 2021 14:38:05 +0200
From: admin+montoulouse-prod@entrouvert.com
Subject: [prod montoulouse.fr passerelle][passerelle.eservices.toulouse-metropole.fr] ERROR (EXTERNAL IP): Int
ernal Server Error:
        /toulouse-axel/axel-famille/clae_children_activities_info
To: admin+prod.montoulouse.combo@entrouvert.com

Internal Server Error: /toulouse-axel/axel-famille/clae_children_activities_info

OperationalError at /toulouse-axel/axel-famille/clae_children_activities_info
FATAL:  les emplacements de connexions restants sont réservés pour les connexions
superutilisateur non relatif à la réplication

Request Method: GET
Request URL:
https://passerelle.eservices.toulouse-metropole.fr/toulouse-axel/axel-famille/clae_children_activities_info?bo
oking_date=2021-04-07&NameID=8d9cbb97bf074e45a8499d5f80e0220e&orig=demarches-montoulouse.eservices.toulouse-me
tropole.fr&algo=sha256&timestamp=2021-04-07T12%3A38%3A05Z&nonce=eee15e994d8840415f3a0617dcf42687&
signature=aJkLmPDug%2BQyEtRAnXLrbRCoV//yqXm87e2hML55hJY%3D
Django Version: 1.11.29
Python Executable: /usr/bin/uwsgi-core
Python Version: 3.7.3
Python Path: ['.', '', '/usr/lib/python37.zip', '/usr/lib/python3.7', '/usr/lib/python3.7/lib-dynload', '/usr/
local/lib/python3.7/dist-packages', '/usr/lib/python3/dist-packages']
Server time: mer, 7 Avr 2021 14:38:05 +0200
Installed Applications:
''
Installed Middleware:
''

Traceback:

File "/usr/lib/python3/dist-packages/django/db/backends/base/base.py" in ensure_connection
  213.                 self.connect()

File "/usr/lib/python3/dist-packages/django/db/backends/base/base.py" in connect
  189.         self.connection = self.get_new_connection(conn_params)

File "/usr/lib/python3/dist-packages/django/db/backends/postgresql/base.py" in get_new_connection
  176.         connection = Database.connect(**conn_params)

File "/usr/lib/python3/dist-packages/psycopg2/__init__.py" in connect
  130.     conn = _connect(dsn, connection_factory=connection_factory, **kwasync)

      The above exception (FATAL:  les emplacements de connexions restants sont réservés pour les connexions
superutilisateur non relatif à la réplication
) was the direct cause of the following exception:

File "/usr/lib/python3/dist-packages/django/core/handlers/exception.py" in inner
  41.             response = get_response(request)

File "/usr/lib/python3/dist-packages/django/core/handlers/base.py" in _get_response
  187.                 response = self.process_exception_by_middleware(e, request)
...
```

#10 - 20 juillet 2021 13:52 - Valentin Deniaud

Valentin Deniaud a écrit :

> Je pense que sur passerelle on a toujours bien actuellement les traces normales, j'ose pas torturer la recette pour confirmer.

La réponse est oui, j'ai bien la trace complète de #55667 dans mes mails.

#11 - 20 juillet 2021 13:58 - Frédéric Péters

- Statut changé de Solution proposée à Solution validée

Envoyons-ça dès maintenant, c'est handicapant au possible de ne pas avoir de détails sur les erreurs dans les endpoints.

#12 - 20 juillet 2021 14:14 - Valentin Deniaud

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

```
commit f6aafa93c39abe55fa809cd7475e4c7e5ef64dd0
Author: Valentin Deniaud <vdeniaud@entrouvert.com>
Date:   Tue Jul 20 12:15:44 2021 +0200

    utils: include exception logging in log_http_request (#55516)
```

#13 - 20 juillet 2021 17:17 - Frédéric Péters

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

#15 - 13 août 2021 09:39 - Benjamin Dauvergne

Je ne sais pas pour le reste mais pour la trace ClicRdv ça vient d'une interaction entre une utilisation direct de jsonresponse qui produit une 500 en cas d'erreur et le fait que django 2.2 en tout cas log toutes les requêtes 500 en faisant passé le message de log pour une exception (mais donc sans trace) :

```
# là on voit que to_json renvoie 500 par défaut
class to_json(object):
    def __init__(self, error_code=500, logger=None, **kwargs):
.....
            else:
                logger.warning("Error occurred while processing request", extra=extras) <- pas de log visible,
 c'est une exception lambda qui pose souci pas une APIError
.....
            data = self.err_to_response(e)
            if getattr(e, 'err_code', None):
                data['err'] = e.err_code
            if getattr(e, 'http_status', None):
                status = e.http_status
            elif isinstance(e, (ObjectDoesNotExist, Http404)):
                status = 404
            elif isinstance(e, PermissionDenied):
                status = 403
            else:
                status = self.error_code <-- ici
```

et coté Django :

```
# log_response est appelé sur chaque réponse retournée
def log_response(message, *args, response=None, request=None, logger=request_logger, level=None, exc_info=None
):
    """
    Log errors based on HttpResponse status.

    Log 5xx responses as errors and 4xx responses as warnings (unless a level
    is given as a keyword argument). The HttpResponse status_code and the
    request are passed to the logger's extra parameter.
    """
    # Check if the response has already been logged. Multiple requests to log
    # the same response can be received in some cases, e.g., when the
    # response is the result of an exception and is logged at the time the
    # exception is caught so that the exc_info can be recorded.
    if getattr(response, '_has_been_logged', False):
        return

    if level is None:
```

```
        if response.status_code >= 500:
            level = 'error' <-- ici ça passe le niveau du log en erreur
        elif response.status_code >= 400:
            level = 'warning'
        else:
            level = 'info'
```

```
    getattr(logger, level)(
        message, *args,
        extra={
            'status_code': response.status_code,
            'request': request,
        },
        exc_info=exc_info,
    )
```

et dans AdminEmailHandler on a ce bout de code qui va prendre record.getMessage() et faire passer ça pour une exception_value :

```
        # Since we add a nicely formatted traceback on our own, create a copy
        # of the log record without the exception data.
        no_exc_record = copy(record)
        no_exc_record.exc_info = None
        no_exc_record.exc_text = None
```

```
        if record.exc_info:
            exc_info = record.exc_info
        else:
            exc_info = (None, record.getMessage(), None)
```

C'est comme ça qu'on se retrouve avec un Internal Server Error: <request.path> mais sans trace. Pour moi on mélange deux choses, le passage à 2.2 et certainement deux types d'erreur différentes, la première est une exception qui est remonté jusqu'à Django (aussi je ne sais pas trop comment on obtient un "Failed to retrieve theme" dans une vue qui renvoie du JSON) pas la deuxième, c'est une 500 interne.

Je ne sais pas trop comment on retourne une 500 sans que Django ne log en erreur alors qu'on ne veut visiblement pas (ou en tout cas si on le souhaite c'est dans jsonresponse qu'il faut traiter ça); on peut poser '__already_logged__' sur HttpResponse mais ça supprime tout le log "HTTP" sur le logger "django.request" (qu'on fait déjà 2 fois dans nginx et uwsgi...).

#16 - 13 août 2021 09:56 - Benjamin Dauvergne

Après réflexion je pense qu'il faut juste ajouter au logger.warning par défaut de to_json un django.utils.log.log_response, ça posera *already_logged* et on pourra passer sys.exc_info() et récupérer le comportement par défaut de Django tout en retournant une erreur formatée en JSON (et ça ne le fera que sur une 500, i.e. pas une exception d'un type géré comme APIError qui retournent une 400).

#18 - 13 août 2021 10:18 - Benjamin Dauvergne

Benjamin Dauvergne a écrit :

> Après réflexion je pense qu'il faut juste ajouter au logger.warning par défaut de to_json un django.utils.log.log_response, ça posera *already_logged* et on pourra passer sys.exc_info() et récupérer le comportement par défaut de Django tout en retournant une erreur formatée en JSON (et ça ne le fera que sur une 500, i.e. pas une exception d'un type géré comme APIError qui retournent une 400).

En fait on log déjà bien ce qu'il faut quand il faut via le if ..getattr(e, 'log_error', True):... on est juste pollué par des mails d'erreur inutiles par Django parce qu'on le laisse décider du niveau de log.

PS: mais si quelqu'un peut sortir un mail d'erreur dont il est sûr qu'elle correspond bien à quelque chose qui aurait du contenir un traceback, je le veux bien ici; en attendant #56139.

#19 - 13 août 2021 10:20 - Benjamin Dauvergne

- Lié à Development #56139: Appeler directement le log Django des réponses dans to_json pour en définir le niveau ajouté

Fichiers

0001-utils-include-exception-logging-in-log_http_request-.patch	2,54 ko	20 juillet 2021	Valentin Deniaud