UnivNautes (historique) - Bug #6165

UnivNautes SP update-metadata-idp: DatabaseError: too many SQL variables

17 décembre 2014 11:43 - CRIMA CRIMA

Statut:	Fermé	**Début:**	17 décembre 2014
Priorité:	Haut	**Echéance:**	
Assigné à:	Thomas Noël	**% réalisé:**	100%
Catégorie:	SP (pffedportal)	**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Non	**Planning:**	

Description

Hi,

On our instance of UnivNautes, we hit a nasty and critical bug about metadata refreshing.

We have discovered the bug when we have requested a metadata update at Renater federation for our IdP about changing x509 certificate...
From that point, UnivNautes SP has cease to work. I finally found an error message on pfSense logs :

```
Dec 15 11:54:19 eduspot update-metadata-idp: DatabaseError: too many SQL variables
```

In Univnaute web config, I have https://federation.renater.fr/renater/idps-renater-metadata.xml for both IdP URL and SP URL.

Google : no match... ok. Then explore.
In SSH shell, tried :

```
[2.0-RELEASE+UNIVNAUTES-20140807-1543][root@eduspot.univ-jfc.fr]/root(10): univnautes-update-metadata.sh
```

Same error. Then explore.

```
bash
cd /usr/local/univnautes/pffedportal
MD=/var/db/metadata-idps.xml
VIRTUAL_ENV="/usr/local/univnautes"
export VIRTUAL_ENV
PATH="$VIRTUAL_ENV/bin:$PATH"
export PATH
python ./manage.py sync-metadata --source="federation" --idp $MD
```

Same error, but on stdout, in red caracters.

Then explore :

```
[2.0-RELEASE+UNIVNAUTES-20140807-1543][root@eduspot.univ-jfc.fr]/usr/local/univnautes/pffedportal(14): ls -lh *db
-rw-r--r--  1 root  wheel    82M Dec 16 10:32 pffedportal.db
```

Tried to flush all tables in the DB, restart manage.py, same error. Restore original SQLite DB. Added --verbosity=2.

```
Updating Sciences Po Grenoble, https://shibboleth.sciencespo-grenoble.fr/idp/shibboleth
Updating Université de Bordeaux IV - Montesquieu, https://idp.u-bordeaux4.fr/idp/shibboleth
Updating Ecole des Mines de Saint-Etienne, https://shibbo.emse.fr/idp/shibboleth
Updating CROUS Rouen - Personnels, https://si.crous-rouen.fr/idp/shibboleth
DatabaseError: too many SQL variables
```

Import failed when reached about 95% of the metadata entries.

Tried to refresh matadata with univnautes-update-metadata.sh then retry manage.py.
Same behavior, but the last log line is about *another* IdP.

Then explore. Google : django sqlite "too many SQL variables".
=> http://stackoverflow.com/questions/7106016/too-many-sql-variables-error-in-django-witih-sqlite3

> Many SQL programmers are familiar with using a question mark ("?") as a host parameter. SQLite also supports named host parameters prefaced by ":", "$", or "@" and numbered host parameters of the form "?123".

> To prevent excessive memory allocations, the maximum value of a host parameter number is SQLITE_MAX_VARIABLE_NUMBER, which defaults to 999.

This is a SQLite *compilation* parameter. No hope for dynamic tweaking.

Then explore : Replaced --verbosity=3.
Last SQL request found in /var/log/authentic.log is very huge and follow this pattern :

```
2014-12-16 16:27:47 authentic: [DEBUG] django.db.backends.(0.002) SELECT "saml_libertyprovider"."id",
"saml_libertyprovider"."name", "saml_libertyprovider"."slug", "saml_libertyprovider"."entity_id",
"saml_libertyprovider"."entity_id_sha1", "saml_libertyprovider"."protocol_conformance", "saml_libertyprovider"."metadata",
"saml_libertyprovider"."public_key", "saml_libertyprovider"."ssl_certificate", "saml_libertyprovider"."ca_cert_chain",
"saml_libertyprovider"."federation_source" FROM "saml_libertyprovider" WHERE ("saml_libertyprovider"."federation_source" =
federation  AND NOT ("saml_libertyprovider"."entity_id" IN (https://wireless.balliol.ox.ac.uk/shibboleth,
https://authenticate.bvdep.com/ukfederation, [...]
```

After the huge "NOT IN", the logger dump the args array that contains a huge quantity of EntityIds. In the args array dump, I found 1842 commas. So we might have 1843 arguments (assuming no comma in URLs, seems reasonable). Crapy.

Some statistics about renater-matadata global XML file :

```
[root@eduspot /var/db]# grep '<md:IDPSSODescriptor' metadata-idps.xml | wc -l
235
[root@eduspot /var/db]# grep '<md:SPSSODescriptor' metadata-idps.xml | wc -l
1609
[root@eduspot /var/db]# grep -E '<md:(IDP|SP)SSODescriptor' metadata-idps.xml | wc -l
1844
```

So, I think that the problem could happen with **any** big federation. Metadata number of entry should not be bound by SQLite argument count limit.

Workaround found for my case :

Basicaly I replace the URL by : https://federation.renater.fr/renater/idps-renater-metadata.xml
I don't use UnivNaute internal IdP so it don't need SP knowledge.
idps-renater-metadata.xml is significantly shorter than renater-metadata.xml and don't trigger the bug.

To apply this correctly, I change the config with web interface, then I was forced to flush SQLite copy of current metadata infos :

```
bash
cd /usr/local/univnautes/pffedportal
MD=/var/db/metadata-idps.xml
VIRTUAL_ENV="/usr/local/univnautes"
export VIRTUAL_ENV
PATH="$VIRTUAL_ENV/bin:$PATH"
export PATH
python ./manage.py sync-metadata --source="federation" --idp --delete --verbosity=2 $MD
python ./manage.py sync-metadata --source="federation" --sp  --delete --verbosity=2 $MD
python ./manage.py sync-metadata --source="federation" --idp --verbosity=2 $MD
reboot
```

Additionnal info, I've tried to instrument the code to see where the big SQL request is generated, but first guess was unsuccessful :

In /usr/local/univnautes/lib/python2.7/site-packages/authentic2/saml/management/commands/sync-metadata.py :
Some "print" added :

```
provider.name = name
provider.metadata = etree.tostring(tree, encoding='utf-8').decode('utf-8').strip()
provider.protocol_conformance = 3
provider.federation_source = options['source']
print "before provider.save()"
provider.save()
print "after provider.save()"
options['count'] = options.get('count', 0) + 1
if idp:
identity_provider, created = LibertyIdentityProvider.objects.get_or_create(
```

liberty_provider=provider)
identity_provider.enabled = True
if idp_policy:
identity_provider.idp_options_policy = idp_policy
print "before identity_provider.save()"
identity_provider.save()
print "after identity_provider.save()"

Log when this version run with --idp option and metadata-renater.xml (the big one) :

```
[...]
Updating Université de Paris 8 - Vincennes, https://idp.univ-paris8.fr/idp/shibboleth
before provider.save()
after provider.save()
before identity_provider.save()
after identity_provider.save()
Updating INSA de Lyon, https://login.insa-lyon.fr/idp/shibboleth
before provider.save()
after provider.save()
before identity_provider.save()
after identity_provider.save()
DatabaseError: too many SQL variables
```

Unfortunatly, I have never practiced Django and it's ORM so don't know how correct and make a patch proposal.

Cheers,
Ludovic Pouzenc

Révisions associées

Révision 7070683d - 16 janvier 2015 16:22 - Thomas Noël

port authentic2: fix #6165 et #6164

Historique

#1 - 15 janvier 2015 17:10 - Thomas Noël

- Statut changé de Nouveau à En cours

- Assigné à mis à Thomas Noël

Sorry for the delay... and thanks a lot for this report.

For your information, here is a patch that will be part of the next release (thanks to Benjamin Dauvergne):

```
diff --git a/authentic2/saml/management/commands/sync-metadata.py b/authentic2/saml/management/commands/sync-m
etadata.py
index befe52e..251d973 100644
--- a/authentic2/saml/management/commands/sync-metadata.py
+++ b/authentic2/saml/management/commands/sync-metadata.py
@@ -316,8 +316,10 @@ Any other kind of attribute filter policy is unsupported.
                         print 'Finally delete all providers for source: %s...' % source
                         LibertyProvider.objects.filter(federation_source=source).delete()
                     else:
-                        to_delete = LibertyProvider.objects.filter(federation_source=source)\
-                                .exclude(entity_id__in=loaded)
+                        to_delete = []
+                        for provider in LibertyProvider.objects.filter(federation_source=source):
+                            if provider.entity_id not in loaded:
+                                to_delete.append(provider)
                         for provider in to_delete:
                             if verbosity > 1:
                                 print _('Deleted obsolete provider %s') % provider.entity_id
```

#2 - 16 janvier 2015 16:29 - Thomas Noël

- Statut changé de En cours à Résolu (à déployer)

- % réalisé changé de 0 à 100

Appliqué par commit univnautes:univnautes-tools|commit:7070683d67bd4bf20fe14525df9e24b09f27a5d2.

#3 - 28 juin 2016 20:44 - Pierre Cros

- Statut changé de Résolu (à déployer) à Fermé