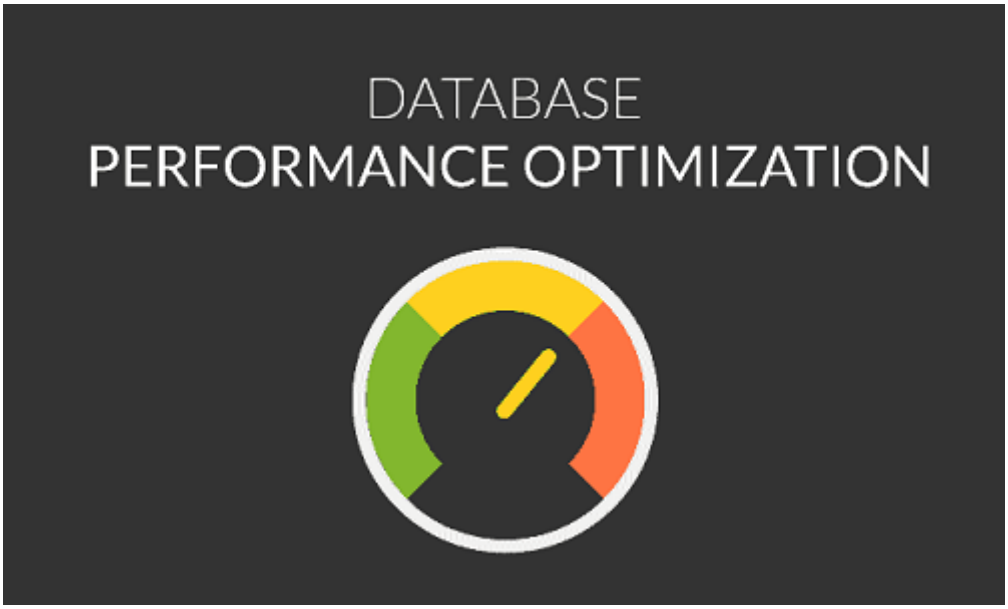
Optimization and Problem Solving of Pandora FMS

Optimization and Problem Solving of Pandora FMS

Pandora FMS optimization and troubleshooting

Introduction

Pandora FMS server can monitor about 2000 devices (between 5 and 80 thousands modules, depending on available hardware); but this also requires fine-tuning the configuration of the database.



This article also explains some techniques to detect and solve problems of your Pandora FMS installation.

MySQL optimization for Enterprise version

To learn more about “Data Backup and Recovery in Pandora FMS”, go to this link.

General Advice

- Unless otherwise specified, this entire topic refers to MySQL version 5.7.
- See also “Upgrading from MySQL 5.7 to MySQL 8”.

To work with tables larger than 2 GiB, it is necessary to follow some guidelines:

- MySQL® recommends using a 64-bit system. 32-bit systems may have serious problems from year 2038 onwards.
- The more RAM and the more CPU, the better the performance. In our experience, RAM is more important than CPU. The minimum for an enterprise level system will be 4 GiB. A good choice for a large system is 16 GiB. Remember that more RAM can speed up key updates by keeping the most used key pages in RAM.
- It is a good idea to be able to remove the system in case of failure. For systems where the database is on another dedicated server for the database use Gigabit Ethernet, preferably with fiber optics rather than copper. Latency is as important as performance.
- Disk optimization is very important for very large databases: databases and tables will have to be split on different disks. In MySQL® you can use symbolic links for this. Use different disks for the system and the database.

The use of SSD disks is recommended due to their speed and improved system latency.

- If possible use:

```
--skip-locking
```

- This will turn off external locking and provide better performance (enabled by default on some systems).
- If you start the client and MySQL® server are in the same machine, use sockets instead of TCP/IP connexions when connecting with MySQL® (this could result in an improvement of a 7.5%). Do it without specifying the host name or the localhost when connecting with MySQL®. Disable the start of the binary session and the *replication* if it only launches a MySQL® host server.
- Using recent versions of MySQL® (5.5 or later) over MySQL® older versions (5.0.x) can offer up to a 20% difference in performance.
- We recommend the use of MySQL® modified version (Percona Server for MySQL®), which offers better performance. By default the plugins programmed are for Percona®.

Please note that performance is greatly affected by the following points:

- Only use binary logs if you use a MySQL® configuration with replication.
- Do not use query traceability logs or slow query logs.

Automatic tools for configuration

There are many tools to optimize the setup of MySQL server.

MySQL Tuning Primer, by Mattew Montgomery, is a command line tool used to check your MySQL performance, and gives you a few tips and suggestions to improve it. Check it at https://bugs.launchpad.net/mysql-tuning-primer

Disable binary replication

If you have configured a Pandora FMS HA system, the binary replication is necessary. This recommendation is only valid if you have a single Pandora FMS server.

It is enabled by default on most GNU/Linux distros. To disable it, edit the `my.cnf` file, usually in `/etc/my.cnf` and comment the following lines:

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

Comment both lines, and then restart MySQL Server.

Disk IO Performance

To learn more about "Data Backup and Recovery in Pandora FMS", go to this link.

There are three very important configuration tokens, directly related to disk IO, and should be considered because improper IO access is usually the most important bottleneck in MySQL.

innodb_log_file_size

```
innodb_log_file_size = 64M
```

This value is set by default, which can be higher (even 512M) without any risk, except for recovery in case of any problem or higher disk occupation. The default value of MySQL is 5M, which is very low for production environments with high transaction volume. To change this value with an already running system:

1. First make a complete DUMP in the databases.
2. Delete the Innodb binary index files (usually in `/var/lib/mysql/ib*`.
3. Change file `my.cnf` with the chosen value.
4. Restart MySQL.
5. Load the SQL DUMP.

Since the process is the same as the one to activate the *`innodb_file_per_table`* token (described below), it is recommended to do the whole process simultaneously.

innodb_io_capacity

```
innodb_io_capacity = 100
```

This parameter has the value 100 by default, but you have to know the IOPS of the system disk. You can find out exactly by looking for IOPS and the exact hard disk model (obtained via smartctl), where the recommended values are: 7500RPM → 100 IOPS, 15000 RPM → 190 IOPS, SSD → 1500 IOPS.

To install smarctl you must request the full smartmontools
package; for example:

`yum install smartmontools`,

`apt install smartmontools`, etc.

innodb_file_per_table

Use a table space for each table (from the MySQL 5.0 Spanish manual)

In MySQL 5.0, it is possible to store each InnoDB table and its index in its own file. This feature is called “multiple tablespaces” because each table has its own table space.

The use of multiple space tables can be useful for users that want to move specific tables to separate physical disks or the ones who want to restore table back ups without interrupting the use of the rest of the InnoDB tables.

It is possible to activate multiple table spaces adding this line to the my.cnf Mysqld section

```
[mysqld]
innodb_file_per_table
```

After restarting the server, InnoDB will store each new created table in its own `name_table.ibd` file in the database directory to which the table belongs to. This is similar to what the MyISAM store motor does, but MyISAM divides the table in a `tbl_name.MYD` data file and a `tbl_name.MYI` index file.

For InnoDB, data and index are kept together in the `.ibd` file. The `tbl_name.frm` file should be created as usual. If the innodb_file_per_table line is take off from `my.cnf` and the server is restarted (see previous instructions for innodb_log_file_size), then InnoDB will create again the tables in the shared table space files.

innodb_file_per_table affects only table creation. If you start the server with this option, then the new tables will be created using `.ibd` files, but you could still have access to the existing tables in the shared table space. If you remove the option, then the new tables will be created in the shared

space, but it will be still possible to have access to the tables created in multiple table spaces

Avoiding Disk Flush in Every Transaction

MySQL establishes `autocommit =1` for each connection by default. This is not bad for MyISAM, since what one person writes in the disk is not guaranteed, but for InnoDB it means that any insert / update / delete in an InnoDB table will be registered on the disk (flush).

So, would it be bad if it always writes on the disk? Not at all. It ensures that when there is any compromising event, the data will be there when the database is restored after an incident. The problem is that the DB performance is limited by the physical speed of the disk.

Given that the disk has to write the data in a disk before the writing has been confirmed, this will take some time. Even when considering a searching average time of 9ms for the disk writing, it is limited to approximately 67 commits/ sec1, which is very slow. And while the disk is busy trying for the sector to be written, it cannot read.

InnoDB can avoid some of this limitations by associating some writing together, but, even with this, this restriction still exists. You can prevent if from writing at the end of each transaction, ensuring that it uses an “automatic” writing system, which writes approximately every second. In case of failure, the data from the last second could be lost, but his is something more bearable considering that it achieves greater efficiency. To do it, use the following configuration token `innodb_flush_log_at_trx_commit = 0`. It has this value in the configuration by default.

Bigger KeyBuffer size

Depending on the system total RAM, it is a very important global parameter that speeds up DELETES and INSERT.

```
key_buffer_size = 4M
```

This is the default value in the configuration.

Other important buffers

There are several buffers that are empty by default in some distributions. Modifying these parameters can improve performance significantly compared to the default one. It is important to make sure that these tokens exist in the MySQL configuration file.

```
query_cache_size = 64M
query_cache_limit = 2M
join_buffer_size = 4M
```

For MySQL version 8, and later versions, the MySQL development team has withdrawn support for *query cache*, for more information please visit the following web link: https://dev.mysql.com/blog-archive/mysql-8-0-retiring-support-for-the-query-cache/ .

Improving InnoDB Concurrency

There is a parameter that can affect Pandora MySQL server performance pretty much. This parameter is `innodb_thread_concurrency`. This parameter is used to specify how many "concurrent threads" MySQL can raun. Misconfiguring this parameter can make it go slower than the default one, so it is especially important to pay attention to several parameters:

- MySQL version. In different versions of MySQL this parameter operates VERY differently.
- Real number of physical processors.

Here you can read the official MySQL documentation.

The recommended value is the number of CPUs (Physical) multiplied by 2 plus the number of disks where InnoDB is located.

In later versions of MySQL (> 5.0.21) the default number is 8. A value of 0 would mean that it "opens up as many threads as possible." So in case of doubt you can use:

```
innodb_thread_concurrency = 0
```

Different people ("Variable's Day Out #5: innodb_thread_concurrency", "Do we still need innodb_thread_concurrency?") have done tests and have found performance problems on servers with multiple physical CPUs when using a very high number, with relatively old versions of MySQL (like 2008 and so on).

MySQL Fragmentation

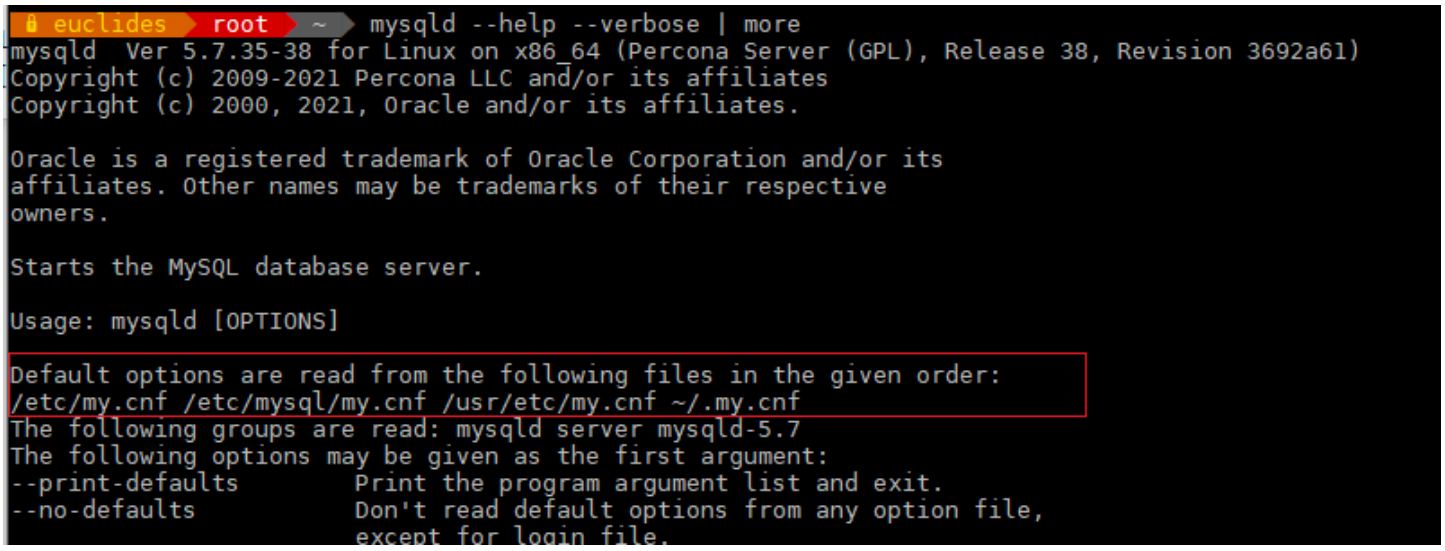
Like the filesystems, databases also will fragment themselves, slowing the whole system down. In a high performance system like Pandora FMS it is vital that the database state does not affect the system performance. In overloaded systems, the database could block and force the monitoring system to fall down.

Setting up the MySQL server could make Pandora FMS faster, so if you have performance problems, the reason might be a problem in MySQL Setup or problems related with the database.

Check my.cnf file

first verify `my.cnf` file and its basic configuration for MySQL. This configuration file is written in INI format and its location can be determined with the following command:

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



`my.cnf` setup should be similar to this one (4GB RAM Server and using an average server hardware). Make sure that you have all these parameters correctly inside secttion `[mysqld]`:

```
[mysqld]
 datadir=/var/lib/mysql
 socket=/var/lib/mysql/mysql.sock
 user=mysql
 character-set-server=utf8
 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

query_cache_type = 1
query_cache_size = 64M
query_cache_min_res_unit = 2k
query_cache_limit = 256K

sql_mode=""

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

For MySQL version 8, and later versions, support for *query cache* has been withdrawn by MySQL development team. If you wish to obtain more information you may visit the following web link: https://dev.mysql.com/blog-archive/mysql-8-0-retiring-support-for-the-query-cache/

If you are using MySQL 8 and do not have an HA environment, disable the binary logs with the following command in section `[mysqld]`:

```
skip-log-bin
```

If there is any change in `my.cnf` file, restart MySQL service.

- Verify the service status with `systemctl status mysqld.service`.
- Take a look at the end of the `/var/log/mysqld.log` for any error.
- For more information check the following link in MySQL website.

Restoring databases

To find out more about Server management and administration, go to this link.

when my.cnf file is modified, one of the most well known inconveniences consists in configuring the new values for transaction logs. If the following error appeats, back up the database and restore the previous configuration (use root user credentials):

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

1. Once the backup is created, stop MySQL service with the following command:

```
systemctl stop mysql
```

2. Go to the previous folder where MySQL data files are located (datadir) by default located at `/var/lib/mysql`:

```
cd /var/lib/
```

3. Move the folder to another location (`/var/lib/mysql` → `/var/lib/mysql_backup`)

```
mv mysql mysql_old
```

4. Create a new folder (`/var/lib/mysql`):

```
mkdir mysql
```

5. Assign the owner of the folder:

```
chown -R mysql. mysql
```

6. Initialize the folder with MySQL data:

```
mysql_install_db --datadir =/var/lib/mysql
```

7. Start MySQL service:

```
systemctl start mysql
```

If you receive the folloring error:

```
failed to retrieve rpm info for /var/lib/musql/ibdata1
```

you propably have SELinux working. You may verify if it was denied when executing the following command:

```
cat /var/log/audit/audit.log | grep /var/lib/mysql/ibdata1
```

- To disable SELinux check this section.
- If you decide to keep working with SELinux, check this other section.

8. Launch the configurator and follow the wizard:

```
mysql_secure_installation
```

You may choose to increase the security level by using complex passwords. In this case the simple password used here in the documentation (`pandora`) cannot be used.

9. Login into MySQL command line. Rebuild the database:

```
mysql> create database pandora;
```

You may need to assign again the password for the root user in MySQL. For that use this command, replacing '`pandora`' by the password you set in step numbre 8 of this same section:

```
SET PASSWORD FOR 'root'@'localhost' = PASSWORD('pandora');
```

10. Assign permissions to the right users replacing '`pandora`' by the password you set in step 8 of this same section:

```
 mysql> grant all privileges on pandora.* to pandora@'localhost' identified by
'pandora';
 mysql> grant all privileges on pandora.* to pandora@'127.0.0.1' identified by
'pandora';
```

11. Load the backups made in step 1:

```
mysql> use pandora;
mysql> source /path/to/your/backup.sql
```

Sometimes MySQL/Percona systems do not load the `my.cnf` configuration tokens correctly (usually because you put these tokens outside `[mysqld]` section)

After configuring `my.cnf` file and restarting MySQL service, check that changes were properly applied. To do that, use SHOW VARIABLES command (the result may contain more than one hundred elements and be different from the following summary)::

```
mysql> show variables like 'innodb%';
+---------------------------------------+-----------------------+
| Variable_name                         | Value                 |
+---------------------------------------+-----------------------+
| innodb_adaptive_hash_index            | ON                    |
| innodb_additional_mem_pool_size       | 1048576               |
```

```
| innodb_autoextend_increment             | 8                       |
| innodb_autoinc_lock_mode                | 1                       |
| innodb_buffer_pool_size                 | 8388608                 |
| innodb_checksums                        | ON                      |
| innodb_commit_concurrency               | 0                       |
| innodb_concurrency_tickets              | 500                     |
| innodb_data_file_path                   | ibdata1:10M:autoextend |
| innodb_data_home_dir                    |   |
| innodb_doublewrite                      | ON                      |
| innodb_fast_shutdown                    | 1                       |
| innodb_file_io_threads                  | 4                       |
| innodb_file_per_table                   | OFF                     |
| innodb_flush_log_at_trx_commit          | 1                       |
| innodb_flush_method                     |   |
| innodb_force_recovery                   | 0                       |
| innodb_lock_wait_timeout                | 50                      |
| innodb_locks_unsafe_for_binlog          | OFF                     |
| innodb_log_buffer_size                  | 1048576                 |
| innodb_log_file_size                    | 5242880                 |
| innodb_log_files_in_group               | 2                       |
| innodb_log_group_home_dir               | ./                      |
| innodb_max_dirty_pages_pct              | 90                      |
| innodb_max_purge_lag                    | 0                       |
| innodb_mirrored_log_groups              | 1                       |
| innodb_open_files                       | 300                     |
| innodb_rollback_on_timeout              | OFF                     |
| innodb_stats_method                     | nulls_equal             |
| innodb_stats_on_metadata                | ON                      |
| innodb_support_xa                       | ON                      |
| innodb_sync_spin_loops                  | 20                      |
| innodb_table_locks                      | ON                      |
| innodb_thread_concurrency               | 8                       |
| innodb_thread_sleep_delay               | 10000                   |
| innodb_use_legacy_cardinality_algorithm | ON                      |
+-----------------------------------------+-------------------------+
```

You may also check the variables one by one:

```
mysql> show variables like 'innodb_log_file_size';
mysql> show variables like 'innodb_io_capacity';
mysql> show variables like 'innodb_file_per_table';
```

```
mysql> show variables like 'innodb_log_file_size';
+----------------------+----------+
| Variable_name        | Value    |
+----------------------+----------+
| innodb_log_file_size | 67108864 |
+----------------------+----------+
1 row in set (0.00 sec)

mysql> show variables like 'innodb_io_capacity';
+--------------------+-------+
| Variable_name      | Value |
+--------------------+-------+
| innodb_io_capacity | 100   |
+--------------------+-------+
1 row in set (0.00 sec)

mysql> show variables like 'innodb_file_per_table';
+-----------------------+-------+
| Variable_name         | Value |
+-----------------------+-------+
| innodb_file_per_table | ON    |
+-----------------------+-------+
1 row in set (0.00 sec)

mysql>
```

Check if isolated datafile for each table is ACTIVE

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

There should be more than 100 files there (depending on the version of pandora), each `.ibd` is the data file of each table, when `innodb_file_per_table` parameter is enabled in file `my.cnf`. If you do not have any of these files, `.idb` means it uses a single file to store all information. The previous one means table fragmentation is also present on all tables and performance will worsen each week.

If you have your database running in a single database, first you will need to recreate the database after correctly configuring file `my.cnf` and restarting MySQL.

Check fragmentation table by table

Using MySQL CLI, execute this query:

```
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  DATA_FREE> 0 order by frag_ratio desc;
```

You should get only the tables with some fragmentation index, for example:

```
+--------+------------------------+-------------+--------------+-----------+---
---------+
```

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

This query works only on tables with more than 10% of fragmentation.

Too big tables (like `tagent_data`) can take a lot of time to get optimized if they are very fragmented. This MAY affect the production system. Therefore, it is recommended not to optimize these kinds of tables, since it could block the system (the optimization process "blocks" a table to rewrite it).

To optimize the `tagent_module_inventory` table (in this case the database is called `pandora`):

```
```

```
optimize tagent_module_inventory table;
```

A warning message will appear:

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

If you check again, you should see the fragmentation is gone:

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

To be able to perform this optimization, there must be enough space on the hard disk to perform the operation. Otherwise an error will appear and the operation will not be performed

System Load

This is more general, but you need to make sure the system IO is not a bottleneck (disk). Execute the vmstat command to get some stats from the System:

```
vmstat 1 10
```

Look at the last columns (CPU WA), a value higher than 10 means there is a disk I/O problem that should be solved.

Having CPU-US high is normal, but CPU-SY should not be over 10~15.

Usually, SWAP-SI and SWAP-SO should have value zero, if not, it means the system is using SWAP memory, which degrades performance. Increase RAM or decrease RAM usage in your applications (Pandora FMS server threads, Buffers in MySQL, etc.)

Sample output of a “normal” system:

```
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 Table Partitioning

To use MySQL table partitioning, use multiple tablespaces described above (`innodb_file_per_table`).

MySQL 5.1 supports table partitioning, which allows you to split large tables into multiple small logical sub-tables. (See MySQL manual for more details, check: hMySQL manual.

If you have large amounts of data in your Pandora FMS database (both the main one and the history one)and feel many console operations which refer to these data (e.g. drawing graph) are quite slow, improve their performance by using table partitioning.

Make sure you have innodb_file_per_table active and your database is using it: you should see in `/var/lib/mysql/pandora_history/*.ibd` a lot of files. If not, dump your database, change the `my.cnf`, restart the mysql, drop your current database, and recreate from the dump.

Once you are sure you have `innodb_file_per_table`, split your two main datatable in different partitions based on fixed dates. This example is valid to split data from year 2015, adapt it to your own needs.

This needs enough disk space. Check how big your `tagent_data.ibd` is, if it is 10G you will need at least 15GB free to start the operation.

This operation may take a long time depending on table size. As an example, it took about one and half hours to split a table which had about 7500 modules' data for 100 days (more than 50,000,000 rows):

Use this query in MySQL CLI:

```
ALTER TABLE tagente_datos PARTITION BY RANGE (utimestamp) (
PARTITION Ene15 VALUES LESS THAN (UNIX_TIMESTAMP('2015-01-01 00:00:00')),
PARTITION Feb15 VALUES LESS THAN (UNIX_TIMESTAMP('2015-02-01 00:00:00')),
PARTITION Mar15 VALUES LESS THAN (UNIX_TIMESTAMP('2015-03-01 00:00:00')),
PARTITION Apr15 VALUES LESS THAN (UNIX_TIMESTAMP('2015-04-01 00:00:00')),
PARTITION May15 VALUES LESS THAN (UNIX_TIMESTAMP('2015-05-01 00:00:00')),
PARTITION Jun15 VALUES LESS THAN (UNIX_TIMESTAMP('2015-06-01 00:00:00')),
PARTITION Jul15 VALUES LESS THAN (UNIX_TIMESTAMP('2015-07-01 00:00:00')),
PARTITION Ago15 VALUES LESS THAN (UNIX_TIMESTAMP('2015-08-01 00:00:00')),
PARTITION Sep15 VALUES LESS THAN (UNIX_TIMESTAMP('2015-09-01 00:00:00')),
PARTITION Oct15 VALUES LESS THAN (UNIX_TIMESTAMP('2015-10-01 00:00:00')),
PARTITION Nov15 VALUES LESS THAN (UNIX_TIMESTAMP('2015-11-01 00:00:00')),
PARTITION Dec15 VALUES LESS THAN (UNIX_TIMESTAMP('2015-12-01 00:00:00')),
PARTITION pActual VALUES LESS THAN (MAXVALUE)
);
```

Execute this query each month for reorganizing partitioning:

```
ALTER TABLE tagente_datos REORGANIZE PARTITION pActual INTO (
PARTITION Feb16 VALUES LESS THAN (UNIX_TIMESTAMP('2016-02-01 00:00:00')),
PARTITION pActual VALUES LESS THAN MAXVALUE);
```

Changing “Feb16” for the current month.

Remember that this operation could take hours, depending on how big your “tagente_datos” table is. You can see the progress by watching the size of the partition files executing:

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

 -rw-rw----  1 mysql mysql 424M dic 23 05:58 #sql-76b4_3f7c#P#Ago15.ibd
 -rw-rw----  1 mysql mysql 420M dic 23 05:51 #sql-76b4_3f7c#P#Apr15.ibd
 -rw-rw----  1 mysql mysql 128K dic 23 05:40 #sql-76b4_3f7c#P#Dec15.ibd
 -rw-rw----  1 mysql mysql 840M dic 23 05:44 #sql-76b4_3f7c#P#Ene15.ibd
 -rw-rw----  1 mysql mysql 440M dic 23 05:47 #sql-76b4_3f7c#P#Feb15.ibd
 -rw-rw----  1 mysql mysql  10M dic 23 05:42 #sql-76b4_3f7c#P#Jan16.ibd
 -rw-rw----  1 mysql mysql 404M dic 23 05:56 #sql-76b4_3f7c#P#Jul15.ibd
 -rw-rw----  1 mysql mysql 436M dic 23 05:54 #sql-76b4_3f7c#P#Jun15.ibd
 -rw-rw----  1 mysql mysql 400M dic 23 05:49 #sql-76b4_3f7c#P#Mar15.ibd
 -rw-rw----  1 mysql mysql 408M dic 23 05:52 #sql-76b4_3f7c#P#May15.ibd
 -rw-rw----  1 mysql mysql  72M dic 23 06:03 #sql-76b4_3f7c#P#Nov15.ibd
 -rw-rw----  1 mysql mysql 404M dic 23 06:03 #sql-76b4_3f7c#P#Oct15.ibd
 -rw-rw----  1 mysql mysql 416M dic 23 06:00 #sql-76b4_3f7c#P#Sep15.ibd
```

DDBB Rebuilding

To find out more abour Pandora FMS backup and data recovery, go to this link.

Partial Rebuilding

MySQL database management system, same as other SQL engines, such as Oracle® is degraded with time due to causes such as data fragmentation produced by deleting and continuous insertion in large tables. In large environments, with a lot traffic volume, there is a very easy way to improve the performance and prevent performance from degrading. This is rebuilding the DDBB from time to time.

To that end, schedule a service stop, which could last approximately 1 hr.

In this service stop, stop the Pandora FMS WEB console and the server too (be careful, leave the Tentacle server so

that it can still receive data and these will be processed as soon as the server works again).

Once they have been stopped, do a DDBB dump (Export); in this example, the database is called `pandora3` and the user must be root:

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

Delete the DDBB:

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

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

Create the DDBB and import the previous data export from the dump you did at first:

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

This could take several minutes, depending on whether the system is large and the hardware is not very powerful. For a system with 1500 agents and approximately 100.000 modules.

It is possible to automatize this process, but, because it is very delicate, the best option is carry it out manually.

Total Rebuilding

This section affects only InnoDB databases. Pandora FMS is built on Innodb databases.

Unfortunately, MySQL is degraded with time, and this affects the global performance of the system.There is no other solution that does not involve rebuilding all the database schemes from scratch, rebuilding the data binary file that MySQL uses to store all the information and the files used to rebuild the transactions.

If you take a look at the `/var/lib/mysql` directory, you can see that there are three files, that have always the same name, and that are, depending on the severity of the case, huge. In this example:

```
-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
```

The `ibdata1` file is the one that stores all the system Innobd data. In a very fragmented system that has not been "rebuilt" or "installed" for a long time, this system will be big but little efficient. The `innodb_file_per_table` parameter, that has been mentioned before, regulates part of this performance.

Similarly, each database has in the `/var/lib/mysql` directory, one directory to define its structure. Delete them too.

The process is quite easy:

1. Dump (via mysqldump) all the schemes to the disk:

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

- Stop MySQL.
- Delete `ibdata1`, `ib_logfile0`, `ib_logfile1` and the InnoDB database directories
- Start MySQL.
- Create `pandora` database again (`create database pandora;`)
- Import the backup file (`all.sql`)

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

The system should work faster now.

Optional Indexes

There are some situations when you can optimize MySQL performance, but giving up other system resources.

This index optimizes speed on graph rendering (a lot), but it uses more disk storage space, and could entail a slightly decrease on INSERT/DELETE operation, due to the Index overhead:

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

At the moment, in the heaviest tables of Pandora FMS in MySQL, this optimization is there by default. It is convenient to ask experts before optimizing MySQL tables.

Slow queries study

In some systems, depending on the type of information you have, you can find some “slow queries” that make the system work worse. You may enable logs of this type of queries over a short period of time (since it harms the system performance) in order to consider trying to optimize queries to tables with indexes. To enable these setings, do the following:

- Edit `my.cnf` and add the following lines:

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

- To be able to use it and set the admin rules:

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

- Restart mysql.
- When finishing analyzing which ones are the slow queries, remember to reset the file `my.cnf` commenting the aggregated lines and restarting again MySQL service.

Optimizing Specific tables

Other less “drastic” solution to solve the fragmentation issue is the use of MYSQL OPTIMIZE tool to optimize certain tables of Pandora FMS. To benefit from it, execute directly from MySQL the following:

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

This will improve the performance, and it should not be necessary to launch it more than once per week. It could be done “WHILE WARM”, while the system is working.

In very big environments, the OPTIMIZE option could be “blocked”. In this case, the best option is to rebuild the DB.

After doing these operations, execute:

```
FLUSH TABLES;
```

From the MySQL manual: *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.*

For installations after 7.0 OUM715, the following configuration is applied by default

Note: If your Pandora FMS installation was done before version 7.0 OUM 715, make the following modifications in your database (principal and historical):

```
alter table tagente_datos add index (id_agente_modulo,utimestamp);
```

This action will add an index to the *tagent_data* table, allowing queries that use both the reporting system, data query or module or custom graphs, to return results in a significantly shorter time than the previous one.

Mysql special tokens

There are some very “special” tokens in MySQL, which can improve or worsen the performance:

- innodb_thread_concurrency:

```
# Set to 0 in mysql 5.1.12 or higher
innodb_thread_concurrency           = 20
```

This parameter in versions 5.1.12 or higher, on 0 value, means there is no limit on concurrency, BUT in previous versions, the same meaning is achieved with value 20.

- innodb_flush_method:

```
innodb_flush_method = O_DIRECT
```

This important parameter has an effect on how information is written on the disk.

- innodb_lock_wait_timeout:

```
innodb_lock_wait_timeout = 90
```

This helps when there is a bottleneck, so that MySQL does not go away and stops. If it lasts more than 90 lock, there is a real problem.

External references

References:

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

MySQL Percona XTraDB

Percona is an improved version of MySQL, particularly regarding scalability (fast growth without affecting or slightly affecting work operations and routines). Make the most out of all the system's CPUs, speeding up also disk transactions.

To configure your percona server, use their excelent online configuration wizard, which will generate the */etc/my.cnf* file: Percona Wizard Configurator

Measuring Pandora FMS for High Capacity

This section describes different methods to configure Pandora FMS in a high capacity environment. It also describes different tools to make load tests, which are useful to adjust the environment to the highest possible process capacity.

Pandora FMS has been configured to support a load of around 2500 agents in systems where database, console and server are in the same machine. The recommended figure is around 2500 agents per system, but this figure varies greatly depending on whether they are XML agents, remote modules, with high or low intervals, or with systems of high capacity or low memory.

All factors greatly alter the number of agents that a system can manage efficiently. In laboratory tests, 10000 agents have been executed in a single server with basic hardware, but strongly optimized.

Example of High Capacity Servers Configuration

For example, for one machine with 16GB of RAM and 4 CPUs that you may want to optimize for the Data server maximum processing capacity (XML)

my.cnf

Only the most important parameters are shown.

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

```
character-set-server=utf8
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 = 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
wait_timeout = 900
key_buffer_size=4M
read_buffer_size=128K
read_rnd_buffer_size=128K
sort_buffer_size=128K
join_buffer_size=4M
query_cache_type = 1
query_cache_size = 64M
query_cache_min_res_unit = 2k
query_cache_limit = 256K
sql_mode=""
[mysqld_safe]
log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
```

For MySQL version 8, and later versions, the MySQL development team has withdrawn support for *query cache*, for more information please visit the following web link: https://dev.mysql.com/blog-archive/mysql-8-0-retiring-support-for-the-query-cache/ .

pandora_server.conf

Only the most important parameters are shown).

```
 verbose 3
 server_threshold 5
 dataserver_threads 1
 max_queue_files 5000
```

You should take into account the following:

- The `verbose` parameter number refers to the amount of information written in the logs, being recommended not to exceed 10. The higher the number, the worse the Pandora FMS performance will be, due to the great amount of information to write in the logs.
- A high number (15) of the parameter `server_threshold` makes the database to bear a lower impact, while the increase in the maximum number of files processed makes the server to look for files and fill up the buffers. These two elements od the setup are closely related. In the case of optimizing the network server, it would be advisable to lower `server_threshold` to 5 or 10.
- A very high number of threads (more than 5) set in dataserver_threads only benefits processes with long E/S waiting time, such as the network server or the plugin server. In the case of the dataserver, that is in constant process, may even affect performance. In systems with a slow database, use even less threads: try out different combinations between 1 and 10. In case of optimizing the system for the networkserver, the number would be a lot greater, between 10 and 30.
- Some configuration parameters could affect a lot Pandora FMS performance, such as the `agent_access` parameter (configurable from the console).

Capacity analysis Tools(Capacity)

Pandora FMS has several tools that can help you to measure properly its hardware and software for the amount of data that it expects to obtain. One of them is useful to “attack” directly the database with fictitious data (dbstress) and the other generates fictitious XML files (xml_stress).

Pandora FMS XML Stress

This is an small script that generates XML data files like the ones sent by Pandora FMS agents. It is placed on:

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

The scripts read agent names from a text file and generate XML data files for each agent according to a configuration file, where modules are defined as templates.

Modules are filled with random data. An initial value and the probability of the module data changing may be specified.

Run the script like this:

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

Sample configuration file called `pandora_xml_stress.conf`:

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

```
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, /tmp by default.
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, 1.0 by default.
xml_version 1.0

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

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

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

# Agent interval, 300 by default.
agent_interval 300

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

# Data file generation end date, now by default.
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, 2 by default.
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, 41121 by default
# 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_3
module_type generic_proc
module_descripcion Module 3 description.
# Initial data.
module_data 1
module_end
```

Send and Receive the Agent Local Configuration

If you activate in your `pandora_xml_stress.conf` the `get_and_send_agent_conf` configuration value to 1, you can make the test load agents work as normal agents, so that they send their configuration file and also the md5.

E From Pandora FMS Console Enterprise, you can change the remote configuration so that in following executions of `pandora_xml_stress`, it uses the customized configuration from the Pandora FMS Enterprise Console instead of doing it through the `pandora_xml_stress.conf` definition.

Besides this, you may configure where to store locally the `.conf` files of your testing agents with the `directory_confs` configuration token in the `pandora_xml_stress.conf` file.

Configuration File

- max_threads. Number of threads where the script will be executed. This improves the E/S.
- agent_file. Path of the name list file path, separated by a new line.
- temporal. Path of the directory where the made-up XML data files are generated.
- log_file. Path of the log where it will report about its execution script.
- xml_version. Version of the XML data file (by default 1.0).
- encoding XML data file encoding (by default ISO-8859-1).
- os_name. Name of the made-up agent Operative System (Linux by default).
- os_version. Version of the made-up agents Operative System (2.6 by default)
- agent_interval. Interval of the made-up agents in seconds (300 by default).
- time_from. Time from which made-up XML data files are generated, in format“ YEAR-MONTH-DAY HOUR:MIN:SEC”
- time_to. Time until which made-up XML data files are generated, in format YEAR-MONTH-DAY HOUR:MIN:SEC“
- get_and_send_agent_conf. Boolean value 0 or 1. When it is active the made-up agents will try to download by remote configuration a more updated version of the standard configuration file of an agent. And they can be edited through the Pandora FMS Enterprise console.
- startup_delay. Time numeric value in seconds before each agent starts to generate files. It is used to avoid race conditions.
- timezone_offset. Numeric value of the time zone offset.
- timezone_offset_range. Numeric value that is useful to generate the timezone in this range randomly.
- latitude_base. Numeric value. It is the latitude where the made-up agents will be shown.
- longitude_base Numeric value. It is the longitude where the fictitious agents will be shown.
- altitude_base Numeric value. It is the altitude where the fictitious agents will be shown.
- position_radius Numeric value. It is the range around the circumference with this radius where the fictitious agent is shown randomly.

Module Definition

The definition of one module in the script configuration file. If remote configuration has been activated, it will also be the same. It is:

```
module_begin
module_name <name of the module>
module_type <type, p.e: generic_data>
module_description <description>
module_exec type=<type_generation_xml_stress>;<other options separated by ;>
module_unit <units>
module_min_critical <value>
module_max_critical <value>
module_min_warning <value>
module_max_warning <value>
module_end
```

Each one can be configured as:

- type_generation_xml_stress: It can have the values RANDOM, SCATTER, CURVE.
- module_attenuation <value>: The generated module value is multiplied by the specified value, usually between 0.1 and 0.9.
- module_attenuation_wdays <value> <value> ... <value>: The module value is only attenuated during the given days, ranging from Sunday (0) to Saturday (6). For example, the following module

simulates a 50% drop in network traffic on Saturdays and Sundays:

```
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>: If set to one, the module's previous value is always added to a new value, resulting in an increasing function.
- Others: See below which options are available, depending on the execution type.

Note that `min_critical`, `max_critical`, `min_warning` and `max_warning` are only available in version 5.0 or later versions.

RANDOM

These have the following options:

- variation. Probability percentage of change regarding the previous value.
- min. Minimum value that the the value could have.
- max. Maximum value that the the value could have.

Numeric

It generates random numeric values between the range values min and max

Booleans

It generates values between 0 and 1.

String

It generates a length string between values min and max. The characters are random between A and Z and capital, lower case letters and also numeric ciphers are included.

External data source (SOURCE)

It allows to use a plain text file as a data source. Options:

- src: Source data file.

The file contains one data per line, there is no limit for lines. For example:

```
4
5
6
10
```

Both numbers and strings are allowed as values. These kinds of modules will use file data to generate module data in Pandora FMS. Data are retrieved sequentially. For example data above will be shown as follows:

```
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

It is only useful for numeric data, and the generated graphics are similar to the ones of a heartbeat, that is, a normal value, and from time to time a *"beat"*.

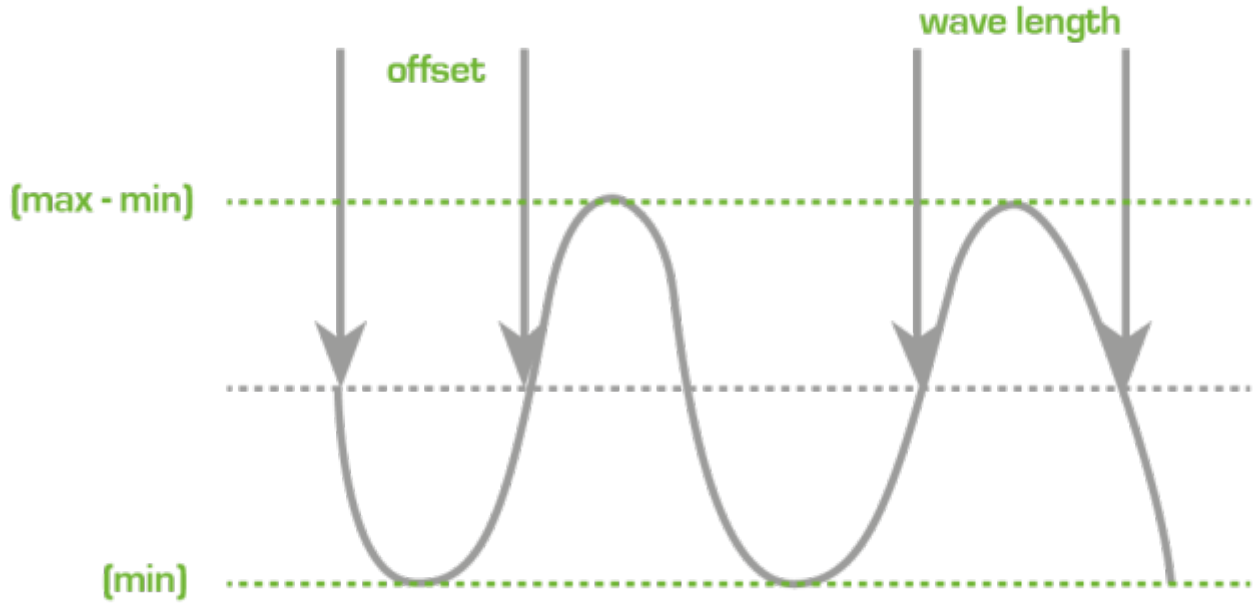
It has the following options:

- min. Minimum value that the value could have.
- max. Maximum value that the value could have.
- prob. Probability percentage that it generates a *"beat"*.
- avg. Average value that should be shown by default if there is no *"beat"*.

CURVE

It generates module data following a trigonometric curve. They have the following options:

- min. Minimum value that the value could have.
- max. Maximum value that the value could have.
- time_wave_length. Numeric value in seconds of the duration of the *"crest"* of the wave.
- time_offset. Numeric value in seconds from the starting point of the wave from time zero with module value zero (similar to the sine graph).



Interesting remarks

- This tool is preconfigured to look for, in all agents, “random”, “curve” or boolean name modules that use an interval between 300 seconds and 30 days.

How to measure Data server Processing Capacity

There is a small script called `pandora_count.sh` that is found in the `/usr/share/pandora_server/util/` directory in Pandora FMS server directory. This script is used to measure the processing rate of XML files by the data server, and it uses as reference all the files yet to be processed at `/var/spool/pandora/data_in`, so to be able to use it, thousands of packages yet to be processed are needed (or they must be generated with the tool mentioned before). Once running, you may stop it pressing the keys CTRL+C.

This script takes into account only the packages currently existing, and it take them away from the packages existing 10 seconds ago, then divides the result by 10, and these will be the files that have been processed in the last 10 seconds, showing the rate per second. It is a rudimentary solution but it is helpful to fix the server configuration.

Pandora FMS DB Stress

This is a small tool to test database performance. It could also be used to «pregenerate» periodical or random data (using trigonometric functions) and fill in made-up modules.

Create an agent and allocate the modules to that agent for automatic data injection with this tool. The names should be these ones:

- *random*: To generate random data.
- *curve*: To generate a matching curve using trigonometric functions. It is useful to use the interpolating work with different intervals, etc.
- *boolean*: To generate random boolean data.

This way, it is possible to use any name that contains the words: *«random»*, *«curve»* and/or *«boolean»*. For example:

- random_1
- curve_other

Only the “data_server” module type can be chosen.

Pandora FMS DB Stress Fine Adjustment

This tool is preconfigured in order to search, in all agents, the module names *“random”*, *“curve”* or *“boolean”*, that use an interval between 300 seconds and 30 days.

If you wish to modify this performance, edit the `pandora_dbstress` *script* and change some variables at the begining of the file:

```
# 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. The first line of the variable corresponding with `target_module` should be fixed for a fix module or `-1` to process all the matching targets.
2. The second line of variable must match `target_agent`, for a specific agent.
3. The third line must match `target_interval`, defined in seconds and which represents the module predefined periodical interval.
4. The fourth line is `target_days` and represents the number of days in the past since the date, in the current *timestamp*.

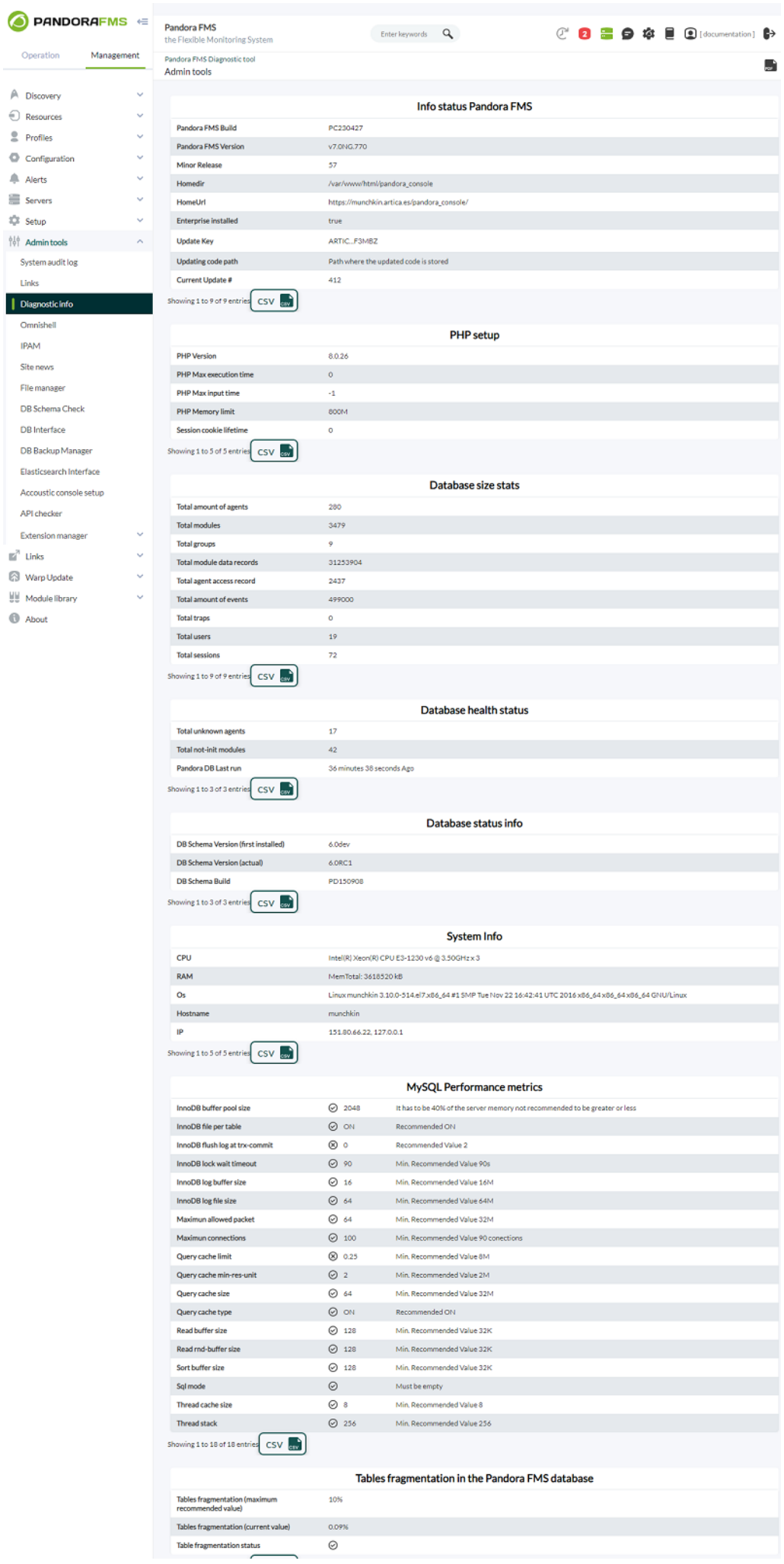
Diagnostic tools in Pandora FMS

Sometimes, users have problems and Pandora FMS developers cannot be of any help without more information about the user's systems. That is why version 3.0 includes two small tools to help solving user problems:

Diagnostic Info

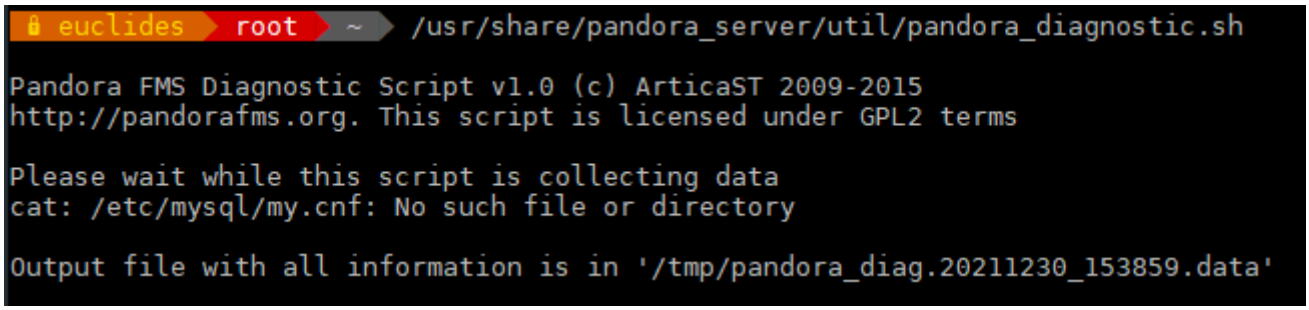
In more current versions of Pandora FMS, there is a feature to obtain diagnostic information about your Pandora FMS installation.

It is inside the section of Admin tools → Diagnostic Info


PANDORAFMS
Operation
Management
Discovery
Resources
Profiles
Configuration
Alerts
Servers
Setup
Admin tools
System audit log
Links
Diagnostic info
Omnishell
IPAM
Site news
File manager
DB Schema Check
DB Interface
DB Backup Manager
Elasticsearch Interface
Accoustic console setup
API checker
Extension manager
Links
Warp Update
Module library
About
Pandora FMS
the Flexible Monitoring System
Enter keywords
[documentation]
Pandora FMS Diagnostic tool
Admin tools
Info status Pandora FMS
Pandora FMS Build PC230427
Pandora FMS Version v7.0NG.770
Minor Release 57
Homedir /var/www/html/pandora_console
HomeUrl https://munchkin.artica.es/pandora_console/
Enterprise installed true
Update Key ARTIC...F3MBZ
Updating code path Path where the updated code is stored
Current Update # 412
Showing 1 to 9 of 9 entries
CSV
PHP setup
PHP Version 8.0.26
PHP Max execution time 0
PHP Max input time -1
PHP Memory limit 800M
Session cookie lifetime 0
Showing 1 to 5 of 5 entries
CSV
Database size stats
Total amount of agents 280
Total modules 3479
Total groups 9
Total module data records 31253904
Total agent access record 2437
Total amount of events 499000
Total traps 0
Total users 19
Total sessions 72
Showing 1 to 9 of 9 entries
CSV
Database health status
Total unknown agents 17
Total not-init modules 42
Pandora DB Last run 36 minutes 38 seconds Ago
Showing 1 to 3 of 3 entries
CSV
Database status info
DB Schema Version (first installed) 6.0dev
DB Schema Version (actual) 6.0RC1
DB Schema Build PD150908
Showing 1 to 3 of 3 entries
CSV
System Info
CPU Intel(R) Xeon(R) CPU E3-1230 v6 @ 3.50GHz x 3
RAM MemTotal: 3618520 kB
Os Linux munchkin 3.10.0-514.el7.x86_64 #1 SMP Tue Nov 22 16:42:41 UTC 2016 x86_64 x86_64 x86_64 GNU/Linux
Hostname munchkin
IP 151.80.66.22, 127.0.0.1
Showing 1 to 5 of 5 entries
CSV
MySQL Performance metrics
InnoDB buffer pool size 2048 It has to be 40% of the server memory not recommended to be greater or less
InnoDB file per table ON Recommended ON
InnoDB flush log at trx-commit 0 Recommended Value 2
InnoDB lock wait timeout 90 Min. Recommended Value 90s
InnoDB log buffer size 16 Min. Recommended Value 16M
InnoDB log file size 64 Min. Recommended Value 64M
Maximun allowed packet 64 Min. Recommended Value 32M
Maximun connections 100 Min. Recommended Value 90 conections
Query cache limit 0.25 Min. Recommended Value 8M
Query cache min-res-unit 2 Min. Recommended Value 2M
Query cache size 64 Min. Recommended Value 32M
Query cache type ON Recommended ON
Read buffer size 128 Min. Recommended Value 32K
Read rnd-buffer size 128 Min. Recommended Value 32K
Sort buffer size 128 Min. Recommended Value 32K
Sql mode Must be empty
Thread cache size 8 Min. Recommended Value 8
Thread stack 256 Min. Recommended Value 256
Showing 1 to 18 of 18 entries
CSV
Tables fragmentation in the Pandora FMS database
Tables fragmentation (maximum recommended value) 10%
Tables fragmentation (current value) 0.09%
Table fragmentation status

In this window, information about Pandora FMS and MySQL configuration is displayed, as well as self-monitoring system graphs.

pandora_diagnostic.sh



It is a tool located at `/usr/share/pandora_server/util` and it provides a lot of information about the system:

- CPU information.
- Uptime and CPU avgload.
- Memory information.
- Kernel/Release information.
- A mysql configuration dump file.
- A Pandora FMS Server configuration dump file (filtering passwords).
- Pandora FMS log information (but not the full log!).
- Disk information.
- Pandora FMS process information.
- Full kernel log information (dmesg).

All information is generated in a `.txt` file, so users can send this information to anyone who wants to help them, for example, in Pandora FMS user forums or in Pandora FMS public mailing lists. This file should not contain any kind of confidential information. Bear in mind that you might want to run with root privileges if you want to get `pandora_server.conf` and `my.cnf` files parsed.

This is an execution example:

```
$ ./pandora_diagnostic.sh

Pandora FMS Diagnostic Script v1.0 (c) ArticaST 2009
http://pandorafms.org. This script is licensed under GPL2 terms

Please wait while this script is collecting data

Output file with all information is in '/tmp/pandora_diag.20090601_164511.data'
```

And here there are some parts of file output:

```
Information gathered at 20090601_164511
Linux raz0r 2.6.28-12-generic #43-Ubuntu SMP Fri May 1 19:27:06 UTC 2009 i686
```

```
GNU/Linux
================================================================================
----------------------------------------------------------------
CPUINFO
----------------------------------------------------------------
processor    : 0
vendor_id    : GenuineIntel
cpu family    : 6
.
.
----------------------------------------------------------------
Other System Parameters
----------------------------------------------------------------
Uptime:  16:45:11 up  5:27,  2 users,  load average: 0.11, 0.12, 0.09
----------------------------------------------------------------
PROC INFO (Pandora)
----------------------------------------------------------------
slerena  11875  0.9  2.1 114436 44336 pts/0    Sl   13:14   1:56 gedit
pandora_diagnostic.sh
slerena  24357  0.0  0.0   4452  1524 pts/0    S+   16:45   0:00 /bin/bash
./pandora_diagnostic.sh
----------------------------------------------------------------
MySQL Configuration file
----------------------------------------------------------------
#
# The MySQL database server configuration file.
#
# You can copy this to one of:
# - "/etc/mysql/my.cnf" to set global options,
.
.
.
----------------------------------------------------------------
Pandora FMS Logfiles information
----------------------------------------------------------------
total 3032
drwxr-xrwx  2 root    root       4096 2009-04-30 20:00 .
drwxr-xr-x 17 root    root       4096 2009-06-01 11:24 ..
-rw-r-----  1 root    sys      377322 2009-04-06 00:12 pandora_agent.log
-rw-r--r--  1 root    root          0 2009-04-06 00:15 pandora_agent.log.err
-rw-r--r--  1 root    root      13945 2009-04-02 21:47 pandora_alert.log
-rw-r--r--  1 slerena slerena 2595426 2009-04-30 20:02 pandora_server.error
-rw-rw-rw-  1 root    root       9898 2009-04-30 20:02 pandora_server.log
-rw-rw-rw-  1 root    root      65542 2009-04-30 20:00 pandora_server.log.old
-rw-r--r--  1 root    root         94 2009-04-06 00:19 pandora_snmptrap.log
-rw-rw-rw-  1 root    root          4 2009-04-03 14:16
pandora_snmptrap.log.index
----------------------------------------------------------------
System disk
----------------------------------------------------------------
S.ficheros            Tamaño Usado  Disp Uso% Montado en
/dev/sda6              91G   49G   37G  58% /
```

```
tmpfs                  1003M     0 1003M   0% /lib/init/rw
varrun                 1003M  260K 1002M   1% /var/run
varlock                1003M     0 1003M   0% /var/lock
udev                   1003M  184K 1002M   1% /dev
tmpfs                  1003M  480K 1002M   1% /dev/shm
lrm                    1003M  2,4M 1000M   1% /lib/modules/2.6.28-12-
generic/volatile
------------------------------------------------------------------------
Vmstat (5 execs)
------------------------------------------------------------------------
procs -----------memory---------- ---swap-- -----io---- -system-- ----cpu----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa
 2  0      0 684840 119888 619624    0    0    15    10  258  474  3  1 95  0
 0  0      0 684768 119888 619640    0    0     0     0  265  391  0  0 100  0
 0  0      0 684768 119892 619636    0    0     0    56  249  325  1  1 99  0
 0  0      0 684768 119892 619640    0    0     0     0  329  580  0  0 100  0
 0  0      0 684776 119892 619640    0    0     0     0  385 1382  1  0 99  0
------------------------------------------------------------------------
System dmesg
------------------------------------------------------------------------
[    0.000000] BIOS EBDA/lowmem at: 0009f000/0009f000
[    0.000000] Initializing cgroup subsys cpuset
[    0.000000] Initializing cgroup subsys cpu
[    0.000000] Linux version 2.6.28-12-generic (buildd@rothera) (gcc version
4.3.3 (Ubuntu 4.3.3-5ubuntu4) )   #43-Ubuntu SMP Fri May 1
19:27:06 UTC 2009 (Ubuntu 2.6.28-12.43-generic)
.
.
------------------------------------------------------------------------
END OF FILE
------------------------------------------------------------------------
560e8fa02818916d4abb59bb50d91f6a  /tmp/pandora_diag.20090601_164511.data
```

Go back to Pandora FMS documentation index