Pandora FMS の最適化と問題解決

Pandora FMS の最適化と問題解決

Pandora FMS ドキュメント一覧に戻る

Pandora FMS サーバは、約 2000のデバイス(5 から 8万モジュール、ハードウエアに依存します)のモニタリングが可能です。ただし、そのためには、データベースの設定を調整する必要があります。

この章ではまたPandora FMS の問題発見および解決のためのテクニックについて説明します。

Percona の最適化

すべてのテストと検証は、Percona Server for MySQL® 8 (推奨オプション) を使用して実行していますPercona Server for MySQL 8 と MySQL® 8 は類似しているため、両方のソリューションの間には優れた互換性があります。ここでの説明では MySQL という用語が使用されていますが、テストは Percona Server for MySQL で実行されることを常に念頭に置いてください。

“Pandora FMS におけるデータバックアップとリカバリ ” についての詳細は、こちらを参照してください。

一般的に推奨する設定

- 特に明示されていない限り、ここでの説明は MySQL バージョン 8 を前提としています。
- “MySQL 5.7 から MySQL 8 へのアップグレード ” も合わせて参照してください。

2GiBを超えるテーブルを持つ巨大なシステムを必要とするのであれば、次のガイドラインに従う必要があります。

- MySQL は 64Bit システムの利用を推奨しますPercona Server for MySQL 8 と MySQL® 8 は類似しているため、両方のソリューションの間には優れた互換性があります。ここでの説明では MySQL という用語が使用されていますが、テストは Percona Server for MySQL で実行されることを常に念頭に置いてください。32Bit システムでは、2038年以降重大な問題があります。
- よりよいパフォーマンスを得るために十分なメモリとCPUを用意します。われわれの経験ではCPUよりもメモリのほうが重要です。エンタープライズレベルであれば最低 4GiB 必要です。巨大なシステムであれば16GiB 割り当てるというのもよいでしょう。メモリが十分あれば、利用されるインデックスの大半をメモリ上に置くことでインデックスの更新が高速化されるということを忘れてはいけません。
- 障害が発生した場合にシステムを切り離すようにできるようにするのが良いでしょう。特定のサーバにデータベースがあるシステムの場合は、ギガビットイーサを利用すべきです。待ち時間はパフォーマンスにとって重要です。
- ディスクの最適化は、とても大きなデータベースにとっては大変重要です。データベースおよびテーブルを異なるディスクに分割すべきですMySQL では、シンボリックリンクを使うことができます。システムおよび、データベースで異なるディスクを使い、一つのハードディスクのアクセスを減らすように

することが重要です。

システムの応答時間を早めるためにSSD を利用することをお勧めします。

- もしMySQLサーバとクライアントを同じマシン上で起動するならTCP/IPのかわりにsocketを使用してMySQLサーバへ接続しましょう(これにより7.5%性能が改善されます)MySQLサーバに接続する際、ホスト名を指定しないか*localhost*と指定することでsocket経由で接続することができますMySQLサーバを1台しか起動しないのであればbinary logの出力とレプリケーションを無効にしましょう。
- Pandora FMS は MySQL で動作し、よりパフォーマンスの高い、修正版の MySQL (Percona Server for MySQL) を使うことを強くお勧めします。デフォルトでは、プラグインは Percona® 用に作成しています。

以下の点においてパフォーマンスは大きく影響を受けることに注意してください。

- MySQL® でレプリケーション設定を行う時のみ binary log を利用してください。
- クエリのトレースログや *スロークエリログ*は利用しないでください。これは特定の場合にのみ推奨されます

バイナリログ出力の停止

Pandora FMS HA システムを設定している場合は、バイナリログは必須です。ここに記載している内容は、Pandora FMS サーバが 1台の場合にのみ有効です。

多くの GNU/Linux ディストリビューションにおいてデフォルトで有効になっています。無効にするには、`my.cnf` ファイル (通常、`/etc/my.cnf` にあります) を編集し、次の行をコメントアウトします。

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

両方の行を*コメントアウト*し、MySQL サーバを再起動します

ディスク I/O パフォーマンス

“Pandora FMS におけるデータバックアップとリカバリ ” に関する詳細は、こちらを参照してください。

ディスク I/O に直結し、考慮すべきとても重要な 3つの設定がありますMySQL において、不適切な I/O アクセスは、通常最も重要なボトルネックになります。

innodb_log_file_size

```
innodb_log_file_size = 64M
```

デフォルトではこの値が設定されていますが、問題が発生した場合の回復とディスク占有率が増えることを除いては、事前検証なしに高くすることができます(512Mでも)。 MySQL のデフォルト値は 5M です。これは、トランザクション量が多い本番環境では非常に低い値です。 稼働中のシステムで値を変更するには、次のようにします。

1. データベースのフルバックアップ(SQL ダンプ)を行います。
2. Innodb バイナリインデックスファイル(通常は `/var/lib/mysql/ib*` です)を削除します。
3. my.cnf を編集します。
4. MySQL を再起動します。
5. ダンプを読み込みます。

サーバは、新たなトランザクションログファイルを新たなサイズで再生成し、通常通り動きだします。この処理は `innodb_file_per_table` トークン(後述)を有効化するのと同じであるため、処理全体を同時に行うことをお勧めします。

innodb_io_capacity

```
innodb_io_capacity = 300
```

デフォルトでは、このパラメータの値は 200 です。ただし、事前にシステムのディスクの IOPS を知る必要があります。正確な IOPS およびハードディスクのモデルを知ることができます。ここでのお勧めの値は、7500RPM → 100 IOPS, 15000 RPM → 190 IOPS, SSD → 1500 IOPS です。詳しくはこちらのリンクを参照ください。

innodb_file_per_table

テーブルごとにファイルを作成します。

Percona では各 InnoDB テーブルとそのインデックスを独自のファイルに保存することができます。この機能は、各テーブルに独自のテーブルスペースがあるため、“ マルチプルテーブルスペース ” と呼ばれます。

マルチプルスペーステーブルの使用は、特定のテーブルを別々の物理ディスクに移動したい場合や、残りのInnoDBテーブルの使用を中断せずにテーブルのバックアップを復元したい場合に役立ちます。

マルチプルテーブルスペースは `my.cnf` の `mysqld` セクションに以下の設定を追加することにより有効化できます。

```
[mysqld]
innodb_file_per_table
```

サーバーを再起動した後、InnoDBは、新しく作成された各テーブルを、テーブルが属するデータベースディレクトリ内の独自の `name_table.ibd` ファイルに保存します。これは MyISAM が行う動作と似ていますが、MyISAM はテーブルを `tbl_name.MYD` データファイルと `tbl_name.MYI` インデッ

クスファイルに分割します。

InnoDB の場合、データとインデックスは .ibd ファイルにまとめて保持されます。tbl_name.frm ファイルは通常どおり作成する必要があります。innodb_file_per_table 行が my.cnf から削除され、サーバが再起動された場合、InnoDB は共有表スペース・ファイルに表を再作成します。

innodb_file_per_table は、テーブルの作成にのみ影響します。 このオプションを使用してサーバを起動すると、.ibd ファイルを使用して新しいテーブルが作成されますが、共有テーブルスペース内の既存のテーブルに引き続きアクセスできます。このオプションを削除すると、共有スペースに新しいテーブルが作成されますが、複数のテーブルスペースに作成されたテーブルにアクセスすることは引き続き可能です。

トランザクションごとのディスク書き込みの回避

デフォルトでは、MySQL は各接続開始時の autocommit の値を autocommit =1 としています。MyISAM の場合、更新結果がディスクに保存されることを保証していないため、この設定はそんなに悪い設定ではありません。しかし InnoDB の場合、InnoDB テーブルへのあらゆる insert / update / delete によって、ディスクへの書き込みが発生するということを意味します。… デフォルトでは、innodb_flush_log_at_trx_commit = 0 となっています。

KeyBuffer の大きさ

システムに搭載された物理メモリ量に依存しますが、非常に重要なグローバル変数であり、これを設定することでDELETEおよびSELECTの実行速度が改善されます。

```
key_buffer_size = 4M
```

上記は、デフォルトの設定値です。

他の重要なバッファ

いくつかのディストリビューションでは、デフォルトで設定されていないいくつかのバッファがあります。これらのパラメータを設定することによりデフォルトより高いパフォーマンスを得ることができます。これらのトークンが MySQL の設定ファイルに存在するかどうかを確認することが重要です。

```
key_buffer_size=4M
read_buffer_size=128K
read_rnd_buffer_size=128K
sort_buffer_size=1M
```

```
join_buffer_size=4M
```

MySQL バージョン 8 以降では、*query cache* のサポートが廃止されました。詳細については以下を参照してください。

https://dev.mysql.com/blog-archive/mysql-8-0-retiring-support-for-the-query-cache/ .

InnoDB の同時実行スレッド

Pandora FMS の MySQL サーバパフォーマンスを向上に有効なパラメータがあります。そのパラメータは、innodb_thread_concurrency です。このパラメータは、MySQL で “ 同時実行スレッド ” をいくつにするのかを設定するのに利用します。

これは高度なパラメータであり、多くの同時実行を行うシステムでパフォーマンスチューニングが必要な場合 のみ手動で変更する必要があります。

このパラメータがうまく設定されていない場合、デフォルトよりも遅くなることがあります。そのため、次のような情報に注意を払うことが特に重要です。

- MySQL バージョン: 異なるバージョンの MySQL では、このパラメータはとても異なる動作をします。
- 物理プロセッサ数: この点については、公式のMySQLドキュメント を参照してください。

推奨値は、CPU(物理)の数に 2を掛けたものに InnoDB が配置されているディスクの数を加えたものです。

innodb_thread_concurrency の値は MySQL のいくつかのバージョンで変更されています、現在デフォルト値は 0 です。 0 は「可能な限り多くのスレッドを開く」ことを意味します。 したがって、良くわからない場合は次のように設定できます。

```
innodb_thread_concurrency = 0
```

MySQL フラグメンテーション

ファイルシステムのように、データベースもフラグメンテーションが発生しシステム全体の速度が低下します Pandora のように高いパフォーマンスを必要とするシステムでは、高速で信頼性の高いデータベースが必要です。高負荷なシステムでは、監視システムが停止する可能性があります。

Pandora FMS で良いパフォーマンスを得るには MySQL の設定がとても重要です。パフォーマンス

の問題がある場合は、MySQL の設定やデータベースに関する問題である可能性があります。

my.cnf 設定の確認

MySQL サーバの“基本設定”である my.cnf から確認します。設定ファイルは、INI フォーマットで書かれており、次のコマンドで場所を確認できます。

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

my.cnf は以下のような設定ファイルになっています(この例は、メモリ 4GB の平均的なサーバハードウエアです)。[mysqld] セクション内のトークンを確認します。

```
[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=1M
 join_buffer_size=4M

 sql_mode=""

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

MySQL 8 を使用しており、HA 環境がない場合は、セクション [mysqld] で次のコマンドを使用してバイナリ ログを無効にします。

```
skip-log-bin
```

my.cnf ファイルを変更した場合は、MySQL サービスを再起動します。

- systemctl status mysqld.service にてサービスの状態を確認します。
- エラーの確認には /var/log/mysqld.log の最後を見てください。
- より詳細は、MySQL サイトの こちらを確認してください。

データベースのリストア

サーバ管理のより詳細は、こちらを確認してください。

my.cnf に特定の変更を加えると (たとえば、innodb_file_per_table パラメータを追加するなど)、サービスを再起動したときにデータベースが機能しなくなる可能性があります。次のエラーが発生した場合は、以前の設定を復元し (管理者権限または *root* ユーザ を使用する必要があります)、データベースをバックアップする必要があります。

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

1. データベースをバックアップします。

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

2. MySQL サーバを停止し、バックアップフォルダへデータを移動します。

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

3. MySQL データ用の新たなフォルダを作成します(関連する権限はステップ5で割り当てられます)。

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

4. パラメータ --datadir でターゲット フォルダを指定して MySQL サーバーを初期化します。このプロセスにより、メモしておく必要がある一時パスワードが生成されます (標準出力に表示されるか、/var/lib/mysqld.log に保存されます)。

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

5. 新しいフォルダーに適切な権限を割り当てます。

```
chown -R mysql:mysql /var/lib/mysql
```

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

6. MySQL サービスを起動し、手順 4 のパスワードを使用して MySQL クライアントでログインします。

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

MySQL/Percona システムでは、多くの場合、my.cnf ファイルの設定パラメータが正しく読み込まれません。これは通常、これらの値が [mysqld] セクションの外部に書き込まれているためです。

7. ユーザ root のパスワードを任意のパスワードに変更します (ここでは Pandor4! を使用します)。

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

8. MySQL クライアントを終了し、新しいパスワードを使用して再度ログインできることを確認します。

9. MySQL クライアントに再度アクセスし、データベースを作成します。

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

10. データベースへバックアップをロードします。

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

11. 以前のインストールと同じ資格情報を使用 (ここでは Pandor4! を使用) して Pandora FMS のアクセスユーザを作成し、データベースに対する権限を付与します。

```
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. my.cnf ファイルを設定して MySQL サービスを再起動した後、これらの変更が正しく適用されていることを確認する必要があります。そのためには、変数を 1 つずつ確認します。

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

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

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

または、次のような一般的な問い合わせでも構いません。

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

Pandora FMS (Web コンソールとサーバの両方) がデータベースに正しく接続でき、すべてが正常に動作することを確認したら、mysql.bak ディレクトリを削除できます。

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

各テーブルごとに分割されたデータファイルがアクティブであるか確認

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

my.cnf ファイルで innodb_file_per_table トークンを有効化した場合、それぞれのテーブルに対応した .ibd ファイルが 100以上(Pandora FMS のバージョンに依存)存在します。このようなファイルが無い場合は、全データが大きなファイルに記録されます。これは、テーブルのフラグメンテーションが全テーブル共通して発生し、毎週パフォーマンスが劣化することを意味します。

すでに単一のデータベースで実行している場合は、mycnf ファイルの設定を変更し MySQL を再起動したあとにデータベースを再作成する必要があります。

特定のテーブルの最適化

断片化の問題を解決するためのもう 1 つの「それほど劇的ではない」解決策は、MySQL OPTIMIZE ツールを使用して Pandora FMS の特定のテーブルを最適化することです。これを利用するには MySQL から直接以下を実行します。

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

これにより、システムが稼働しているときに時間の経過とともにパフォーマンスが低下するのを防ぎます。

非常に大規模な環境では、OPTIMIZEオプションが「ブロック」される可能性があります。この場合、最善のオプションは DB を再構築 することです。

これらの操作を行った後、次を実行します。

```
FLUSH TABLES;
```

MySQL マニュアルより:

InnoDB テーブルの場合OPTIMIZE TABLE は ALTER TABLE にマップされ、テーブルを再構築してインデックス統計を更新し、クラスター化インデックス内の未使用領域を解放します

MySQL 特殊トークン

MySQL には、パフォーマンスを向上させたり低下させたりできる非常に「特別な」トークンがいくつかあります。

- innodb_flush_method:

```
innodb_flush_method = O_DIRECT
```

この重要なパラメータは、ディスクに情報が書き込まれる方法に影響します。

- innodb_lock_wait_timeout:

```
innodb_lock_wait_timeout = 90
```

これはボトルネックがある場合に役立ち、MySQL が停止することはありません。90 ロックを超えて続く場合は、実際に問題が発生しています

テーブルごとのフラグメンテーションを確認

MySQL の CLI を使って、以下のクエリを実行します。

```
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;
```

フラグメンテーションのあるテーブルが次のように表示されます。

```
+--------+------------------------+-------------+--------------+-----------+---
---------+
| 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 |
+--------+-------------------------+-------------+--------------+-----------+---
---------+
```

このクエリは、断片化が 10 % を超えるテーブルでのみ機能します。大きすぎるテーブル (tagente_datos など) は、断片化が激しい場合、最適化に長い時間がかかることがあります。これは、実稼働システムに影響を与える可能性があります。

このような大きなテーブルを最適化する場合は注意が必要です。通常の環境は 1 年に 1 回、大規模な環境は 6 か月ごとに最適化できます。

tagent_module_inventory テーブルを最適化する (この場合、データベースの名前は pandora) には次のようにします。

```
OPTIMIZE TABLE pandora.tagent_module_inventory;
```

次のような警告メッセージが表示されます。

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

再度確認すると、フラグメンテーションが無くなっていることがわかります。

```
+--------+------------------------+-------------+--------------+-----------+---
---------+
| 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 |
+--------+------------------------+-------------+--------------+-----------+---
---------+
```

最適化を実行するには、操作を実行できるだけのディスクの空き容量が必要です。そうでないとエラーとなり処理が実行されません。

システム負荷

より一般的な事として、システムの(ディスク) I/O がボトルネックになっていないことを確認する必

要があります。システムの状態を取得するのにvmstat コマンドを実行します。

```
vmstat 1 10
```

最後のカラム (CPU WA) を見て、10より大きい値であればディスク I/O の問題があり、解決する必要があります。

CPU-US が高いのは通常です。しかし、CPU-SY は 10～15 を超えないようにすべきです。

SWAP-SI および SWAP-SO はゼロであるべきです。そうでなければシステムがスワップメモリを使っていることを意味し、パフォーマンスを落とします。メモリを増やすか、アプリケーションが使うメモリを減らす(Pandora サーバのスレッドMySQL のバッファの調整など)必要があります。

以下は、“ 正常 ” なシステムの出力サンプルです。

```
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
```

MySQL テーブルパーティショニング

MySQL テーブルパーティショニングを使用するには、上記で説明した *複数のテーブルスペース* システム (innodb_file_per_table) も使用する必要があります。

MySQL はテーブルパーティショニングをサポートしており、これにより、非常に大きなテーブルを

論理的なサブディビジョンなどの小さなチャンクに分散できます (詳細については、MySQL マニュアル を参照してください)。

もし Pandora FMS (メインおよびヒストリの両方の)データベースに大量のデータがあり、グラフ描画のようなデータを参照する処理がとても遅いと感じるなら、テーブルパーティショニングを行うことでパフォーマンスを改善できるでしょう。

自動パーティショニング

Pandora FMS は、Web コンソールの一般設定のセクション ヒストリデータベース から設定されている場合、ヒストリデータベースの月次パーティショニングを自動的に実行します。詳細については、こちらを参照してください。

手動パーティショニング

Pandora FMS をインストールし、ヒストリデータベースを有効にすると、これが最大量のデータを保持するものになります。テーブルパーティション分割の実行が推奨されます。 これに指定されているテーブルは、正確には tagente_datos、tagente_datos_string、tagente_datos_inc、および tevento です。テーブル tagente_datos の手動による実際の処理については、以下で説明します (「自動パーティション分割」も参照してください)。

まず、ディレクトリ /var/lib/mysql/pandora_history/*.ibd に多くのファイル (テーブルごとに 1 つ) があることを確認する必要があります。そうでない場合は、データベースの *dump* を実行し、my.cnf ファイルの設定を変更し、MySQL を再起動し、現在のデータベースを削除して *dump* から再作成する必要があります。

innodb_file_per_table が有効になっていることを確認したら、2 つのメインデータベースを異なるパーティションに分離します。

- この操作を完了するには、十分なディスク領域が必要です。tagente_datos.ibd ファイルのサイズを確認する必要があります。たとえば、このファイルが 10 GB を占める場合、操作を開始するには 15 GB の空き領域が必要になります。
- この操作は、テーブルのサイズに応じて長い時間がかかることがあります。たとえば、100 日間 (50,000,000 行以上) の約 7500 個の *モジュールデータ* を含むテーブルを分割するには、1 時間半 かかります。

これは、2023 年のこれまでと今後の月全体をパーティション分割する例です。処理を開始するには、MySQL CLI で次のクエリを実行する必要があります。

```
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)
);
```

次に、パーティションを再編成するために、次のクエリを毎月実行する必要があります。

```
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);
```

“Jan24” を当月に変更します。

この操作は、tagente_datos テーブルの大きさに応じて、数時間かかる場合があることに再度注意してください。次のコマンドを実行して、パーティションファイルのサイズを監視することで、処理を確認できます。

```
[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
```

データベースの再構成

Pandora FMS のバックアップとデータ復旧の詳細については、こちら を参照してください。

部分的再構成

MySQLは他のデータベースシステム、たとえば Oracle® と同様、時間がたつにつれ、性能が劣化していきます。これは大きなテーブルに対してデータの削除と追加を続けることによって発生するデータのフラグメンテーションによるものです。大量のトラフィックが発生する大きな環境において、性能の改善および劣化を防ぐ非常に簡単な方法があります。それは定期的にデータベースの再構築を実施することです。

そのために、1時間程度のサービス停止を計画すべきです。

計画的なサービス停止時に、Pandora FMS Webコンソールとサーバも停止すべきです (注意 tentacle サーバはデータを受け取れるようにしておき、サーバが復旧次第データを処理できるようにします)

停止したらDB ダンプ (エクスポート) を実行します。この例では、データベースの名前は pandora3 で、ユーザは root です。

```
mysqldump -u root -p pandora3> /tmp/pandora3.sql
```

データベースを削除します。

```
mysql -u root -p
```

```
DROP DATABASE pandora3;
 Query OK, 87 ROWS affected (1 MIN 34.37 sec)
```

データベースを作成し、先ほどエクスポートしたデータをインポートします。

```
CREATE DATABASE pandora3;
USE pandora3;
SOURCE /tmp/pandora3.sql
```

データベースのサイズとマシンで使用可能なリソースに応じて、数秒から数分かかる場合があります。

この処理を自動化することは可能ですが、非常にデリケートなため、手動で実行するのが最善のオプションです。

全体の再構築

まずPFMS サーバを停止する必要があります。

```
/etc/init.d/pandora_server stop
```

または、systemd がインストールされている場合:

```
systemctl stop pandora_ha
```

次に、すべてのスキーマとデータのエクスポートを実行する必要があります。データベースが大きい場合に備えて、データベースのバックアップを保存するのに十分なスペースがあるパーティション内のディレクトリを選択します。次のコマンドを実行します。

```
mysqldump -uroot -p pandora --single-transaction > backup.sql
```

処理が完了すると、現在のディレクトリ内のファイル backup.sql にデータが格納されます。

これが完了したら、MySQL を次のように停止します。

```
systemctl stop mysqld.service
```

MySQL インストールディレクトリ (/var/lib/mysql) にアクセスし、binlog.000001、binlog.000002、… および binlog.index という名前のファイルを削除します。さらに、データベース pandora と同じ名前のディレクトリも削除します。

```
rm -rf /var/lib/mysql/pandora
rm -rf /var/lib/mysql/binlog.0*
rm -rf /var/lib/mysql/binlog.index
```

これらの手順が完了したら、次のコマンドで MySQL を再起動します。

```
systemctl start mysqld.service
```

次のコマンドで MySQL ターミナルにアクセスします。

```
mysql -uroot -p pandora
```

pandora データベースが選択されますが空の状態です。Pandora FMS データベースから実行した バックアップをデータベースにインポートするために次のコマンドを使用します。

```
SOURCE backup.sql;
```

最後に、PFMS サーバ を起動する必要があります。

```
/etc/init.d/pandora_server start
```

または、systemd がインストールされている場合:

```
systemctl start pandora_ha
```

オプションのインデックス

他のシステムリソースを犠牲にしてMySQL パフォーマンスを最適化できるいくつかの場合があります。

以下のインデックスの最適化はグラフ生成を(とても)高速化しますが、多くのディスクスペースを必要とします。また、インデックスのオーバーヘッドにより、若干 INSERT/DELETE 処理が遅くなります。

```
ALTER TABLE `pandora`.`tagente_datos` ADD INDEX `id_agente_modulo_utimestamp`
(`id_agente_modulo`,`utimestamp`);
```

現在MySQL の Pandora FMS の最も重いテーブルでは、この最適化がデフォルトになっています。MySQL テーブルを最適化する前に専門家に相談するのが良いです。

スロウクエリ

いくつかのシステムでは保持している情報によって、通常よりもシステムのパフォーマンスが悪いスロウクエリが見られることがあります。テーブルを最適化するために、(システムのパフォーマンスに影響する)一定時間を超えたこのタイプのクエリをログに記録するようにできます。これを有効にするには次のようにします。

- my.cnf を編集し、次の設定を加えます。

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

- 利用できるように次のように設定します。

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

- mysql を再起動します。
- スロウクエリの分析が終了したら、ファイル my.cnf をリセットして、集約された行にコメントを付け、MySQL サービスを再起動することを忘れないでください。

参考サイト

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

Pandora FMS のキャパシティ計測

この章では、高いキャパシティが必要な環境での Pandora FMS を設定するための異なる手法を説明します。また、処理を実行する環境を調整するのに便利な負荷テストを行うためのツールについても説明します。

Pandora FMS は、1つのサーバでデータベース、コンソール、サーバを動かした場合、2500エージェントに対応できるように設定されています。推奨するエージェント数は、1システムあたり 2500 です。しかし、この数は、XML エージェントの割合、リモートモジュールの割合、監視間隔、システムのメモリ量に応じて変化します。

これらの全ての要素が、1つのシステムで管理できるエージェント数に関わります。テスト環境では、通常のハードウエアの 1台のサーバで 10000 エージェントを実行できましたが、高度な最適化をしています。

高キャパシティサーバの設定例

例えば16GB の RAM および 8 CPU の RHEL 8 マシンで、データサーバが最大のパフォーマンス (XML 処理) を出すように最適化したいと思います。

my.cnf

重要なパラメータのみを示しています。

```
[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=1M
join_buffer_size=4M
sql_mode=""

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

pandora_server.conf

重要なパラメータのみを示しています。

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

以下の点を認識しておく必要があります。

- verbose パラメータで設定された数値は、ログに書き込まれる情報の量を指し、3 を超えないようにすることをお勧めします。数値が高くなると、ログに書き込まれる情報量が多くなるため、Pandora FMS のパフォーマンスが低下します
- パラメータ server_threshold の数が大きい(15)と、データベースへの影響が少なくなり、処理されるファイルの最大数を大きくすると、サーバはファイルを検索してバッファをいっぱいにします。 これら 2つの要素の設定は密接に関連しています。 ネットワークサーバを最適化する場合は、server_threshold を 5 または 10 より下げることをお勧めします。
- dataserver_threads に設定された非常に多数のスレッド(5以上)は、ネットワークサーバやプラグインサーバなどE/S 待機時間が長い処理がある場合にのみメリットがあります。継続的に処理されているデータサーバの場合は、パフォーマンスに影響を与える可能性さえあります。 データベースの速度が遅いシステムでは、使用するスレッドをさらに少なくします。1～10 のさまざまな組み合わせを試してください。ネットワークサーバ用にシステムを最適化する場合、その数は 10～30の間の大きな値にします。

キャパシティ分析ツール (Capacity)

Pandora FMS には、ハードウエアおよびソフトウエアで、取得可能なデータ量を適切に計測できるいくつかのツールがあります。一つは、ダミーデータで直接データベースへアクセスするもの (dbstress)、もう一つは、ダミーの XML ファイルを生成するもの (xml_stress) です。

Pandora FMS の XML 負荷

Pandora FMS エージェントから送られるような XML データファイルを生成する小さなスクリプトです。デフォルトでは以下にあります。

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

スクリプトは、テキストファイルからエージェント名を読み取り、設定ファイルに従って各エージェントの XML データファイルを生成します。ここで、モジュールはテンプレートとして定義されています。

モジュールの値はランダムな値になります。モジュールデータの初期値および変化率は設定可能です。

スクリプトは次のように実行します。

```
./pandora_xml_stress.pl < 設定ファイル >
```

pandora_xml_stress.conf という < 設定ファイル > に指定するファイルの例を示します。

```
# 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
```

エージェントのローカル設定の送受信

pandora_xml_stress.conf で、get_and_send_agent_conf を 1 に設定して実行した場合、テスト負荷エージェントで、通常のエージェントのように設定ファイルおよび md5 を送ることができます。PFMS Web コンソールから、pandora_xml_stress の以降の実行のリモート設定を変更できます。

ほかにも、pandora_xml_stress.conf 内の directory_confs にて、テストエージェントの設定ファイル .conf をどこに保存するかを設定できます。

設定ファイル

- max_threads スクリプトの実行スレッド数。処理率を改善します。
- agent_file 行ごとに名前を書いたファイルのパス。
- temporal 架空の XML データファイルを生成するディレクトリのパス。
- log_file スクリプトの実行について情報を出力するログのパス。
- xml_version XML データファイルのバージョン。(デフォルトは 1.0)
- encoding XML データファイルのエンコーディング。(デフォルトは ISO-8859-1)
- os_name 仮想エージェントの OS 名。(デフォルトは Linux)
- os_version 仮想エージェントの OS バージョン。(デフォルトは 2.6)
- agent_interval 仮想エージェントの実行秒間隔。(デフォルトは 300)
- time_from 仮想 XML データの開始時間。“ 年-月-日 時間:分:秒 ” というフォーマットにて。
- time_to 仮想 XML データの終了時間。“ 年-月-日 時間:分:秒 ” というフォーマットにて。
- get_and_send_agent_conf 0 または 1 の値です。有効な場合、仮想エージェントは、リモート設定により、より新しいエージェントの設定ファイルをダウンロードしようとします。Pandora FMS Enterprise のコンソールから編集可能になります。
- startup_delay それぞれのエージェントがファイル生成を開始するまでの時間を秒で指定します。競合を回避するために利用します。
- timezone_offset タイムゾーンのオフセット値です。
- timezone_offset_range ランダムに指定した範囲でタイムゾーンを生成します。
- latitude_base 数値です。仮想エージェントを表示する緯度です。
- longitude_base 数値です。仮想エージェントを表示する経度です。
- altitude_base 数値です。仮想エージェントを表示する高度です。
- position_radius 数値です。指定した半径内の円に仮想エージェントがランダムに表示されます。

スクリプト設定ファイル内の 1 つのモジュールの定義。リモート設定が有効になっている場合も同様です。次のようになります。

```
module_begin
module_name < module_name >
module_type < module_type_data>
module_description < description >
module_exec type =< xml_stress_type_generation >;< another_option >;<
```

```
another_option > …
module_unit < units >
module_min_critical <value>
module_max_critical <value>
module_min_warning <value>
module_max_warning <value>
module_end
```

```
module_begin
module_name <モジュール名>
module_type <タイプ, 例: generic_data>
module_description <説明>
module_exec type=<type_generation_xml_stress>;<; 区切りの他のオプション>
module_unit <ユニット>
module_min_critical <値>
module_max_critical <値>
module_min_warning <値>
module_max_warning <値>
module_end
```

それぞれの項目は次のように設定可能です。

- type_generation_xml_stress: RANDOM,SCATTER,CURVEのいずれかを設定できます。
- module_attenuation <値>: モジュールの値を指定した値で掛け合わせます。通常 0.1 と 0.9 の間です。
- module_attenuation_wdays <値> <値> … <値>: 指定した日にのみモジュールの値を計算します。日曜(0) から土曜(6) を指定できます。例えば、以下のモジュールでは、土曜と日曜にネットワークトラフィックを 50% にします。

```
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 <値>: 1に設定すると、モジュールの以前の値を常に新たな値に加えます。値が増え続ける機能です。
- その他: 実効タイプに依存して、どのようなオプションがあるかは以下を確認してください。

RANDOM

次のオプションがあります。

- variation 前回の値から変化が発生する可能性を % で指定します。
- min 値の最小値を指定します。
- max 値の最大値を指定します。

Numeric

min と max の間の数値をランダムに生成します。

Booleans

0 または 1 の値を生成します。

String

min と max の間の長さの文字列を生成します。文字は、A から Z の間のランダムで、大文字、小文字を含み、また数字や記号を含みます。

外部データソース (SOURCE)

データのソースとしてプレーンテキストファイルを利用することができます。次のオプションがあります。

- src: ソースデータファイル

ファイルは、1行に1データを含む形式で、行数に制限はありません。例えば次の通りです。

```
4
5
6
10
```

二種類のデータ(数値と文字列)を扱うことができます。これらのモジュールは、ファイルのデータ順番に読み込んでPandora でのモジュールデータを生成します。上記のデータの例では、次のように表示されます。

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

SCATTER

数値データの場合のみ有用で、ハートビートのようなグラフを生成します。これは通常の値で、ある時間に *“beat”* を出します。

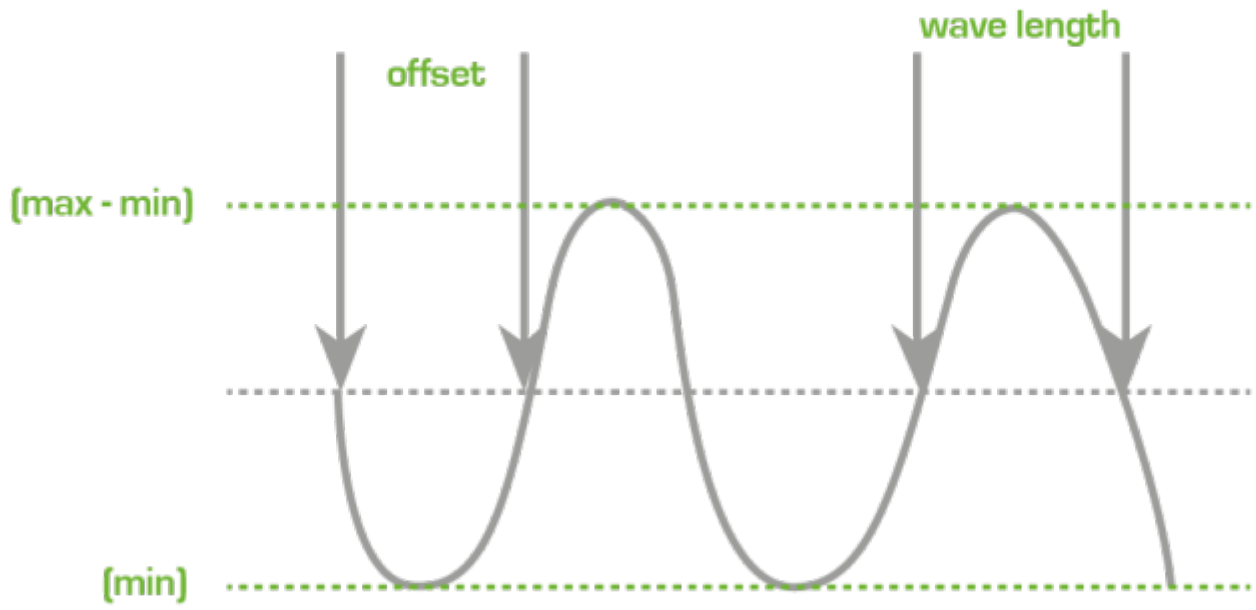
次のオプションがあります。

- min とりうる最小の値。
- max とりうる最大の値。
- prob *“beat”* を生成する頻度を % で指定します。
- avg *“beat”* が無い場合に、デフォルトで表示する平均値。

CURVE

三角関数を使った曲線でモジュールデータを生成します。 次のオプションがあります。

- min とりうる最小の値。
- max とりうる最大の値。
- time_wave_length 山が出現する時間。
- time_offset モジュールの値が 0 の時の波形の開始タイミングを秒で指定します。(正弦波に似ています)



注意事項: このツールは、すべてのエージェントで、300秒から 30日の間隔を使用する “random”、“curve”またはブール名のモジュールを検索するように事前設定されています。

データサーバの処理能力の計測方法

pandora_count.sh というスクリプトが Pandora FMS サーバパッケージの /usr/share/pandora_server/util/ ディレクトリにあります。このスクリプトは、データサーバの XML ファイル処理率を計測するのに利用します。このツールは、/var/spool/pandora/data_in に残っているファイルを利用します。そのため、未処理の数千のファイルを用意しておく (もしくは前述のツールで生成しておく) 必要があります。実行後は、CTRL+C で停止できます。

このスクリプトは、現在存在するファイルをカウントし、10秒前に処理したものを除外し、結果を 10 で割ることによって、1秒間の処理率を求めています。これは初歩的なソリューションですが、サーバの設定を修正するための情報を提供します。

Pandora FMS の DB 負荷

データベースパフォーマンスを確認するためのツールがあります。これはまた、架空のデータを定期的もしくは不定期に生成するためにも利用できます。エージェントを作成し、このツールを使って自動的にデータを挿入するためのモジュールを作成しておく必要があります。

- *random*: 不定期データを生成します。
- *curve*: 三角関数を利用して曲線データを生成します。異なる間隔で補完を見るのに便利です。
- *boolean*: ランダムな boolean データを生成します。

random, *curve* および *boolean* という語を含む任意の名前を使うことができます。

- random_1
- curve_other

data_server モジュールのみ選択することができます。

*Pandora FMS の DB 負荷ツール*の調整

このツールは、すべてのエージェントから、*random*、*curve* または *boolean* という名前がついかモジュールを検索するようにあらかじめ設定されています。また、実行間隔は 300秒から 30日の間を指定できます。

もしこの設定を変更したい場合は、*pandora_dbstress* スクリプトを編集し、ファイルの先頭にあるいくつかの変数を変更する必要があります。

```
# 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;
```

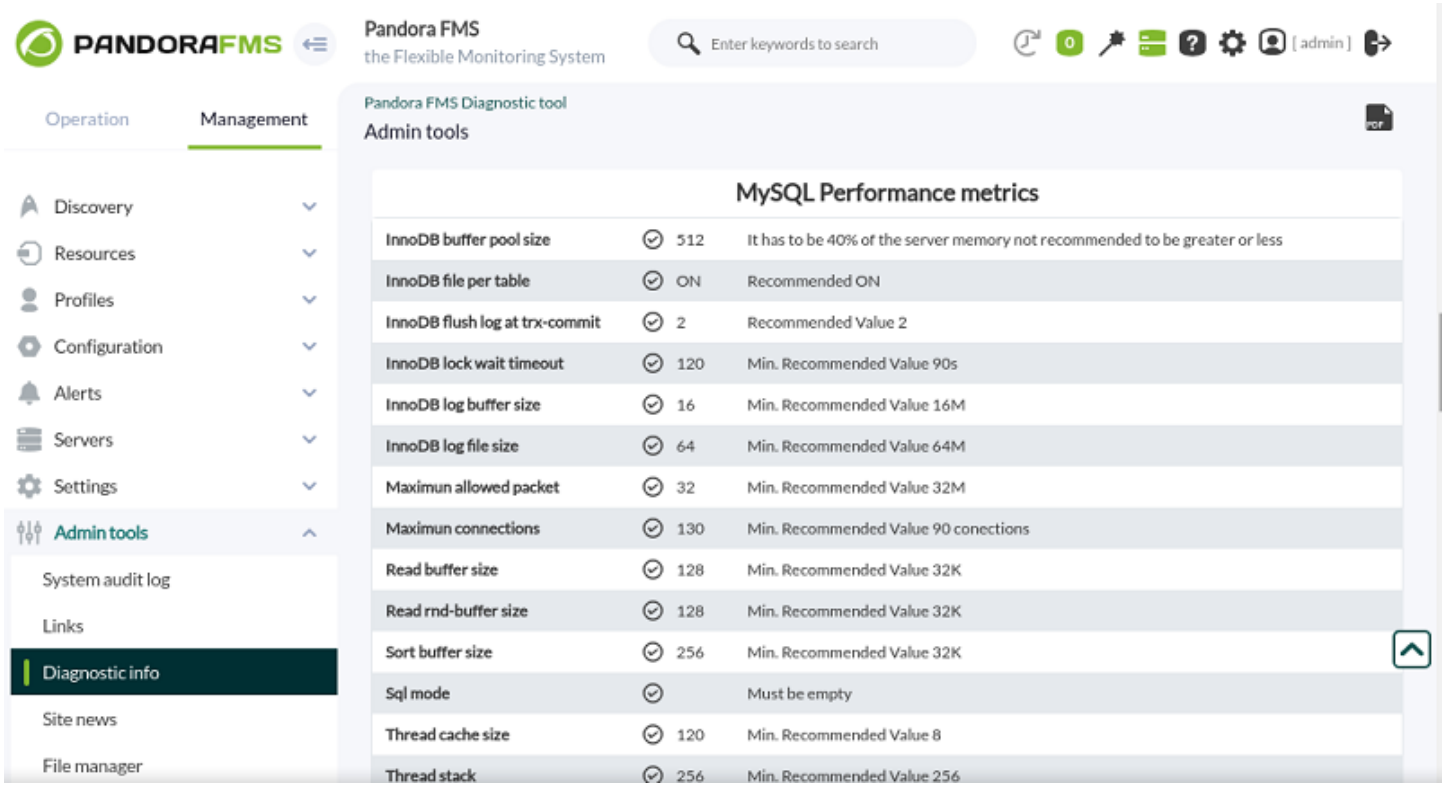
1. 最初の行の target_module の設定では、対象のモジュールを指定するか -1 を指定すると全てが対象になります。
2. 2行目の target_agent は、エージェントの指定です。
3. 3行目の target_interval は、秒単位で実行間隔を指定します。
4. 4行目の target_days は、現在日時から何日後までかを表します。

Pandora FMS の診断ツール

場合によっては、Pandora FMS サポートからの直接的な支援が必要な問題が発生することがあります。サポートチームとのコミュニケーションを容易にするために、Pandora FMS サーバーにはそのためのツールがいくつかあります。

診断情報

このツールは 管理(Management) → 管理ツール(Admin tools) → 診断情報(Diagnostic Info) セクションにあり、Pandora FMS の設定とデータベースに関する最も重要な情報を提供するように設計されています。



Pandora FMS ドキュメント一覧に戻る