Optimisation et solution des problèmes de Pandora FMS

Generated:
https://pandorafms.com/manual/!777/
Permanent link:
https://pandorafms.com/manual/!777/fr/documentation/pandorafms/complex_environments_and_optimization/08_optimization
2024/10/03 18:41

Optimisation et solution des problèmes de Pandora FMS

Le serveur de Pandora FMS est capable de surveiller environ 2000 appareils (entre 5000 et 80 000 modules, selon le matériel disponible), ce qui nécessite également une configuration fine de la base de données.

De plus, dans ce chapitre, certaines techniques de détection et de dépannage des problèmes d'installation de Pandora FMS sont expliquées.

Optimisation de Percona

Tous les tests et validations sont effectués avec Percona Server for MySQL® 8 (option recommandée). En raison des similitudes entre Percona Server for MySQL 8 et MySQL® 8, il existe une grande compatibilité entre les deux solutions. Pour les besoins pratiques de ce sujet, le terme MySQL est utilisé, en gardant toujours à l'esprit que les tests sont effectués sur Percona Server for MySQL.

Pour en savoir plus sur “Sauvegarde et procédures de restauration”, veuillez suivre ce lien.

Conseils généraux

- Sauf indication contraire, l'ensemble de cette rubrique fait référence à la version 8 de MySQL.
- Voir aussi « Installation et mise à niveau vers MySQL 8 ».

Pour travailler avec des tableaux de plus de 2 GB, il est nécessaire de suivre certaines directives:

- MySQL recommande d'utiliser un système 64 bits. Les systèmes 32 bits pourraient connaître de sérieux problèmes à partir de 2038.
- Plus la mémoire vive et le processeur sont importants, meilleures sont les performances. D'après notre expérience, la RAM est plus importante que le CPU. Le minimum pour un système de niveau entreprise sera de 4 GB. Un bon choix pour un grand système est 16 GB. N'oubliez pas qu'une plus grande quantité de RAM peut accélérer les mises à jour des clés en conservant les pages clés les plus utilisées dans la RAM.
- Il est bon de pouvoir mettre le système hors service en cas de panne. Pour les systèmes où la base

de données se trouve sur un autre serveur de base de données dédié, utilisez l'Ethernet Gigabit, de préférence avec de la fibre optique plutôt que du cuivre. La latence est aussi importante que les performances.
- L'optimisation des disques est très importante pour les très grandes bases de données : les bases de données et les tables doivent être réparties sur différents disques. Dans MySQL, les liens symboliques peuvent être utilisés à cette fin. Utilisez des disques différents pour le système, la base de données et (le cas échéant) les journaux de réplication binaires.

L'utilisation de disques SSD est recommandée en raison de leur vitesse et de la latence améliorée du système.

- Si le client et le serveur MySQL se trouvent sur la même machine, utilisez des sockets au lieu de connexions TCP/IP lors de la connexion à MySQL (cela peut donner une amélioration de 7,5 %). Vous pouvez le faire sans spécifier le nom d'hôte ou localhost lors de la connexion au serveur MySQL. Désactivez la connexion binaire et la réplication si vous n'avez qu'un seul serveur hôte MySQL.
- Pandora FMS fonctionne sur MySQL et nous recommandons l'utilisation de la version modifiée de MySQL (Percona Server for MySQL), qui offre de meilleures performances. Les *plugins* programmés sont pour Percona®.

Veuillez noter que les points suivants influencent grandement les performances:

- Utilisez uniquement des journaux binaires si vous utilisez une configuration MySQL avec réplication.
- N'utilisez pas *logs tracing queries* ou *slow query logs*. Ceci n'est recommandé que dans des cas spécifiques.

Désactiver la réplication binaire

Si vous avez configuré un système Pandora FMS HA, la réplication binaire est nécessaire. Cette recommandation n'est valable que si vous disposez d'un seul serveur Pandora FMS.

Par défaut, il est activé dans la plupart des distributions GNU/Linux. Pour le désactiver, éditez le fichier `my.cnf` généralement dans `/etc/my.cnf` et commentez les lignes suivantes :

```
# log-bin=mysql-bin
# binlog_format=mixed
```

Vous devez commenter les deux lignes, puis redémarrer le serveur MySQL.

Performance d'accès au disque

Pour en savoir plus sur “Data backup and recovery in Pandora FMS”, veuillez suivre ce

lien.

Avec d'autres paramètres clés, il y en a trois qui sont particulièrement pertinents quand il s'agit de la performance d'accès au disque, qui est souvent le goulot d'étranglement en ce qui concerne MySQL.

innodb_log_file_size

```
innodb_log_file_size = 64M
```

Cette valeur est fixée par défaut et peut être supérieure (même 512M) sans préjudice, sauf pour la récupération en cas de problème et d'occupation supérieure du disque. La valeur par défaut de MySQL est de 5M, ce qui est très faible pour les environnements de production avec des volumes de transactions élevés. Pour modifier cette valeur avec un système déjà en fonctionnement:

1. D'abord faites un DUMP complet et supprimer les fichiers d'index binaires d'Innodb.
2. Effacez les fichiers d'indez binaires d'Innodb (généralement dans `/var/lib/mysql/ib*`).
3. Changez `my.cnf`, avec la valeur choisie.
4. Redémarrez MySQL.
5. Chargez le dump SQL.

Puisque le processus est le même que pour activer le jeton `innodb_file_per_table` (décrit un peu plus bas, nous recommandons de faire tout le processus simultanément):

innodb_io_capacity

```
innodb_io_capacity = 300
```

Par défaut ce paramètre a la valeur 200, mais nous devons connaître au préalable les IOPS du disque système. Vous pouvez savoir exactement en cherchant IOPS et le modèle exact du disque dur, où les valeurs recommandées sont : 7500RPM → 100 IOPS, 15000 RPM → 190 IOPS, SSD → 1500 IOPS. Plus d'informations sur ce lien.

innodb_file_per_table

Utiliser un espace de table pour chaque table:

Dans MySQL 5.0, vous pouvez stocker chaque table InnoDB et ses index dans son propre fichier. Cette fonctionnalité est appelée *multiple tablespaces* (plusieurs espaces de table) car, en fait, chaque table a son propre espace de table.

L'utilisation de plusieurs espaces de table peut être bénéfique pour les utilisateurs qui veulent déplacer des tables spécifiques vers des disques physiques séparés ou qui veulent restaurer des sauvegardes de tables sans interrompre l'utilisation des autres tables InnoDB.

Plusieurs tablespaces peuvent être activés en ajoutant cette ligne à la section `mysqld` de `my.cnf`:

```
[mysqld]
innodb_file_per_table
```

Après le redémarrage du serveur, InnoDB stockera chaque nouvelle table créée dans son propre fichier `name_table.ibd` dans le répertoire de la base de données à laquelle la table appartient. Ceci est similaire à ce que fait le moteur de stockage MyISAM, mais MyISAM divise la table en un fichier de données `tbl_name.MYD` et un fichier d'index `tbl_name.MYI`.

Pour InnoDB, les données et les index sont stockés ensemble dans le fichier `.ibd`. Le fichier `tbl_name.frm` est toujours créé comme d'habitude.

innodb_file_per_table n'affecte que la création des tables. Si vous démarrez le serveur avec cette option, les nouvelles tables seront créées à l'aide de fichiers .ibd, mais vous pouvez toujours avoir accès aux tables existantes dans l'espace de tables partagé. Si vous supprimez cette option, les nouvelles tables seront créées dans l'espace partagé, mais il sera toujours possible d'avoir accès aux tables créées dans plusieurs espaces de tables.

Evitant l'écriture à chaque transaction

Par défaut, MySQL définit `autocommit = 1` pour chaque connexion. Ce n'est pas mauvais pour MyISAM, car ce que l'on écrit n'est pas garanti sur disque, mais pour InnoDB cela signifie que chaque insert / update / delete dans une table InnoDB entraînera une écriture sur disque (flush).

Qu'est-ce qui ne va pas avec l'écriture sur disque ? Rien du tout. Ils s'assurent que toute compromission garantit que les données sont présentes lorsque la base de données est redémarrée après un crash. Le problème est que le fonctionnement de la base de données est limité par la vitesse physique du disque.

Comme le disque doit écrire les données sur un disque avant la confirmation de l'écriture, cela prend du temps. En supposant même un temps de recherche moyen de 9ms pour l'écriture sur disque, nous sommes limités à environ 67 commits/sec1, c'est très lent. Et pendant que le disque est occupé à essayer de faire écrire le secteur, il ne fait aucune lecture.

InnoDB peut éviter une partie de cette limitation en écrivant ensemble, mais la limitation existe toujours. On peut l'empêcher d'écrire à la fin de chaque transaction en le faisant mettre dans un système d'écriture “ automatique ”, qui écrit environ toutes les secondes. En cas d'échec, nous pouvons perdre les données de la dernière seconde, ce qui est plus que supportable si nous essayons de gagner en efficacité. Pour ce faire, nous utiliserons le jeton de configuration suivant : `innodb_flush_log_at_trx_commit = 0` La valeur par défaut est cette valeur dans la configuration.

Plus grande taille de KeyBuffer

En fonction de la RAM totale du système, c'est un paramètre global très important qui accélère les DELETE et INSERT.

```
key_buffer_size = 4M
```

C'est la valeur par défaut dans la configuration.

Autres paramètres importants

Il y a plusieurs tampons qui par défaut sur certaines distributions sont vides. La modification de ces paramètres peut donner de bien meilleures performances que celles par défaut. Il est important de s'assurer que ces jetons existent dans le fichier de configuration de MySQL.

```
key_buffer_size=4M
read_buffer_size=128K
read_rnd_buffer_size=128K
sort_buffer_size=128K
join_buffer_size=4M
```

Pour MySQL version 8 et versions ultérieures, l'équipe de développement de MySQL ont fini le support pour *query cache*, utilisé dans les versions précédentes de Pandora FMS; si vous souhaitez obtenir plus d'informations visitez le lien suivant: https://dev.mysql.com/blog-archive/mysql-8-0-retiring-support-for-the-query-cache/ .

Amélioration de la concurrence dans InnoDB

Il y a un paramètre qui peut affecter beaucoup la performance du serveur MySQL avec Pandora FMS. Ce paramètre est `innodb_thread_concurrency`. Ce paramètre est utilisé pour spécifier combien de "*concurrent threads*" MySQL peut exécuter.

Il s'agit d'un paramètre avancé qui ne doit être modifié que manuellement si des réglages de performance sont nécessaires sur des systèmes à forte concurrence.

Un mauvais réglage de ce paramètre peut entraîner un fonctionnement plus lent que la valeur par défaut, il est donc particulièrement important de prêter attention à plusieurs aspects:

- Version de MySQL: Dans les différentes versions de MySQL, ce paramètre se comporte très

différemment.
- Nombre de processeurs réels (physiques): à cet égard, vous pouvez accéder à l'option la documentation officielle de MySQL.

La valeur recommandée est le nombre de CPU (physiques) multiplié par 2 plus le nombre de disques où se trouve InnoDB.

La valeur de `innodb_thread_concurrency` a été modifiée dans différentes versions de MySQL, actuellement la valeur par défaut est `0`. Une valeur de `0` signifie “ouvrir autant de threads que possible”. Par conséquent, en cas de doute, elle peut être utilisée:

```
innodb_thread_concurrency = 0
```

Fragmentation de MySQL

Tout comme les systèmes de fichiers, les bases de données se fragmentent également, ce qui entraîne une perte de performance de l'ensemble du système. Dans un système à haute performance, tel que le Pandora FMS, il est essentiel que la santé de la base de données n'affecte pas le fonctionnement du système. Dans les systèmes surchargés, la base de données peut être bloquée à la fin, ce qui entraîne une chute de l'ensemble du système.

Une bonne configuration de MySQL pourrait rendre le travail de Pandora FMS plus rapide. Si vous avez des problèmes de performance, ce sera probablement parce que vous n'avez pas configuré MySQL correctement ou à cause d'un problème lié à la base de données.

Vérification du fichier my.cnf

Commencez par vérifier le fichier `my.cnf` et sa configuration de base pour MySQL. Ce fichier de configuration est écrit au format INI et son emplacement peut être déterminé avec la commande suivante:

```
mysqld --help --verbose | more
```

La configuration de votre `my.cnf` devrait être similaire à celle-ci (4 GB de RAM et en utilisant un matériel de configuration moyenne). Vérifiez que vous avez bien saisi tous ces paramètres dans la section `[mysqld]`:

```
[mysqld]
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock
user=mysql
```

```
character-set-server=utf8mb4
skip-character-set-client-handshake

max_allowed_packet = 64M
innodb_buffer_pool_size = 800M
innodb_lock_wait_timeout = 90
innodb_file_per_table
innodb_flush_log_at_trx_commit = 0
innodb_flush_method = O_DIRECT
innodb_log_file_size = 64M
innodb_log_buffer_size = 16M
innodb_io_capacity = 100
thread_cache_size = 8
thread_stack    = 256K
max_connections = 100

key_buffer_size=4M
read_buffer_size=128K
read_rnd_buffer_size=128K
sort_buffer_size=128K
join_buffer_size=4M

sql_mode=""

[mysqld_safe]
log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
```

Si vous utilisez MySQL 8 et que vous n'avez pas d'environnement HA, désactivez les journaux binaires avec l'instruction suivante dans la section `[mysqld]`:

```
skip-log-bin
```

Toute modification que vous apportez au fichier `my.cnf` devra redémarrer le service MySQL.

- Vérifiez l'état du service avec `systemctl status mysqld.service`.
- Vérifiez la fin du fichier `/var/log/mysqld.log` pour voir si des erreurs se sont produites.
- Pour plus d'informations consultez le lien suivant The Error Log dans le site web de MySQL.

Reconstruire les bases de données

Pour en savoir plus ser la “Gestion et administration des serveurs”, veuillez aller ver ce lien.

En apportant certaines modifications à `my.cnf` (par exemple, en ajoutant le paramètre

`innodb_file_per_table`), il est probable que la base de données ne fonctionnera pas lorsque le service sera redémarré. Si vous obtenez l'erreur suivante, vous devrez restaurer la configuration précédente (vous devrez utiliser les informations d'identification de l'utilisateur root ou *root user*) et sauvegarder la base de données:

```
InnoDB: Error: log file ./ib_logfile0 is of different size 0 5242880 bytes
InnoDB: than specified in the .cnf file 0 67108864 bytes!
```

1. Effectuer une *sauvegarde* de la base de données:

```
mysqldump -uroot -p pandora > /home/pandora/pandora.sql
```

2. Arrêtez le serveur MySQL et déplacez les données dans un dossier dans *backup*:

```
systemctl stop mysql
mv /var/lib/mysql /var/lib/mysql.bak
```

3. Créez un nouveau dossier pour les données MySQL (les permis correspondants seront attribués à la cinquième étape):

```
mkdir /var/lib/mysql
```

4. Initialiser le serveur MySQL en spécifiant le dossier de destination dans le paramètre `--datadir`. Ce processus génère un mot de passe temporaire qui doit être écrit (il sera affiché par la sortie standard ou stocké dans `/var/lib/mysqld.log`):

```
mysqld --initialize --datadir /var/lib/mysql
```

5. Attribuez les autorisations nécessaires au nouveau dossier:

```
chown -R mysql:mysql /var/lib/mysql
chcon -R system_u:object_r:mysqld_db_t:s0 /var/lib/mysql
```

6. Démarrez le service MySQL et connectez-vous avec le client MySQL, en utilisant le mot de passe de l'étape 4:

```
systemctl start mysql
mysql -uroot -p
```

Il arrive souvent que les systèmes MySQL/Percona ne chargent pas correctement les paramètres de configuration du fichier `my.cnf`, généralement parce que ces valeurs ont été écrites en dehors de la section `[mysqld]` .

7. Changez le mot de passe de l'utilisateur root pour celui de votre choix (ici nous utilisons

Pandor4!):

```
ALTER USER 'root'@'localhost' IDENTIFIED BY 'Pandor4!';
```

8. Quittez le client MySQL et vérifiez que vous pouvez vous connecter à nouveau en utilisant le nouveau mot de passe.

9. Connectez-vous à nouveau au client MySQL et créez la base de données:

```
CREATE DATABASE pandora;
USE pandora;
```

10. Charger la *sauvegarde* de la base de données:

```
SOURCE /home/pandora/pandora.sql
```

11. Créez les utilisateurs d'accès pour Pandora FMS en utilisant les mêmes informations d'identification que dans l'installation précédente (ici Pandor4! est utilisé) et donnez-leur des permissions sur la base de données:

```
CREATE USER 'pandora'@'localhost' IDENTIFIED WITH mysql_native_password BY
'Pandor4!';
CREATE USER 'pandora'@'127.0.0.1' IDENTIFIED WITH mysql_native_password BY
'Pandor4!';
GRANT ALL PRIVILEGES ON pandora.* TO 'pandora'@'localhost';
GRANT ALL PRIVILEGES ON pandora.* TO 'pandora'@'127.0.0.1';
```

12. Après avoir configuré le fichier `my.cnf` et redémarré le service MySQL, vous devez vérifier que ces changements ont été correctement appliqués. Pour ce faire, vous pouvez vérifier les variables une par une:

```
SHOW VARIABLES LIKE 'innodb_log_file_size';
```

```
SHOW VARIABLES LIKE 'innodb_io_capacity';
```

```
SHOW VARIABLES LIKE 'innodb_file_per_table';
```

Ou lors d'une consultation générale, par exemple:

```
SHOW VARIABLES LIKE "innodb%";
```

Une fois que vous avez vérifié que Pandora FMS (la console Web et le serveur) peut se connecter correctement à la base de données et que tout fonctionne correctement, vous pouvez supprimer le répertoire `mysql.bak`:

```
rm -rf /var/lib/mysql.bak
```

Vérifier que les fichiers de données isolés pour chaque table sont ACTIFS

```
ls -lah /var/lib/mysql/pandora/*.ibd | wc -l
```

Il devrait avoir plus de 100 fichiers (selon la version de Pandora FMS). Chaque `.ibd` sera le fichier de données pour chaque table, lorsque vous aurez activé le paramètre `innodb_file_per_table` dans le fichier `my.cnf` . Si vous n'avez aucun de ces fichiers `.ibd`, cela signifie que vous n'utilisez qu'un seul fichier pour stocker toutes les informations. Cela signifie que la fragmentation des tableaux est commune au reste des tableaux, ce qui pourrait indiquer que la performance sera pire chaque semaine qui passe.

Si vous avez votre base de données fonctionnant sous une seule base de données, vous devrez d'abord recréer la base de données après avoir correctement configuré le fichier `my.cnf` et redémarré MySQL.

Optimisation de tableaux spécifiques

Une autre solution moins *radicale* pour pallier le problème de la fragmentation est l'utilisation de l'outil OPTIMIZE de MySQL pour optimiser certaines tables FMS de Pandora. Pour cela, directement depuis MySQL, exécutez:

```
OPTIMIZE TABLE tagente_datos;
OPTIMIZE TABLE tagente;
OPTIMIZE TABLE tagente_datos_string;
OPTIMIZE TABLE tagent_access;
OPTIMIZE TABLE tagente_modulo;
OPTIMIZE TABLE tagente_estado;
```

Cela améliore les performances et il ne devrait pas être nécessaire de le lancer plus d'une fois par semaine, en pouvant le faire *hot* pendant que le système fonctionne.

Dans les très gros systèmes, l'OPTIMIZE peut se *bloquer* et ne pas être une alternative, il est donc préférable de reconstruire la base de données.

Après avoir effectué ces opérations, il est pratique de les exécuter:

```
FLUSH TABLES;
```

D'après le manuel de MySQL:

For InnoDB tables, OPTIMIZE TABLE is mapped to ALTER TABLE, which rebuilds the table to update index statistics and free unused space in the clustered index.

(Pour les tables InnoDB, OPTIMIZE TABLE est mis en correspondance avec ALTER TABLE, qui reconstruit la table pour mettre à jour les statistiques de l'index et libérer l'espace inutilisé dans l'index en grappe).

Jetons MySQL spéciaux

Il existe des jetons MySQL très *spéciaux*, qui peuvent aider ou nuire aux performances:

- `innodb_flush_method`:

```
innodb_flush_method = O_DIRECT
```

Ce paramètre affecte la façon dont il est écrit sur le disque.

- `innodb_lock_wait_timeout`:

```
innodb_lock_wait_timeout = 90
```

Il empêche qu'en cas de blocage occasionnel, MySQL *abandonne* (MySQL est parti) et s'arrête. Si elle dépasse 90 secondes, c'est un gros problème.

Vérification de la table de fragmentation par table

En utilisant le CLI de MySQL, vous devriez exécuter cette requête:

```
Select ENGINE, TABLE_NAME,Round( DATA_LENGTH/1024/1024) as data_length ,
round(INDEX_LENGTH/1024/1024)
 as index_length, round(DATA_FREE/ 1024/1024) as data_free,
(data_free/(index_length+data_length))
 as frag_ratio from information_schema.tables
 where TABLE_TYPE = 'BASE TABLE' and DATA_FREE> 0 order by frag_ratio desc;
```

Vous ne devriez voir que les tables avec un certain indice de fragmentation, par exemple:

```
+--------+------------------------+-------------+--------------+-----------+---
---------+
| ENGINE | TABLE_NAME             | data_length | index_length | data_free |
frag_ratio |
+--------+------------------------+-------------+--------------+-----------+---
---------+
| InnoDB | tserver_export_data    |           0 |            0 |         5 |
320.0000 |
| InnoDB | tagent_module_inventory |           0 |            0 |         6 |
25.6000 |
| InnoDB | tagente_datos_inventory |           4 |            0 |        40 |
```

```
9.8842 |
| InnoDB | tsesion_extended        |           1 |            0 |         4 |
3.3684 |
| InnoDB | tagent_access           |           2 |            7 |        27 |
2.9845 |
| InnoDB | tpending_mail           |           2 |            0 |         4 |
2.6392 |
| InnoDB | tagente_modulo          |           2 |            0 |         4 |
2.1333 |
| InnoDB | tgis_data_history       |          24 |           11 |        67 |
1.9075 |
| InnoDB | tsesion                 |           2 |            0 |         4 |
1.7778 |
| InnoDB | tupdate                 |           3 |            0 |         3 |
1.1852 |
| InnoDB | tagente_datos           |         186 |          194 |       399 |
1.0525 |
| InnoDB | tagente_datos_string    |          15 |            9 |        24 |
0.9981 |
| InnoDB | tevento                 |         149 |           62 |        46 |
0.2183 |
| InnoDB | tagente_datos           |        2810 |         2509 |        65 |
0.0122 |
| InnoDB | tagente_datos_string    |         317 |          122 |         5 |
0.0114 |
+--------+-------------------------+-------------+--------------+-----------+---
---------+
```

Cette requête ne fonctionnera que sur les tables ayant une fragmentation supérieure à 10 %. Les tables qui sont trop grandes (comme `tagent_donnees`) pourraient prendre trop de temps à optimiser si elles sont trop fragmentées. Cela pourrait avoir un certain impact sur les systèmes de production.

La prudence est de mise lorsqu'il s'agit d'optimiser des tables aussi volumineuses. Un environnement normal peut être optimisé une fois par an et les environnements plus importants tous les six mois.

Pour optimiser la table `tagent_module_inventory`:

```
optimize table  tagent_module_inventory;
```

Un message apparaîtra pour vous le dire:

```
"Table does not support optimize, doing recreate + analyze instead".
```

Maintenant, si vous vérifiez à nouveau la fragmentation des tableaux, vous pouvez voir que la fragmentation a disparu:

```
+--------+-------------------------+-------------+--------------+-----------+---
---------+
| ENGINE | TABLE_NAME              | data_length | index_length | data_free |
frag_ratio |
+--------+-------------------------+-------------+--------------+-----------+---
---------+
| InnoDB | tserver_export_data     |           0 |            0 |         5 |
320.0000 |
| InnoDB | tagente_datos_inventory |           4 |            0 |        40 |
9.8842 |
| InnoDB | tsesion_extended        |           1 |            0 |         4 |
3.3684 |
| InnoDB | tagent_access           |           2 |            7 |        27 |
2.9845 |
| InnoDB | tpending_mail           |           2 |            0 |         4 |
2.6392 |
| InnoDB | tagente_modulo          |           2 |            0 |         4 |
2.1333 |
| InnoDB | tgis_data_history       |          24 |           11 |        67 |
1.9075 |
| InnoDB | tsesion                 |           2 |            0 |         4 |
1.7778 |
| InnoDB | tupdate                 |           3 |            0 |         3 |
1.1852 |
| InnoDB | tagente_datos           |         186 |          194 |       399 |
1.0525 |
| InnoDB | tagente_datos_string    |          15 |            9 |        24 |
0.9981 |
| InnoDB | tevento                 |         149 |           62 |        46 |
0.2183 |
| InnoDB | tagente_datos           |        2810 |         2509 |        65 |
0.0122 |
| InnoDB | tagente_datos_string    |         317 |          122 |         5 |
0.0114 |
+--------+-------------------------+-------------+--------------+-----------+---
---------+
```

Pour pouvoir effectuer cette optimisation, il sera nécessaire de disposer de l'espace nécessaire sur le disque dur pour effectuer l'opération. Sinon, une erreur apparaîtra et l'opération ne sera pas effectuée.

Charge du système

Ceci est d'ordre général, mais nous devons nous assurer que le système d'E/S n'est pas un goulot d'étranglement (disques).Exécutez la commande suivante pour collecter les informations sur le système:

```
vmstat 1 10
```

Maintenant, regardez les dernières colonnes (CPU-WA), une valeur supérieure à 10 signifie que votre disque d'E/S a un problème qui doit être corrigé.

Il est normal d'avoir quelques valeurs de CPU-US plus grands que zéro, sinon cela signifierait que votre système utilise la mémoire SWAP, ce qui est considéré comme un tueur de performance. Vous devrez augmenter la mémoire RAM ou diminuer la mémoire RAM utilisée dans vos applications (threads Pandora FMS, buffers MySQL, etc).

Nous vous montrons l'exemple de sortie d'un système “ normal ”:

```
procs -----------memory---------- ---swap-- -----io---- --system-- -----cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 0  0  46248  78664 154644 576800    0    0     2   147    0    9  7 10 83  0  0
 0  0  46248  78656 154644 576808    0    0     0     0   49   37  0  0 100  0
0
 2  0  46248  78904 154648 576740    0    0     0   184  728 2484 63  6 31  0  0
 0  0  46248  79028 154648 576736    0    0    16   616  363  979 21  0 79  0  0
 1  0  46248  79028 154648 576736    0    0     0    20   35   37  0  1 98  1  0
 0  0  46248  79028 154648 576736    0    0     0     0   28   22  0  0 100  0
0
 1  0  46248  79028 154648 576736    0    0     0  3852  141  303  0  0 98  2  0
 2  0  46248  78904 154660 576660    0    0     0   188  642 2354 56  4 40  0  0
 1  0  46248  78904 154660 576680    0    0     0    88  190  634 13  0 86  1  0
 1  0  46248  78904 154660 576680    0    0     0    16   35   40  0  0 100  0
0
 1  0  46248  78904 154660 576680    0    0     0     0   26   21  0  0 100  0
0
 0  0  46248  78904 154660 576680    0    0     0     0   27   27  0  0 100  0
0
 1  0  46248  78904 154724 576616    0    0   112   192  608 2214 52  4 44  0  0
 0  0  46248  78904 154724 576616    0    0     0    76  236  771 16  0 84  0  0
 0  0  46248  78904 154724 576616    0    0     0    20   38   38  0  0 100  0
0
 0  0  46248  78904 154724 576616    0    0     0     0   31   21  0  0 100  0
0
 0  0  46248  78904 154740 576608    0    0     0  3192  187  322  1  0 96  3  0
 1  0  46248  79028 154756 576544    0    0    16   192  632 2087 53  5 42  0  0
 0  0  46248  79028 154760 576568    0    0     0    56  255  927 19  2 79  0  0
 0  0  46248  79028 154768 576564    0    0     0    20   33   44  0  0 100  0
0
```

Partitionnement des tables MySQL

Pour utiliser le partitionnement des tables MySQL, vous devriez également utiliser le système de “multiples tablespaces” décrit ci-dessus (`innodb_file_per_table`).

MySQL supporte le partitionnement des tables, ce qui vous permet de distribuer de très grandes tables en plus petits morceaux, comme des subdivisions logiques (vous pouvez consulter le manuel de MySQL pour plus de détails).

Si vous avez de grandes quantités de données dans votre base de données Pandora FMS et que vous pensez que beaucoup d'opérations de la console qui se réfèrent à ces données (par exemple drawing graph) sont assez lentes, vous améliorerez vos performances en utilisant le partitionnement des tables.

Partitionnement automatique

Pandora FMS effectue automatiquement un partitionnement mensuel dans la base de données historique s'il est configuré à partir de la console Web, section Base de données historique, de la configuration générale. Pour plus de détails, voir ce lien.

Partitionnement manuel

Après avoir installé Pandora FMS et activé sa base de données historique, c'est celle qui contiendra la plus grande quantité de données et c'est celle qui est recommandée pour effectuer un partitionnement des tables. Précisément les tables indiquées pour cela sont `tagente_datos`, `tagente_datos_string`, `tagente_datos_inc` et `tevento`. Le processus pratique manuel pour la table `tagente_datos` est décrit ci-dessous (voir aussi "Partitionnement automatique").

Il faut d'abord vérifier que le répertoire `/var/lib/mysql/pandora_history/*.ibd` contient bien plusieurs fichiers (un par table). Si ce n'est pas le cas, vous devrez faire un *dump* de la base de données, changer la configuration du fichier `my.cnf`, redémarrer MySQL, supprimer la base de données actuelle et la recréer à partir du *dump*.

Une fois que vous vous êtes assuré que `innodb_file_per_table` est activé, séparez les deux bases de données principales en différentes partitions.

- Cette opération nécessite un espace disque suffisant pour être menée à bien. Vous devrez vérifier la taille du fichier `tagente_datos.ibd`. S'il occupe par exemple 10 gigaoctets, vous aurez besoin de 15 gigaoctets d'espace libre pour lancer l'opération.
- Cette opération peut prendre beaucoup de temps, en fonction de la taille de la table. Par exemple, il faudrait une heure et demie pour diviser une table contenant environ 7500 *modules data* pour 100 jours (plus de 50 000 000 lignes).

Il s'agit d'un exemple de partitionnement de l'ensemble de l'année 2023 jusqu'à présent et pour les mois à venir. Pour commencer le processus, vous devez exécuter la requête suivante dans l'interface de commande MySQL:

```
ALTER TABLE tagente_datos PARTITION BY RANGE (utimestamp) (
PARTITION Jan23 VALUES LESS THAN (UNIX_TIMESTAMP('2023-02-01 00:00:00')),
PARTITION Feb23 VALUES LESS THAN (UNIX_TIMESTAMP('2023-03-01 00:00:00')),
PARTITION Mar23 VALUES LESS THAN (UNIX_TIMESTAMP('2023-04-01 00:00:00')),
PARTITION Apr23 VALUES LESS THAN (UNIX_TIMESTAMP('2023-05-01 00:00:00')),
PARTITION May23 VALUES LESS THAN (UNIX_TIMESTAMP('2023-06-01 00:00:00')),
PARTITION Jun23 VALUES LESS THAN (UNIX_TIMESTAMP('2023-07-01 00:00:00')),
PARTITION Jul23 VALUES LESS THAN (UNIX_TIMESTAMP('2023-08-01 00:00:00')),
PARTITION Aug23 VALUES LESS THAN (UNIX_TIMESTAMP('2023-09-01 00:00:00')),
PARTITION Sep23 VALUES LESS THAN (UNIX_TIMESTAMP('2023-10-01 00:00:00')),
PARTITION Oct23 VALUES LESS THAN (UNIX_TIMESTAMP('2023-11-01 00:00:00')),
PARTITION Nov23 VALUES LESS THAN (UNIX_TIMESTAMP('2023-12-01 00:00:00')),
PARTITION Dec23 VALUES LESS THAN (UNIX_TIMESTAMP('2024-01-01 00:00:00')),
PARTITION pActual VALUES LESS THAN (MAXVALUE)
);
```

La requête suivante devrait alors être exécutée chaque mois pour réorganiser le partitionnement:

```
ALTER TABLE tagente_datos REORGANIZE PARTITION pActual INTO (
PARTITION Jan24 VALUES LESS THAN (UNIX_TIMESTAMP('2024-02-01 00:00:00')),
PARTITION pActual VALUES LESS THAN MAXVALUE);
```

Remplacer “Jan24” par le mois dans lequel vous vous trouvez.

Rappelez-vous que cette opération peut prendre des heures, selon la taille de la table `tagente_datos`. Vous pouvez vérifier le processus en observant la taille des fichiers de partitionnement en exécutant:

```
[root@histdb pandora_history]# ls -lah | grep "#sql"

 -rw-rw----  1 mysql mysql 424M feb 24 05:58 #sql-76b4_3f7c#P#Jan23.ibd
 -rw-rw----  1 mysql mysql 420M feb 24 05:51 #sql-76b4_3f7c#P#Feb23.ibd
 -rw-rw----  1 mysql mysql 128K feb 24 05:40 #sql-76b4_3f7c#P#Mar23.ibd
 -rw-rw----  1 mysql mysql 840M feb 24 05:44 #sql-76b4_3f7c#P#Apr23.ibd
 -rw-rw----  1 mysql mysql 440M feb 24 05:47 #sql-76b4_3f7c#P#May23.ibd
 -rw-rw----  1 mysql mysql  10M feb 24 05:42 #sql-76b4_3f7c#P#Jun23.ibd
 -rw-rw----  1 mysql mysql 404M feb 24 05:56 #sql-76b4_3f7c#P#Jul23.ibd
 -rw-rw----  1 mysql mysql 436M feb 24 05:54 #sql-76b4_3f7c#P#Aug23.ibd
 -rw-rw----  1 mysql mysql 400M feb 24 05:49 #sql-76b4_3f7c#P#Sep23.ibd
 -rw-rw----  1 mysql mysql 408M feb 24 05:52 #sql-76b4_3f7c#P#Oct23.ibd
 -rw-rw----  1 mysql mysql  72M feb 24 06:03 #sql-76b4_3f7c#P#Nov23.ibd
 -rw-rw----  1 mysql mysql 404M feb 24 06:03 #sql-76b4_3f7c#P#Dec23.ibd
 -rw-rw----  1 mysql mysql 416M feb 24 06:00 #sql-76b4_3f7c#P#jan23.ibd
```

Reconstruire la base de données

Pour en savoir plus sur la “ Sauvegarde et restauration des données sur Pandora FMS ”,

allez vers ce lien.

Reconstruction partielle

Le système de gestion de base de données de MySQL, comme d'autres moteurs SQL tels qu'Oracle®, se dégrade avec le temps en raison de causes telles que la fragmentation des données produite par la suppression et l'insertion continue dans de grandes tables. Dans les grands environnements avec un volume de trafic élevé, il existe un moyen très simple d'améliorer les performances et d'empêcher leur dégradation: reconstruire périodiquement la base de données.

Pour ce faire, un arrêt de service doit être programmé, ce qui peut prendre environ 1 heure.

Dans cet arrêt de service, ce que vous devez faire est d'arrêter la console Pandora FMS WEB, et le serveur Attention: vous pouvez laisser le serveur Tentacle en marche pour continuer à recevoir des données, et ces données seront traitées dès que le serveur sera à nouveau opérationnel.

Une fois arrêté, faites un dump de la base de données (Export) ; dans cet exemple la base de données s'appelle `pandora3` et l'utilisateur doit être root:

```
mysqldump -u root -p pandora3> /tmp/pandora3.sql
Entrez le mot de passe :
```

On efface la base de données:

```
mysql -u root -p
Enter password:
```

```
mysql> drop database pandora3;
Query OK, 87 rows affected (1 min 34.37 sec)
```

Après créez la base de données et importez les donnés depuis l'exportation précédente:

```
mysql> create database pandora3;
Query OK, 1 row affected (0.01 sec)
mysql> use pandora3;
mysql> source /tmp/pandora3.sql
```

Cette opération peut prendre quelques secondes ou plusieurs minutes, en fonction de la taille de la base de données et des ressources disponibles sur la machine.

Vous pouvez automatiser ce processus, mais en raison de sa nature délicate, il est préférable de le faire manuellement.

Reconstruction total

Ce chapitre ne concerne que les bases de données d'InnoDB. Pandora FMS est construit sur les bases de données Innodb.

Malheureusement, MySQL se dégrade avec le temps, ce qui affecte les performances de l'ensemble du système. Il n'y a pas d'autre solution qui ne passe par la reconstruction de tous les schémas de base de données à partir de 0, en reconstruisant le fichier de données binaires que MySQL utilise pour stocker toutes les informations et les fichiers utilisés pour reconstruire les transactions.

Si vous regardez le répertoire `/var/lib/mysql` vous pouvez voir qu'il y a trois fichiers, qui sont toujours appelés les mêmes et qui sont, selon la gravité du cas, des géants. Par exemple :

```
-rw-rw----  1 mysql mysql 4.8G 2012-01-12 14:00 ibdata1
-rw-rw----  1 mysql mysql 5.0M 2012-01-12 14:00 ib_logfile0
-rw-rw----  1 mysql mysql 5.0M 2012-01-12 14:00 ib_logfile1
```

Le fichier `ibdata1` est celui qui contient toutes les données Innodb du système. Dans un système très fragmenté, cela prend beaucoup de temps sans " refaire " ou " réinstaller " ce système sera gros et peu efficace. Le paramètre innodb_file_per_table dont nous avons parlé précédemment, régule une partie de ce comportement.

De plus, chaque base de données possède dans le répertoire `/var/lib/mysql` un répertoire pour définir sa structure. Vous devrez les supprimer aussi.

La procédure est simple :

- Dump (via mysqldump) tous les schémas et données sur le disque :

```
mysqldump -u root -p -A> all.sql
```

- Arrêtez MySQL.
- Supprimer les répertoires de la base de données InnDB et `ibdata1`, `ib_logfile0`, `ib_logfile1`
- Démarrer MySQL.
- Créer à nouveau une base de données pandora (`create database pandora;`)
- Importer le fichier de sauvegarde (`all.sql`)

```
mysql -u root -p
mysql> source all.sql;
```

Le système devrait fonctionner ostensiblement plus vite maintenant.

Indices facultatifs

Dans certaines situations, il est possible d'optimiser le fonctionnement de MySQL au détriment des ressources du système.

Cet indice, sert à optimiser la vitesse d'obtention des graphiques au prix d'une plus grande utilisation du disque et d'une légère diminution des performances des suppressions / insertions dans les tableaux de données :

```
ALTER TABLE `pandora`.`tagent_donnees`  ADD  INDEX  `id_agent_module_utimestamp`
(  `id_agent_module`  , `utimestamp`  );
```

Actuellement dans les tables plus lourdes de Pandora FMS dans MySQL vient cette optimisation par défaut. Il est conseillé de faire appel à un expert pour optimiser les tables MySQL.

Requêtes lentes ou "slow queries"

Dans certains systèmes, selon le type d'information dont nous disposons, nous pouvons trouver des “requêtes lentes” qui aggravent le système. Nous pouvons activer le journal de ce type de requêtes pendant une courte période de temps (car cela nuit aux performances du système) afin d'étudier les requêtes pour essayer d'optimiser les tables avec des index.

Pour activer ce système, nous devons procéder comme suit:

- Éditez le fichier my.cnf et ajoutez les lignes suivantes:

```
slow_query_log=1
long_query_time=2
slow_query_log_file=/var/log/mysql_slow.log
```

- Afin de l'utiliser créez-le et établiez les régulations administratives:

```
touch /var/log/mysql_slow.log
chown mysql:mysql /var/log/mysql_slow.log
chmod 640 /var/log/mysql_slow.log
```

- Redémarrer MySQL.
- Lorsque vous avez terminé l'analyse des requêtes lentes, n'oubliez pas de réinitialiser le fichier `my.cnf` en commentant les lignes ajoutées et en redémarrant le service MySQL.

Références

- http://dev.mysql.com/tech-resources/presentations/presentation-oscon2000-20000719/index.html
- http://jeremy.zawodny.com/mysql/mysql-optimization.html

Dimensionnement du Pandora FMS pour la haute capacité

Cette section décrit différentes méthodes pour configurer Pandora FMS dans un environnement à haute capacité. Il décrit également différents outils pour effectuer des tests de charge, utiles pour ajuster l'environnement à la plus grande capacité de traitement possible.

Pandora FMS a été configuré pour supporter une charge d'environ 2500 agents dans des systèmes où la base de données, la console et le serveur se trouvent sur la même machine. Le nombre maximum recommandé est d'environ 2500 agents (environ 60000 modules) par système, mais ce nombre varie beaucoup selon qu'il s'agit d'agents XML, de modules distants, avec des intervalles élevés ou faibles, ou avec des systèmes ayant beaucoup de capacité ou peu de mémoire.

Tous ces facteurs modifient grandement le nombre d'agents qu'un système peut gérer efficacement. Lors de tests en laboratoire, il a été possible d'exécuter 10000 agents dans un seul serveur avec un matériel de base, mais fortement optimisé.

Exemple de configuration d'un serveur de grande capacité

En supposant qu'une machine RHEL 8 avec 16 Go de RAM et 8 CPU soit optimisée pour la capacité de traitement maximale du serveur de données (XML).

my.cnf

Seuls les paramètres les plus significatifs sont affichés.

```
[mysqld]
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock
user=mysql
character-set-server=utf8mb4
skip-character-set-client-handshake
# Disabling symbolic-links is recommended to prevent assorted    security risks
symbolic-links=0
# MySQL optimizations for Pandora FMS
```

```
# Please check the documentation in http://pandorafms.com for  better results
max_allowed_packet = 64M
innodb_buffer_pool_size = 6400M
innodb_lock_wait_timeout = 90
innodb_file_per_table
innodb_flush_log_at_trx_commit = 0
innodb_flush_method = O_DIRECT
innodb_log_file_size = 64M
innodb_log_buffer_size = 16M
innodb_io_capacity = 300
thread_cache_size = 8
thread_stack    = 256K
max_connections = 100
key_buffer_size=4M
read_buffer_size=128K
read_rnd_buffer_size=128K
sort_buffer_size=128K
join_buffer_size=4M
sql_mode=""

[mysqld_safe]
log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
```

pandora_server.conf

Seuls les paramètres les plus importants sont indiqués.

```
verbose 3
server_threshold 5
xxxxserver_threads 8
max_queue_files 5000
```

Aspects à prendre en compte:

- Le nombre établi dans le paramètre `verbose` se réfère à la quantité d'informations qui sont écrites dans les *logs*, étant recommandé de ne pas dépasser 3. Plus le nombre est élevé, plus les performances de Pandora FMS diminuent en raison de la grande quantité d'informations à écrire dans les *logs*.
- Une valeur élevée (15) du paramètre `server_threshold` fait que la DB souffre moins, tandis que l'augmentation du nombre maximum de fichiers traités fait que le serveur *recherche des fichiers* et remplit les *tampons*. Ces deux éléments de la configuration sont étroitement liés. Dans le cas d'une optimisation du *serveur réseau*/ il peut être intéressant d'abaisser le paramètre `server_threshold` à 5 ou 10.
- Le nombre très élevé de threads (plus de 5) défini dans xxxxserver_threads ne profite qu'aux processus avec de longues attentes d'E/S, comme *network server* ou *plugin server*. Dans le cas de *dataserver*, qui est en processus tout le temps, définir un trop grand nombre de threads peut même

nuire aux performances. Sur les systèmes dotés d'une base de données lente, essayez différentes combinaisons entre 1 et 10 ; avec des disques plus rapides et des processeurs multicœurs, ce nombre peut être augmenté. Dans le cas de l'optimisation du système pour le *serveur de réseau*, le nombre peut être beaucoup plus élevé, entre 10 et 30.

Outils d'analyse de la capacité (Capacity)

Pandora FMS dispose de plusieurs outils qui vous aideront à dimensionner correctement votre matériel et vos logiciels en fonction du volume de données que vous vous attendez à obtenir. L'un d'eux est utile pour “attaquer” directement la base de données avec des données fictives (dbstress) et l'autre génère des fichiers XML fictifs (xml_stress)

Pandora FMS XML Stress

Il s'agit d'un petit script qui génère des fichiers de données XML comme ceux envoyés par les agents Pandora FMS. Par défaut, il est situé dans:

```
/usr/share/pandora_server/util/pandora_xml_stress.pl
```

Les scripts lisent les noms des agents à partir d'un fichier texte et génèrent des fichiers de données XML pour chaque agent en fonction du fichier de configuration, où les modules sont définis comme des modèles.

Les modules sont remplis avec des données aléatoires. Il est possible de spécifier la valeur initiale et la probabilité de changement des données d'un module.

Exécutez le script de cette façon:

```
./pandora_xml_stress.pl <configuration file>
```

Exemple de configuration d’un fichier:

```
# Maximum number of threads, by default 10.
max_threads 10

# File containing a list of agent names (one per line).
agent_file agent_names.txt

# Directory where XML data files will be placed, by default /tmp.
temporal /var/spool/pandora/data_in

# Pandora FMS XML Stress log file, logs to stdout by default.
log_file pandora_xml_stress.log
```

```
# XML version, by default 1.0.
xml_version 1.0

# XML encoding, by default ISO-8859-1.
encoding ISO-8859-1

# Operating system (shared by all agents), by default Linux.
os_name Linux

# Operating system version (shared by all agents), by default 2.6.
os_version 2.6

# Agent interval, by default 300.
agent_interval 300

# Data file generation start date, by default now.
time_from 2009-06-01 00:00:00

# Data file generation end date, by default now.
time_to 2009-06-05 00:00:00

# Delay after generating the first data file for each agent to avoid
# race conditions when auto-creating the agent, by default 2.
startup_delay 2

# Address of the Tentacle server where XML files will be sent (optional).
# server_ip 192.168.50.1

# Port of the Tentacle server, by default 41121.
# server_port 41121

# Module definitions. Similar to pandora_agent.conf.

module_begin
module_name Module 1
module_type generic_data
module_description A long description.
module_max 100
module_min 10
module_exec type=RANDOM;variation=60;min=20;max=80
module_end

module_begin
module_name Module 2
module_type generic_data
module_description A long description.
module_max 80
module_min 20
module_exec type=SCATTER;prob=1;avg=40;min=0;max=80
module_end

module_begin
```

```
module_name Module 3
module_type generic_data
module_description A long description.
module_max 80
module_min 20
module_exec type=CURVE;min=20;max=80;time_wave_length=3600;time_offset=0
module_end

module_begin
module_name Module 4
module_type generic_data_string
module_description A long description.
module_max 100
module_min 10
module_exec type=RANDOM;variation=60;min=20;max=80
module_end

module_begin
module_name Module_5
module_type generic_proc
module_descripcion Module 3 description.
# Initial data.
module_data 1
module_end
```

Envoyer et recevoir la configuration locale de l'agent

En activant dans votre `pandora_xml_stress.conf` la valeur de configuration `get_and_send_agent_conf` à 1, vous pouvez faire en sorte que les agents de test de charge se comportent comme des agents normaux, puisqu'ils envoient leur fichier de configuration et le md5.

Depuis Pandora FMS Console web, vous pouvez modifier la configuration distante de sorte que lors des exécutions successives du `pandora_xml_stress`, il utilise la configuration personnalisée de Pandora FMS Console web au lieu de la définition de `pandora_xml_stress.conf`.

À part cela, vous pouvez configurer où sauvegarder localement les `.conf` de vos agents de test avec le jeton de configuration `directory_confs` dans le fichier `pandora_xml_stress.conf`.

Fichier de configuration

- max_threads Nombre de threads dans lesquels le script va s'exécuter, cela améliore les E/S.
- agent_file Chemin du fichier de la liste des noms, séparé par une nouvelle ligne.
- temporal Chemin du répertoire où les fichiers de données XML factices seront générés.
- log_file Chemin du fichier de log où le script informera de son exécution.
- xml_version Version des fichiers de données XML (par défaut 1.0).
- encoding Encodage des fichiers de données XML (par défaut ISO-8859-1).
- os_name Nom du système d'exploitation des agents factices (Linux par défaut).
- os_version Version du système d'exploitation des agents factices (par défaut 2.6).
- agent_interval: Intervalle des agents factices en secondes (par défaut 300).

- time_from Date de début pour générer les fichiers de données XML fictifs, au format `yyyy-MM-dd HH:mm:ss`.
- time_to Date de fin pour générer les fichiers de données XML fictifs, au format `yyyy-MM-dd HH:mm:ss`.
- get_and_send_agent_conf Valeur booléenne 0 ou 1, quand elle est active les agents factices vont essayer de télécharger par configuration à distance une version plus actuelle du fichier de configuration standard d'un agent. Et depuis la console d'entreprise de Pandora FMS, vous pouvez les éditer.
- startup_delay Valeur numérique du temps en secondes avant que chaque agent ne commence à générer les fichiers, elle est utilisée pour éviter les conditions de course.
- timezone_offset Valeur numérique de l'offset du fuseau horaire.
- timezone_offset_range Valeur numérique qui est utilisée pour générer dans cette plage les fuseaux horaires de manière aléatoire.
- latitude_base Valeur numérique. Il s'agit de la zone géographique d'latitude à utiliser pour définir les agents fictifs.
- longitude_base Valeur numérique. Il s'agit de la zone géographique d'longitude à utiliser pour définir les agents fictifs.
- altitude_base Valeur numérique. Il s'agit de la zone géographique d'altitude à utiliser pour définir les agents fictifs.
- position_radius Valeur numérique, est la plage autour de la circonférence avec ce rayon que l'agent factice apparaît de façon aléatoire.

Définition des modules

La définition d'un module dans le fichier de configuration du script et si vous avez activé la configuration à distance sera également la même:

```
module_begin
module_name <nom_du_module>
module_type <type_de_donnees_du_module>
module_description <description>
module_exec
type=<type_generation_xml_stress>;<autres_options_separees_par_des_points_virgules>
module_unit <unites>
module_min_critical <value>
module_max_critical <value>
module_min_warning <value>
module_max_warning <value>
module_end
```

Et vous pouvez chacun le configurer comme:

- tipo_generación_xml_stress: vous pouvez prendre les valeurs RANDOM , SCATTER , CURVE.
- module_attenuation <value>: La valeur générée est multipliée par la valeur donnée, généralement entre 0.1 et 0.9.
- module_attenuation_wdays <value> <value> ... <value>: La valeur du module n'est atténuée que les jours donnés, qui vont du dimanche (0) au samedi (6). Par exemple, le module suivant simule une diminution de 50 % du trafic réseau les samedis et dimanches:

```
```

```
module_begin
module_name Network Traffic
module_type generic_data
module_description Incoming network traffic (Kbit/s)
module_exec type=RANDOM;variation=50;min=0;max=1000000
module_unit Kbit/s
module_min_critical 900000
module_attenuation 0.5
module_attenuation_wdays 0 6
module_end
```

- module_incremental <value>: Si elle est configurée à 1, la valeur précédente du module est toujours ajoutée à une nouvelle valeur, ce qui donne une fonction croissante.
- autres : voir ci-dessous les options dont il dispose, en fonction du type de génération de module.

Aléatoire (RANDOM)

Ils ont les options suivantes :

- Probabilité de *variation* en % qu'elle varie par rapport à la valeur précédente.
- min valeur minimum que la valeur peut avoir.
- max valeur maximale que la valeur peut avoir.

Numérique

Génération de valeurs numériques aléatoires entre les valeurs min et max.

Booléens

Valeurs générées entre 0 et 1.

Chaîne

Une chaîne de longueur est générée entre la valeur min et la valeur max, les caractères sont aléatoires entre A et Z y compris les majuscules et minuscules et les chiffres.

Source externe de donnéese (SOURCE)

Vous permet de choisir un fichier de texte brut comme source de données. Options :

- src : fichier source des données.

Le fichier contient une donnée par ligne, il n'y a pas de limite de ligne. Par exemple :

```
4
5
6
10
```

Il supporte tous les types de valeurs (chaînes numériques et textuelles). Ce type de modules va utiliser chacune des données du fichier pour générer les données des modules dans Pandora, les données sont prises de manière séquentielle. Par exemple, les données du module précédent seraient vues dans Pandora comme ceci :

```
4 5 6 10 4 5 6 10 4 5 6 10 4 5 6 10 4 5 6 10 4 5 6 10
```

Dispersion (SCATTER)

Elle ne s'applique qu'aux données numériques, et les graphiques générés sont semblables à ceux d'un battement de coeur, c'est-à-dire une valeur normale et occasionnellement un “battement de coeur”.

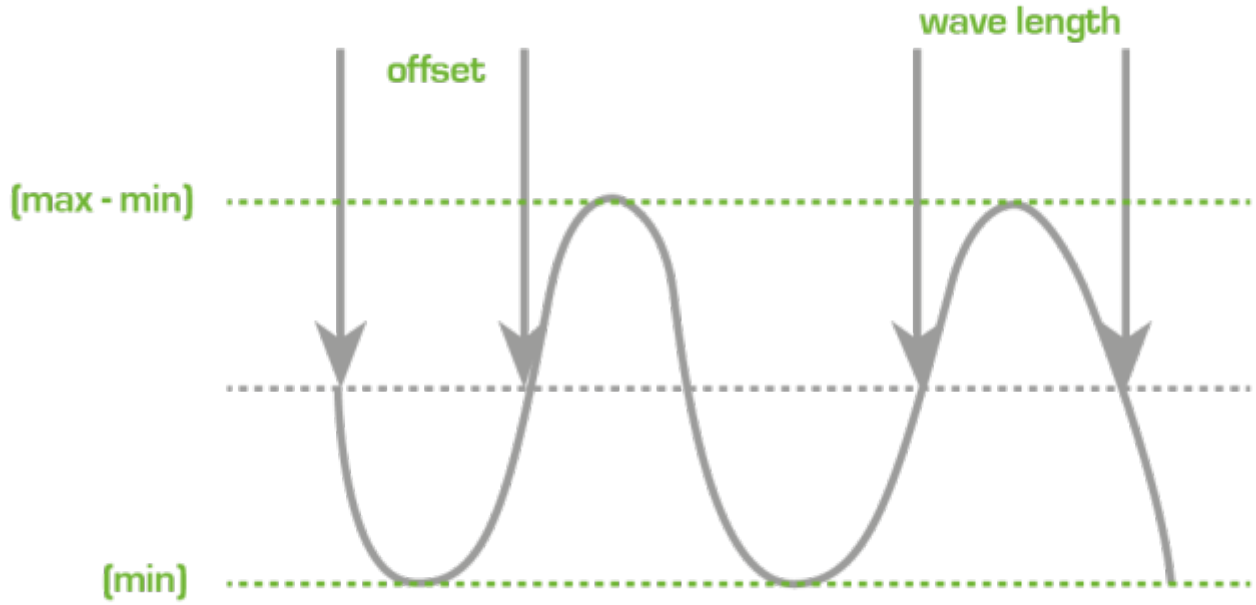
Et vous avez les options suivantes :

- min valeur minimale que la valeur peut avoir.
- max valeur maximale que la valeur peut avoir.
- Prob probabilité en % qu'il génère un *battement de coeur*.
- avg Valeur moyenne qui doit être affichée par défaut s'il n'y a pas de “*battement de coeur*”.

Courbe (CURVE)

Il génère des données de module suivant une courbe trigonométrique. Ils ont les options suivantes :

- min valeur minimale que la valeur peut avoir.
- max valeur maximale que la valeur peut avoir.
- Valeur numérique en secondes de la durée du *pic* de l'onde.
- time_offset valeur numérique en secondes du début de l'onde à partir du temps zéro avec valeur zéro du module (similaire au graphique sinusoïdal).



Notes d'intérêt :

- Cet outil est préconfiguré pour chercher, dans tous les agents, les modules de nom “ random ”, “ curve ” ou “boolean ”, et qu'utilisent une intervalle d'entre 300 secondes et 30 jours.

Comment mesurer la capacité de traitement du serveur de données

Il y a un petit script appelé `pandora_count.sh` qui se trouve dans le répertoire `/usr/share/pandora_server/util/` du serveur FMS Pandora. Ce script est utilisé pour mesurer le taux de traitement des fichiers XML par le serveur de données, et utilise comme référence le total des fichiers en attente de traitement dans `/var/spool/pandora/data_in` ; pour pouvoir l'utiliser, il faut avoir des fichiers en attente de traitement pour plusieurs milliers de paquets (ou pour les générer avec l'outil mentionné précédemment). Une fois en exécution, vous pouvez l'arrêter en appuyant sur CTRL+C.

Ce script calcule simplement le taux de traitement des fichiers du serveur. Il s'agit d'une mesure quelque peu grossière, mais utile pour se faire une idée de l'efficacité des paramètres de configuration du serveur.

Pandora FMS DB Stress

C'est un petit outil pour tester la performance de votre base de données. Il peut également être utilisé pour générer des données périodiques ou aléatoires (à l'aide de fonctions trigonométriques) et remplir des modules fictifs.

Vous devez créer un agent et lui affecter des modules pour l'injection automatique de données avec cet outil. Les noms doivent être appelés avec la notation suivante:

- *random*: pour générer des données aléatoires.
- *curve*: pour générer une courbe de coïncidences à l'aide de fonctions trigonométriques. Utile pour voir le travail d'interpolation avec différents intervalles, etc.
- *boolean*: pour générer des données booléennes aléatoires. Vous pouvez donc utiliser n'importe quel nom contenant les mots aléatoire « *random* », « *curve* » et/ou « *boolean* », par exemple:

- random_1
- courbe_autre

Vous pouvez seulement choisir le type de module " *data_server* ".

Réglage fin de l'outil *Pandora FMS DB Stress*

Cet outil est préconfiguré pour rechercher, dans tous les agents, les modules nommés « *random* », « *curve* » ou « *boolean* », et qui utilisent un intervalle entre 300 secondes et 30 jours.

Si vous voulez modifier ce comportement, vous devez éditer le *script* `pandora_dbstress` et modifier certaines variables au début du fichier:

```
# Configure here target (AGENT_ID for Stress)
my $target_module = -1; # -1 for all modules of that agent
my $target_agent = -1;
my $target_interval = 300;
my $target_days = 30;
```

- La première ligne de la variable correspondant à `target_module`, doit être définie pour un module fixe ou `-1` pour traiter toutes les cibles correspondantes.
- La deuxième ligne de la variable correspond à `target_agent`, pour un agent spécifique.
- La troisième ligne correspond à `target_interval`, défini en secondes et qui représente l'intervalle de périodicité prédéterminée du module.
- La quatrième ligne est `target_days` et représente le nombre de jours passés depuis la date, dans *timestamp*, actuelle.

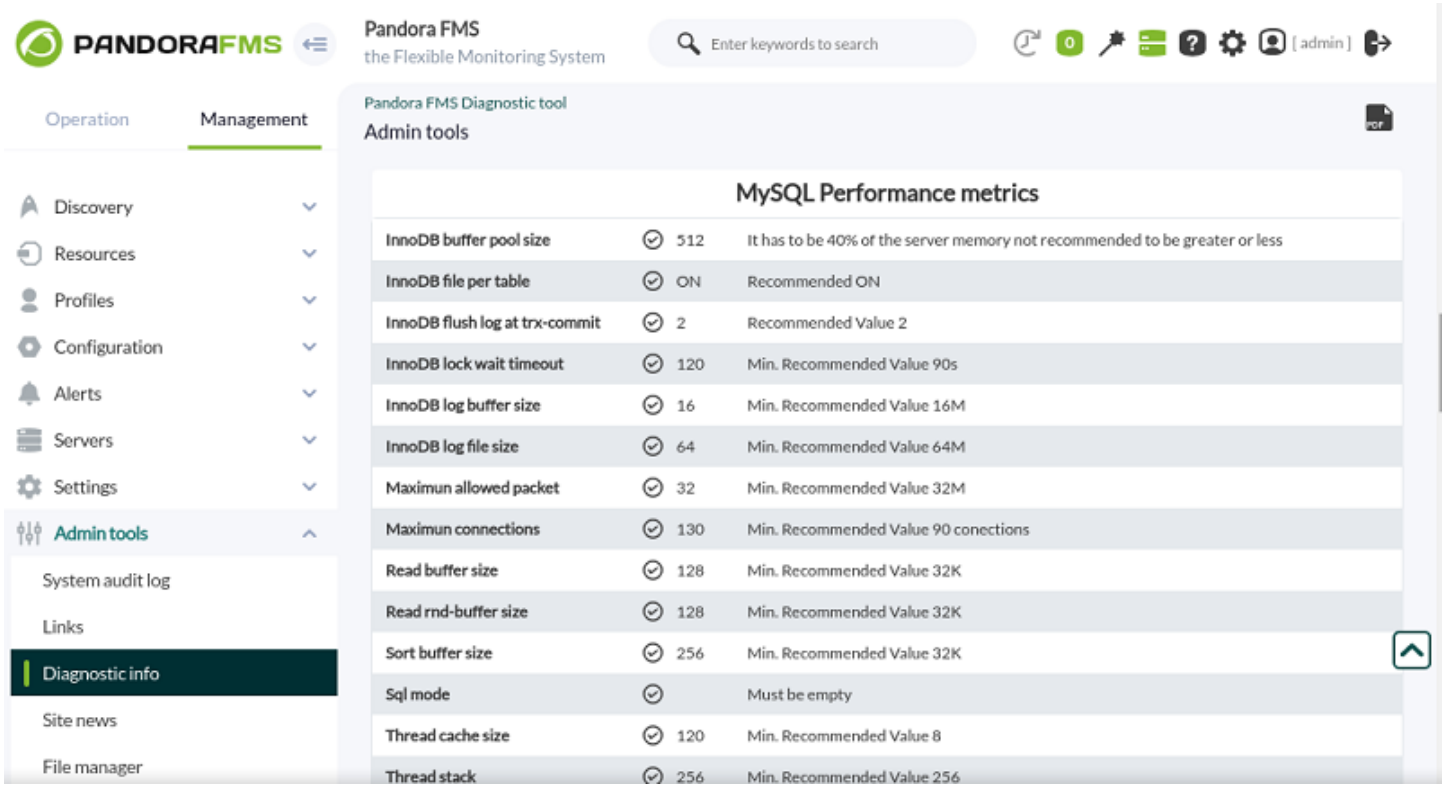
Dépannage et outils de diagnostic dans Pandora FMS

Il y a parfois des problèmes pour lesquels vous avez besoin d'une aide directe de l'équipe d'assistance de Pandora FMS. Pour faciliter la communication avec l'équipe d'assistance, les serveurs de Pandora FMS disposent de certains outils.

Diagnostic Info

Cet outil se trouve dans la section Management → Admin tools → Diagnostic Info, et est conçu pour

fournir les informations les plus importantes sur la configuration de Pandora FMS et sa base de données.



Retour à l'index de documentation du Pandora FMS