Passerelle - Bug #31058

iParapheur: bug sur une erreur de transport

04 mars 2019 09:52 - Nicolas Roche

Statut:	Fermé	**Début:**	04 mars 2019
Priorité:	Normal	**Echéance:**	
Assigné à:	Nicolas Roche	**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Oui	**Planning:**	

Description

Si une erreur survient entre la récupération du WSDL et l'appel à un service (par exemple ping),
alors on reçoit cette erreur :

```
err_class    "AttributeError"
err_desc    "'NoneType' object has no attribute 'read'"
data    null
err    1
```

A la place de la vraie erreur qui n'est pas remontée :

```
err_class    "passerelle.utils.jsonresponse.APIError"
err_desc    "Transport Error: (None, u\"Invalid URL 'plouf': No schema supplied. Perhaps you meant
 http://plouf?\")"
data    null
err    1
```

Le test suivant permet de mettre en évidence l'erreur :

```
@mock.patch('passerelle.utils.Request.get')
@mock.patch('passerelle.utils.Request.post', side_effect=ConnectionError('mocked error'))
@mock.patch('passerelle.contrib.iparapheur.soap.HttpAuthenticated.open')
def test_send_transporterror(http_open, mocked_post, mocked_get, app, conn, xmlmime, wsdl_file):
    http_open.return_value = open(xmlmime)
    mocked_get.return_value = mock.Mock(content=open(wsdl_file).read(), status_code=200)
    url = reverse('generic-endpoint', kwargs={'connector': 'iparapheur',
                                              'endpoint': 'ping', 'slug': conn.slug})
    url += '?apikey=%s' % API_KEY
    resp = app.get(url)
    assert resp.json['err'] == 1
    assert resp.json['data'] is None
    assert 'mocked error' in resp.json['err_desc']
```

Demandes liées:

Lié à Passerelle - Development #30258: iParapheur: pouvoir surcharger les URL...	**Fermé**	**31 janvier 2019**
Lié à Passerelle - Support #31120: iparapheur: passer ce connecteur à zeep	**Fermé**	**06 mars 2019**

Historique

#1 - 04 mars 2019 09:55 - Nicolas Roche

- Lié à Development #30258: iParapheur: pouvoir surcharger les URL des appels webservices ajouté

#2 - 04 mars 2019 11:59 - Nicolas Roche

- Fichier 0001-iparapheur-correct-message-on-transport-error-31058.patch ajouté

- Tracker changé de Support à Bug

- Sujet changé de iParapheur: bug sur une erreur de trasport à iParapheur: bug sur une erreur de transport

- Statut changé de Nouveau à Solution proposée

- Assigné à mis à Nicolas Roche

- Patch proposed changé de Non à Oui

#3 - 04 mars 2019 15:21 - Benjamin Dauvergne

Et avec juste TransportError(e, None) ça donne quoi ?

#4 - 04 mars 2019 18:27 - Nicolas Roche

ça donne la même erreur :

```
err_class    "AttributeError"
err_desc    "'NoneType' object has no attribute 'read'"
data    null
err    1
```

Actuellement ce troisième argument permet de détecter l'exception et de la renvoyer proprement :

```
# Exception different from suds.tranport.TransportError
if not exc.args or not isinstance(exc.args[0], tuple):
    raise exc
# TransportError Exception
raise APIError('Transport Error: %s' % exc)
```

#5 - 04 mars 2019 18:48 - Benjamin Dauvergne

Nicolas Roche a écrit :

> ça donne la même erreur :
> [...]
> Actuellement ce troisième argument permet de détecter l'exception et de la renvoyer proprement :
> [...]

Oulala oui je pense que ça date d'avant le passage à request, je remplacerai ça par un simple except TransportError as e:.

#6 - 05 mars 2019 09:43 - Nicolas Roche

Ce serait l'idéal mais alors je n'arrive pas distinguer l'exception pour l'attraper :
suds/client.py::failed() intercepte l'exception pour la lever de manière générique.
Par ailleurs cette même fonction plante (et lève une autre exception) si l'on omet le 3ème argument.

```
    def failed(self, binding, error):
        """
        Request failed, process reply based on reason
        @param binding: The binding to be used to process the reply.
        @type binding: L{suds.bindings.binding.Binding}
        @param error: The http error message
        @type error: L{transport.TransportError}
        """
        status, reason = (error.httpcode, tostr(error))
        reply = error.fp.read()
        log.debug('http failed:\n%s', reply)
        if status == 500:
            if len(reply) > 0:
                r, p = binding.get_fault(reply)
                self.last_received(r)
                return (status, p)
            else:
                return (status, None)
        if self.options.faults:
            raise Exception((status, reason))
        else:
            return (status, None)
```

Je peux surcharger cette fonction mais ça me semble un peut violent non ?

#7 - 05 mars 2019 10:11 - Frédéric Péters

Perso je serais pour ignorer tout ça et plutôt créer un ticket pour passer ce connecteur à zeep. (et s'il y a le même problème dedans au moins ça sera corrigé pour tout le monde).

#8 - 05 mars 2019 10:26 - Benjamin Dauvergne

Ouaip je pense qu'on a perdu assez de temps avec suds aussi et normalement c'est peu ou prou la même API client.op(blabla), si tu te sens de faire ça Nicolas, go.

#9 - 05 mars 2019 10:51 - Nicolas Roche

ok, mais regarde un peu et je demande aussi autour de moi (Emmanuel) avant d'ouvrir le ticket.

#10 - 06 mars 2019 10:51 - Nicolas Roche

- Lié à Support #31120: iparapheur: passer ce connecteur à zeep ajouté

#11 - 12 mars 2019 18:54 - Benjamin Dauvergne

J'ai l'impression qu'on peut fermer celui-ci vu que le #31120 le rend inutile.

#12 - 15 mars 2019 15:15 - Benjamin Dauvergne

- Statut changé de Solution proposée à Information nécessaire

#13 - 18 mars 2019 13:59 - Nicolas Roche

- Statut changé de Information nécessaire à Fermé

Oui, je le ferme.

Fichiers

0001-iparapheur-correct-message-on-transport-error-31058.patch	2,13 ko	04 mars 2019	Nicolas Roche