Optimización y solución de problemas

Optimización y solución de problemas

El servidor de Pandora FMS es capaz de monitorizar unos 2000 dispositivos (entre 5 mil y 80 mil módulos, dependiendo del hardware disponible); para ello es necesario, además, *afinar la configuración de la base de datos* (BBDD).

Adicionalmente, en este capítulo, se explican algunas técnicas para detectar y solucionar problemas en la instalación de Pandora FMS.

Optimización MySQL para la versión Enterprise

Para conocer más acerca del "Respaldo y recuperación de datos en Pandora FMS", diríjase a este enlace.

Consejos generales

- A menos que se especifique otra cosa, todo este tema se refiere a MySQL versión 5.7 .
- Véase también "Actualización de MySQL 5.7 a MySQL 8".

Para trabajar con tablas mayores de 2 GiB es necesario seguir algunas pautas:

- MySQL® recomienda usar un sistema de 64 bits. Sistemas de 32 bits podrán tener serios problemas a partir del año 2038 debido al desbordamiento de las variables de fecha.
- A más memoria RAM y más CPU, mejor rendimiento. Acorde con nuestra experiencia, la memoria RAM es más importante que la CPU. El mínimo para un sistema a nivel de empresa será 4 GiB. Una buena opción para un gran sistema son 16 GiB. Recuerde que más memoria RAM puede acelerar las actualizaciones clave mediante el mantenimiento de las páginas clave más usadas en la RAM.
- Es buena idea ser capaz de retirar el sistema en caso de fallo. Para sistemas donde la base de datos está en otro servidor dedicado para la base de datos utilice Ethernet Gigabit, preferiblemente con fibra óptica en vez de cobre. La latencia es tan importante como el rendimiento.
- La optimización del disco es muy importante para bases de datos muy grandes: habrá que partir las bases de datos y las tablas en diferentes discos. En MySQL® se pueden usar enlaces simbólicos para ello. Utilice diferentes discos para el sistema y la base de datos.

Se recomienda el uso de almacenamiento SSD debido a su rapidez y la mejora de latencia en el sistema.

- Si es posible use:

```
--skip-locking
```

- ya que lo anterior apagará el bloqueo externo y proporcionará un mejor rendimiento (activado de forma predeterminada en algunos sistemas).
- Si el cliente y el servidor MySQL® están en la misma máquina, utilice *sockets* en lugar de conexiones TCP/IP al conectar con MySQL® (esto puede dar una mejora del 7,5%). Puede hacer esto sin especificar el nombre del anfitrión o el *localhost* al conectar al servidor MySQL®. Deshabilite el inicio de sesión binario y la *replicación* si tiene solo un servidor anfitrión MySQL®.
- Utilizar versiones recientes de MySQL® (5.5 o posteriores) respecto a versiones más antiguas de MySQL® (5.0.x) pueden ofrecer hasta una diferencia del 20% en rendimiento.
- Se recomienda el uso de la versión modificada de MySQL® (Percona Server for MySQL®), la cual ofrece un mayor rendimiento. Por defecto los *plugins* programados son para Percona®.

Tenga en cuenta que afecta mucho al rendimiento los siguientes puntos:

- Solamente utilice *logs* binarios si utiliza una configuracion de MySQL® con replicación.
- No utilice *logs* de trazabilidad de *queries* o *slowquery logs*.

Herramientas automáticas de configuración

Existen bastantes herramientas para optimizar la configuración del servidor MySQL.

MySQL Tuning Primer, de Mattew Montgomery, es una herramienta de línea de comando para verificar la configuración de un servidor y ver su rendimiento y posibles mejoras, dando algunas pistas y sugerencias para mejorarlo. Está en https://bugs.launchpad.net/mysql-tuning-primer

Desactivar replicación binaria

Si tiene configurado un sistema HA de Pandora FMS la replicación binaria es necesaria. Esta recomendación solamente es válida si tiene un servidor único de Pandora FMS.

Por defecto viene habilitada en la mayoría de distros de GNU/Linux. Para desactivarla, edite el fichero `my.cnf`, habitualmente en `/etc/my.cnf` y comente las siguientes lineas:

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

Hay que comentar las dos líneas, y luego reinicie el servidor MySQL®.

Rendimiento de acceso a disco

Para conocer más acerca del “Respaldo y recuperación de datos en Pandora FMS”, diríjase a este enlace.

Junto con otros parámetros clave, hay tres que son especialmente relevantes en cuanto a rendimiento de acceso a disco, que suele ser el cuello de botella respecto a MySQL.

innodb_log_file_size

```
innodb_log_file_size = 64M
```

Por defecto se establece este valor, que puede ser mayor y llegar incluso a 512 Megabytes sin perjuicio alguno (*excepto para recuperación en caso de problema ya que tiene una mayor ocupación de disco*). El valor por defecto de MySQL es de 5M, el cual es muy bajo para entornos de producción con gran volumen de transacciones. Para alterar este valor con un sistema ya en funcionamiento:

1. Primero deberá hacer un DUMP completo de las bases de datos.
2. Borrar los ficheros índices binarios de Innodb (generalmente en `/var/lib/mysql/ib*`).
3. Cambiar el fichero `my.cnf` con el valor escogido.
4. Reiniciar el MySQL.
5. Cargar el DUMP de SQL.

Dado que el proceso es el mismo que hay que hacer para activar el token `innodb_file_per_table` (descrito un poco más abajo), se recomienda hacer todo el proceso simultáneamente.

innodb_io_capacity

```
innodb_io_capacity = 100
```

Por defecto, este parámetro tiene el valor 100, pero debe conocer previamente los IOPS del disco del sistema. Se puede conocer exactamente buscando IOPS y el modelo exacto del disco duro (obtenido via smartctl), donde los valores recomendados son: 7500RPM → 100 IOPS, 15000 RPM → 190 IOPS, SSD → 1500 IOPS

Para instalar smarctl debe solicitar el paquete completo smartmontools; por ejemplo:

`yum install smartmontools`,

`apt install smartmontools`, etcétera.

innodb_file_per_table

Usar un espacio de tablas para cada tabla (tomado del manual de MySQL 5.0):

En MySQL 5.0, se puede almacenar cada tabla InnoDB y sus índices en su propio fichero. Esta característica se llama *multiple tablespaces* (espacios de tablas múltiples) porque, en efecto, cada tabla tiene su propio espacio de tablas.

El uso de múltiples espacios de tablas puede ser beneficioso para usuarios que desean mover tablas específicas a discos físicos separados o quienes deseen restaurar respaldos de tablas sin interrumpir el uso de las demás tablas InnoDB.

Se pueden habilitar múltiples espacios de tablas agregando esta línea a la sección `mysqld` del fichero `my.cnf`:

```
[mysqld]
innodb_file_per_table
```

Después de reiniciar el servidor, InnoDB almacenará cada nueva tabla creada en su propio fichero `nombre_tabla.ibd` en el directorio de la base de datos a la que pertenece la tabla. Esto es similar a lo que hace el motor de almacenamiento MyISAM, pero MyISAM divide la tabla en un fichero de datos `tbl_name.MYD` y el fichero de índice `tbl_name.MYI`.

Para InnoDB, los datos y los índices se almacenan juntos en el fichero `.ibd`. El fichero `tbl_name.frm` se sigue creando como es usual. Si se quita la línea innodb_file_per_table del archivo `my.cnf` y se reinicia el servidor (ver instruciones anteriores para innodb_log_file_size), InnoDB creará nuevamente las tablas dentro de los ficheros del espacio de tablas compartido.

El parámetro innodb_file_per_table afecta solamente a la creación de tablas. Si se inicia el servidor con esta opción, las tablas nuevas se crearán empleando ficheros `.ibd`, pero aún se puede acceder a las tablas existentes en el espacio de tablas compartido. Si se remueve la opción, las nuevas tablas se crearán en el espacio compartido, pero aún se podrá acceder a las tablas creadas en espacios de tablas múltiples.

Evitando escritura con cada transacción

MySQL por defecto establece `autocommit = 1` para cada conexión. Lo cual es inocuo para MyISAM, ya que lo que se escribe no está garantizado en el disco, pero para InnoDB significa que cada insert / update / delete en una tabla InnoDB se traducirá en una escritura en el disco (*flush*).

¿Qué tiene de malo que escriba en el disco? Nada en absoluto. Se aseguran de que ante cualquier compromiso se garantiza que el dato esté allí cuando se reinicie la base de datos después de un accidente. El problema es que el funcionamiento de la BBDD está limitado por la velocidad física del disco.

Dado que el disco tiene que escribir los datos antes de la confirmación de la escritura, esto toma su tiempo. Suponiendo incluso un tiempo medio de búsqueda de 9 milisegundos por la escritura en disco, esto limita a aproximadamente 67 *commits* por segundo, esto es muy lento. Y mientras el disco está ocupado tratando de que el sector sea escrito, no está haciendo lecturas.

InnoDB puede evitar parte de esta limitación realizando algunas escrituras juntas, *pero aún así la limitación existe*. Se puede evitar que escriba al final de cada transacción, haciendo que ponga un sistema "automático" de escritura, que escribe aproximadamente cada segundo. En caso de fallo, puede que se pierda los datos del último segundo, algo más que asumible si se trata de ganar eficiencia. Para ello, usaremos el siguente token de configuración: `innodb_flush_log_at_trx_commit = 0` . Por defecto viene este valor en la configuración.

Mayor tamaño del KeyBuffer

Dependiendo de la RAM total del sistema, es un parámetro global muy importante que acelera las instrucciones DELETES e INSERT.

```
key_buffer_size = 4M
```

Este es el valor que viene por defecto en la configuración.

Otros parámetros importantes

Hay varios *buffer* que por defecto, en algunas distribuciones, faltan o están *comentados*. Modificar y/o agregar estos parámetros puede dar un rendimiento muy superior al que se obtiene por defecto. Es importante asegurarse de que existen estos *tokens*, con los siguientes valores, en el fichero de configuración de MySQL:

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

Para MySQL versión 8, y versiones posteriores, el equipo de desarrollo de MySQL ha retirado el soporte para *query cache*, si desea obtener más información puede visitar el siguiente enlace web (en inglés): https://dev.mysql.com/blog-archive/mysql-8-0-retiring-support-for-the-query-cache/

Mejorando la concurrencia de InnoDB

Existe un parámetro que puede afectar bastante al rendimiento del servidor MySQL con Pandora

FMS. Este parámetro es `innodb_thread_concurrency`. Dicho parámetro sirve para especificar cuántos "*hilos concurrentes*" puede ejecutar MySQL. Una mala configuración de este parámetro puede hacer que vaya más lento que por defecto, por lo que es especialmente importante prestar atención a varios aspectos:

- Versión de MySQL. En diferentes versiones de MySQL este parámetro se comporta muy diferente.
- Número de procesadores reales (físicos).

Al respecto, puede acceder a la documentacion oficial de MySQL.

El valor recomendado es el número de CPU (físicas) multiplicado por 2 más el número de discos donde se ubica InnoDB.

En las últimas versiones de MySQL (> 5.0.21) el valor por defecto es 8. Un valor 0 significaría que "abra tantos hilos como le sea posible". Por lo tanto, si hay dudas, se puede usar:

```
innodb_thread_concurrency = 0
```

Diferentes personas ("Variable's Day Out #5: innodb_thread_concurrency", "Do we still need innodb_thread_concurrency?") han hecho pruebas y han detectado problemas con el rendimiento en servidores con muchas CPU físicas cuando se usa un número muy alto, con versiones de MySQL antiguas (del año 2008).

Fragmentación MySQL

Al igual que los filesystems, las bases de datos tambien se fragmentan, haciendo que todo el sistema pierda rendimiento. En un sistema de alto rendimiento, como Pandora FMS es vital que la salud de la BBDD no afecte al funcionamiento del sistema. En sistemas sobrecargados, al final la BBDD puede bloquearse, resultando una caída de todo el sistema.

Una buena configuración de MySQL podría hacer que Pandora FMS trabajase mas rápido. Si tiene problemas de rendimiento probablemente será porque no tiene MySQL correctamente configurado o por algún problema relacionado con la base de datos.

Comprobación del fichero my.cnf

Primero debe verificar el fichero `my.cnf` y su configuración básica para MySQL. Dicho archivo de configuración está escrito en formato INI y su ubicación puede ser determinada con el siguiente comando:

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

La configuración de `my.cnf` debería ser similar a esta (4 GB de RAM y usando un hardware con una configuración media). Compruebe que tiene todos estos parámetros correctamente dentro en la sección `[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
```

Para MySQL versión 8, y versiones posteriores, el equipo de desarrollo de MySQL ha retirado el soporte para *query cache*, si desea obtener más información puede visitar el siguiente enlace web (en inglés): https://dev.mysql.com/blog-archive/mysql-8-0-retiring-support-for-the-query-cache/

Si utiliza MySQL 8 y no tiene un entorno con HA, deshabilite los *log* binarios con la siguiente instrucción en la sección `[mysqld]`:

```
skip-log-bin
```

Con cualquier cambio que haga en el fichero `my.cnf` deberá reiniciar el servicio MySQL.

- Verifique el estado del servicio con `systemctl status mysqld.service`.
- Compruebe al final del fichero `/var/log/mysqld.log` para ver si ha ocurrido algún error.
- Para más información consulte el siguiente enlace "The Error Log" en el sitio web de MySQL.

Reconstruir bases de datos

Para conocer más acerca de la "*Gestión y administración de los servidores*", diríjase a este enlace.

Cuando es modificado el fichero `my.cnf` sucede uno de los inconvenientes más conocidos el cual consiste en configurar los nuevos valores para los *registros de transacciones*. Si aparece el siguiente error deberá realizar un respaldo de la base de datos y restaurar la configuración previa (deberá usar credenciales de usuario raíz o *root user*):

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

1. Una vez haya creado el respaldo de la base de datos, detenga el servicio MySQL con el siguiente comando:

```
systemctl stop mysql
```

2. Vaya a la carpeta previa donde se encuentran los ficheros de datos de MySQL (datadir), por defecto ubicado en `/var/lib/mysql`:

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

3. Mueva la carpeta a otra ubicación (de `/var/lib/mysql` → `/var/lib/mysql_backup`):

```
mv mysql mysql_backup
```

4. Cree una nueva carpeta (`/var/lib/mysql`):

```
mkdir mysql
```

5. Asigne el propietario de la carpeta:

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

6. Inicialice la carpeta con los datos de MySQL:

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

7. Inicie el servicio MySQL:

```
systemctl start mysql
```

Si recibe el siguiente error:

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

probablemente tenga funcionando SELinux. Usted puede confirmar si se ha producido una denegación (*denied*) al ejecutar el siguiente comando:

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

- Para desactivar SELinux consulte esta sección.
- Si decide seguir trabajando con SELinux consulte esta otra sección.

8. Lance el configurador y siga el asistente:

```
mysql_secure_installation
```

Puede seleccionar elevar el nivel de seguridad usando contraseñas complejas. En ese caso la simple contraseña de ejemplo utilizada aquí en la documentación (`pandora`) no podrá ser utilizada.

9. Inicie sesión en la línea de comandos de MySQL. Reconstruya la base de datos:

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

Tal vez necesite asignar de nuevo la contraseña para el usuario root en MySQL. Para ello utilice este comando, *sustituyendo* `'pandora'` *por la contraseña que haya establecido en el paso número 8 de esta misma sección*:

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

10. Asigne los permisos a los usuarios correctos, *sustituyendo* `'pandora'` *por la contraseña que haya establecido en el paso número 8 de esta misma sección*:

```
 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. Cargue el respaldo realizado en el paso 1:

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

Muchas veces los sistemas con MySQL/Percona no cargan correctamente los parámetros de configuración del fichero `my.cnf`, generalmente porque estos valores han sido escritos fuera de la sección `[mysqld]` .

Después de haber configurado el fichero `my.cnf` y reiniciado el servicio MySQL, deberá comprobar que estos cambios han sido correctamente aplicados. Para hacer esto use el comando SHOW VARIABLES (el resultado puede contener más de cien elementos y diferir del siguiente resumen):

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

También puede consultar las variables una por una:

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

Comprobación de que los ficheros de datos aislados para cada tabla están ACTIVADOS

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

Debería tener más de 100 ficheros (dependiendo de la versión de Pandora FMS). Cada `.ibd` será el fichero de datos para cada tabla, cuando tiene activado el parámetro `innodb_file_per_table` en el fichero `my.cnf`. Si no tuviese ninguno de estos ficheros `.ibd` significará que utiliza un único archivo para almacenar toda la información. Lo anterior implica que la fragmentación de las tablas es común al resto de tablas y esto podría anticipar que el rendimiento será peor cada semana que pase.

Si tiene su base de datos corriendo bajo una única base de datos, necesitará en primer lugar recrear la base de datos después de haber configurado correctamente el fichero `my.cnf` y reiniciar MySQL.

Comprobación de la fragmentación tabla por tabla

Usando el CLI de MySQL, deberá ejecutar esta consulta:

```
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;
```

Debería ver solamente las tablas con algún indice de fragmentación, por ejemplo:

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

Esta consulta funcionará solamente en las tablas que posean mas de un 10% de fragmentación.

Las tablas demasiado grandes (como `tagente_datos`) podrían tardar demasiado tiempo en optimizarse si están muy fragmentadas. Esto podría causar algún tipo de impacto en los sistemas de producción. Por lo que recomendamos no optimizar este tipo de tablas tan grandes, ya que podría causar un bloqueo del sistema (el proceso de optimización “bloquea” la tabla para reescribirla)

Para optimizar la tabla `tagent_module_inventory` (en este ejemplo la base de datos es llamada `pandora`)

```
optimize table pandora.tagent_module_inventory;
```

Aparecerá un mensaje:

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

Ahora, si comprueba de nuevo la fragmentación de las tablas, puede ver que la fragmentación ha desaparecido:

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

Para poder realizar esta optimización será imprescindible tener el espacio necesario en el disco duro para realizar la operación. En caso contrario saldrá un error y no se realizará la operación.

Carga del sistema

Esto es de ámbito general, pero debe estar seguro de que el sistema de E/S (entrada y salida) no es un cuello de botella (escritura y lectura en dispositivos de almacenamientos/discos). Necesita ejecutar el siguiente comando para recoger la información del sistema:

```
vmstat 1 10
```

Ahora, observe en las últimas columnas (CPU-WA): un valor por encima de 10 significará que existe un problema de E/S que deberá ser arreglado.

Tener unos valores de CPU-US mayoras a cero es normal, pero CPU-SY no deberá ser mayor de 10-15.

En condiciones normales debería tener SWAP-SI y SWAP-SO con valores en 0, de no ser así significa que el sistema está usando memoria SWAP, considerado como un destructor de rendimiento. Necesitará incrementar la memoria RAM o decrementar la memoria RAM usada en las aplicaciones (hilos de Pandora FMS, buffers de MySQL, etc).

Le mostramos la salida de ejemplo de un *sistema normal*:

```
procs -----------memory---------- ---swap-- -----io---- --system-- -----cpu-----
-
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id  wa
st
 0  0  46248  78664 154644 576800    0    0     2   147    0    9  7 10 83  0  0
 0  0  46248  78656 154644 576808    0    0     0     0   49   37  0  0 100 0  0
 2  0  46248  78904 154648 576740    0    0     0   184  728 2484 63  6 31  0  0
 0  0  46248  79028 154648 576736    0    0    16   616  363  979 21  0 79  0  0
 1  0  46248  79028 154648 576736    0    0     0    20   35   37  0  1 98  1  0
```

```
 0  0  46248  79028 154648 576736    0    0     0     0   28   22  0  0 100 0  0
 1  0  46248  79028 154648 576736    0    0     0  3852  141  303  0  0 98  2  0
 2  0  46248  78904 154660 576660    0    0     0   188  642 2354 56  4 40  0  0
 1  0  46248  78904 154660 576680    0    0     0    88  190  634 13  0 86  1  0
 1  0  46248  78904 154660 576680    0    0     0    16   35   40  0  0 100 0  0
 1  0  46248  78904 154660 576680    0    0     0     0   26   21  0  0 100 0  0
 0  0  46248  78904 154660 576680    0    0     0     0   27   27  0  0 100 0  0
 1  0  46248  78904 154724 576616    0    0   112   192  608 2214 52  4 44  0  0
 0  0  46248  78904 154724 576616    0    0     0    76  236  771 16  0 84  0  0
 0  0  46248  78904 154724 576616    0    0     0    20   38   38  0  0 100 0  0
 0  0  46248  78904 154724 576616    0    0     0     0   31   21  0  0 100 0  0
 0  0  46248  78904 154740 576608    0    0     0  3192  187  322  1  0 96  3  0
 1  0  46248  79028 154756 576544    0    0    16   192  632 2087 53  5 42  0  0
 0  0  46248  79028 154760 576568    0    0     0    56  255  927 19  2 79  0  0
 0  0  46248  79028 154768 576564    0    0     0    20   33   44  0  0 100 0  0
```

Particionado de tablas en MySQL

Para usar Particionado de tablas MySQL, debería usar también el sistema de *multiples tablespaces* descrito arriba (`innodb_file_per_table`).

MySQL 5.1 soporta particionado de tablas, lo que permite distribuir tablas muy grandes en trozos más pequeños, como subdivisiones lógicas (puede consultar el manual de MySQL para más detalles.)

Si tiene grandes cantidades de datos en la base de datos (principal e histórica) Pandora FMS y estima que muchas operaciones de la Consola que se refieren a esos datos (por ejemplo *drawing graph*) son lentas, entonces mejorará su rendimiento utilizando particionado de tablas.

Debe comprobar en un primer momento que el directorio `/var/lib/mysql/pandora_history/*.ibd` tiene muchos ficheros (uno por tabla). Si no es así, necesitará hacer un *dump* de la base de datos, cambiar la configuración del fichero `my.cnf`, reiniciar MySQL, borrar la base de datos actual y recrearla desde el *dump*.

Una vez que se haya asegurado que `innodb_file_per_table` está activado, separe las dos bases de datos principales en diferentes particiones. Éste es un ejemplo para hacer la partición de todo el 2021 hasta ahora y para futuros meses.

Esta operación necesitará espacio suficiente en el disco para se completada. Habrá que comprobar qué tan grande es el archivo `tagente_datos.ibd`. Si por ejemplo ocupa 10 gigabytes, necesitará 15 GB de espacio libre para empezar la operación.

Esta operación puede llevar mucho tiempo, dependiendo del tamaño de la tabla. Por ejemplo, llevaría una hora y media partir una tabla con aproximadamente 7500 *modules data* para 100 días

(más de 50.000.000 filas).

Para comenzar el proceso deberá ejecutar la siguiente consulta en el CLI de MySQL:

```
ALTER TABLE tagente_datos PARTITION BY RANGE (utimestamp) (
PARTITION Ene15 VALUES LESS THAN (UNIX_TIMESTAMP('2021-01-01 00:00:00')),
PARTITION Feb15 VALUES LESS THAN (UNIX_TIMESTAMP('2021-02-01 00:00:00')),
PARTITION Mar15 VALUES LESS THAN (UNIX_TIMESTAMP('2021-03-01 00:00:00')),
PARTITION Apr15 VALUES LESS THAN (UNIX_TIMESTAMP('2021-04-01 00:00:00')),
PARTITION May15 VALUES LESS THAN (UNIX_TIMESTAMP('2021-05-01 00:00:00')),
PARTITION Jun15 VALUES LESS THAN (UNIX_TIMESTAMP('2021-06-01 00:00:00')),
PARTITION Jul15 VALUES LESS THAN (UNIX_TIMESTAMP('2021-07-01 00:00:00')),
PARTITION Ago15 VALUES LESS THAN (UNIX_TIMESTAMP('2021-08-01 00:00:00')),
PARTITION Sep15 VALUES LESS THAN (UNIX_TIMESTAMP('2021-09-01 00:00:00')),
PARTITION Oct15 VALUES LESS THAN (UNIX_TIMESTAMP('2021-10-01 00:00:00')),
PARTITION Nov15 VALUES LESS THAN (UNIX_TIMESTAMP('2021-11-01 00:00:00')),
PARTITION Dec15 VALUES LESS THAN (UNIX_TIMESTAMP('2021-12-01 00:00:00')),
PARTITION pActual VALUES LESS THAN (MAXVALUE)
);
```

Luego habría que ejecutar cada mes la siguiente consulta para reorganizar el particionamiento:

```
ALTER TABLE tagente_datos REORGANIZE PARTITION pActual INTO (
PARTITION Feb2022 VALUES LESS THAN (UNIX_TIMESTAMP('2022-02-01 00:00:00')),
PARTITION pActual VALUES LESS THAN MAXVALUE);
```

Cambiando “Feb2022” por el mes en el que se encuentre.

Recuerde de nuevo que esta operación podría tardar horas, dependiendo de lo grande que sea la tabla `tagente_datos`. Puede ir comprobando el proceso viendo el tamaño de los archivos de particionamiento ejecutando:

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

 -rw-rw----  1 mysql mysql 424M dic 23 05:58 #sql-76b4_3f7c#P#Ago21.ibd
 -rw-rw----  1 mysql mysql 420M dic 23 05:51 #sql-76b4_3f7c#P#Apr21.ibd
 -rw-rw----  1 mysql mysql 128K dic 23 05:40 #sql-76b4_3f7c#P#Dec21.ibd
 -rw-rw----  1 mysql mysql 840M dic 23 05:44 #sql-76b4_3f7c#P#Ene21.ibd
 -rw-rw----  1 mysql mysql 440M dic 23 05:47 #sql-76b4_3f7c#P#Feb21.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#Jul21.ibd
 -rw-rw----  1 mysql mysql 436M dic 23 05:54 #sql-76b4_3f7c#P#Jun21.ibd
 -rw-rw----  1 mysql mysql 400M dic 23 05:49 #sql-76b4_3f7c#P#Mar21.ibd
 -rw-rw----  1 mysql mysql 408M dic 23 05:52 #sql-76b4_3f7c#P#May21.ibd
 -rw-rw----  1 mysql mysql  72M dic 23 06:03 #sql-76b4_3f7c#P#Nov21.ibd
 -rw-rw----  1 mysql mysql 404M dic 23 06:03 #sql-76b4_3f7c#P#Oct21.ibd
 -rw-rw----  1 mysql mysql 416M dic 23 06:00 #sql-76b4_3f7c#P#Sep21.ibd
```

Reconstruccion de la BBDD

Para conocer más acerca del “Respaldo y recuperación de datos en Pandora FMS”, diríjase a este enlace.

Reconstruccion parcial

El sistema de gestión de base de datos de MySQL, al igual que otros motores SQL, como Oracle®, se degrada con el tiempo por causas como la fragmentación de datos producida por el borrado y la inserción continuada en grandes tablas. En grandes entornos con mucho volumen de tráfico existe una forma muy sencilla de mejorar el rendimiento e impedir que el rendimiento se degrade: reconstruir la BBDD de forma periódica.

Para ello hay que programar una parada de servicio, que puede durar aproximadamente 1 hora.

En esta parada de servicio debe usted detener la Consola web PFMS y el servidor PFMS. Atención: dejar el servidor Tentacle funcionando para que siga recibiendo datos, y esos se procesarán tan pronto como el servidor esté operativo de nuevo.

Una vez detenidos, realizar un *volcado* de la BBDD (*Export*); en este ejemplo la base de datos es llamada `pandora3` y el usuario debe ser root:

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

Ahora realice el borrado de la BBDD:

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

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

El siguiente paso es recrear la BBDD y seguidamente hacer una importación de datos desde el *dump* realizado al principio:

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

Esto puede tardar unos cuantos minutos,dependiendo de si el sistema es grande y el hardware de baja potencia, para un sistema con 1500 agentes y aproximadamente 100.000 módulos.

Se puede automatizar este proceso, pero por su naturaleza
delicada, es mejor hacerlo manualmente.

Reconstrucción total

Este capítulo solo afecta a bases de datos InnoDB. Pandora FMS está construido sobre bases de datos InnoDB.

MySQL se degrada con el tiempo, cosa que afecta a todo el rendimiento del sistema. La única solución es reconstruir todos los esquemas de base de datos desde cero, reconstruyendo el fichero binario de datos que MySQL usa para almacenar toda la información y los ficheros usados para reconstruir las transacciones.

Si observa el directorio `/var/lib/mysql` encontrará que hay tres ficheros que siempre se llaman igual y son, en función de lo grave del caso, *gigantes*. Por ejemplo:

```
-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
```

El fichero `ibdata1` es el que alberga todos los datos InnoDB del sistema. En un sistema muy fragmentado, que lleva mucho tiempo sin "rehacer" o sin "reinstalar" este sistema será grande y poco eficiente. El parámetro `innodb_file_per_table` que se describe antes, regula parte de este comportamiento.

Así mismo, cada base de datos tiene dentro del directorio `/var/lib/mysql` un directorio para definir su estructura. *Deberá borrarlos también*.

- Volcar (vía mysqldump) todos los esquemas y datos a disco:

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

- Detener MySQL.
- Borrar `ibdata1`, `ib_logfile0`, `ib_logfile1` y los directorios de bases de datos InnoDB.
- Iniciar MySQL.
- Crear la base de datos de `pandora` de nuevo (`create database pandora;`).
- Importar el fichero de respaldo (en este ejemplo, `all.sql`)

```
mysql -u root -p
mysql> create database pandora;
mysql> use pandora;
```

```
mysql> source all.sql;
```

Con estos cambios el sistema debería ir más rápido ahora.

Indices opcionales

En algunas situaciones es posible optimizar el funcionamiento de MySQL a costa de recursos del sistema.

Este índice sirve para optimizar la velocidad de obtención de gráficas a costa de un mayor uso de disco y un ligero descenso en el rendimiento de borrados e inserciones en las tablas de datos:

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

Actualmente en las tablas más pesadas de Pandora FMS en MySQL dicha optimización viene por defecto. Es conveniente preguntar a expertos antes de optimizar las tablas de MySQL.

Consultas lentas

En algunos sistemas, en función del tipo de información almacenado,puede encontrar algunas *slow queries* (consultas lentas) que hacen que el sistema vaya peor. Puede activar el *log* de este tipo de consultas durante un período corto de tiempo (ya que perjudica al rendimiento del sistema) a fin de estudiar las consultas a intentar optimizar las tablas con índices.

Para activar este sistema hay que hacer lo siguiente:

- Editar el archivo `my.cnf` y añadir las siguientes lineas:

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

- Para poder utilizarlo debe crearlo y establecer las reglas administrativas:

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

- Reiniciar MySQL.
- Al finalizar de analizar cuáles son las *slow queries* recuerde restablecer el fichero `my.cnf` *comentando* las líneas agregadas y reiniciando de nuevo el servicio MySQL.

Optimización de tablas específicas

Otra solución menos *radical* para paliar el problema de la fragmentación es el uso de la herramienta OPTIMIZE de MySQL en ciertas tablas de Pandora FMS. Para ello directamente desde MySQL, ejecutar:

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

Esto mejora el rendimiento y debería realizarlo solo una vez a la semana, pudiendo hacerse *en caliente*, es decir mientras el sistema trabaja.

En sistemas muy grandes el comando OPTIMIZE puede quedarse *bloqueado* y dejar de ser una alternativa: entonces es mejor reconstruir la BBDD.

Después de hacer estas operaciones, conviene ejecutar:

```
FLUSH TABLES;
```

Cita del manual de MySQL:

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

(Para las tablas de tipo InnoDB, OPTIMIZE TABLE se asigna a ALTER TABLE, que reconstruye la tabla para actualizar las estadísticas de índice y espacio libre no utilizado en el índice agrupado.)

En instalaciones a partir de 7.0 OUM 715 la siguiente configuración viene aplicada por defecto.

Nota: Si la instalación de Pandora FMS se realizó antes de la versión 7.0 OUM 715, se recomienda que realice las siguientes modificaciones en sus base de datos (principal e histórico):

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

Esta acción agregará un índice a la tabla `tagente_datos`, permitiendo que las consultas que utilizan tanto el sistema de informes, consulta de datos o gráficas de módulo o personalizadas devuelvan resultados en un tiempo sensiblemente menor al anterior.

Tokens especiales de MySQL

Existen algunos *tokens* muy especiales de MySQL que pueden ayudar o empeorar el rendimiento:

- `innodb_thread_concurrency`:

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

Este parametro en versión 5.1.12 o superiores, si vale 0 significa que no hay límite a la concurrencia, en versiones inferiores a la señalada, si vale 20 o más, implica lo mismo: sin límite a la concurrencia.

- `innodb_flush_method`:

```
innodb_flush_method = O_DIRECT
```

Este parámetro afecta a cómo se escribe en disco.

- `innodb_lock_wait_timeout`:

```
innodb_lock_wait_timeout = 90
```

Evita que ante un atasco ocasional el MySQL "se rinda" (*MySQL has gone away*) y se detenga. Si sobrepasa los 90 segundos es un gran problema.

Referencias

Referencias:

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

MySQL Percona XTraDB

Percona es una versión mejorada de MySQL, sobre todo respecto a la *escalabilidad* (crecimiento rápido sin afectar o afectar poco las operaciones y rutinas de trabajo). Aprovecha mejor los sistemas de múltiples CPU y acelera el acceso a disco.

Para configurar MySQL Percona utilice este excelente *kit* de herramientas: Percona Toolkit.

Dimensionamiento Pandora FMS para alta capacidad

Esta sección describe diferentes métodos para configurar Pandora FMS en un entorno de alta capacidad. Así mismo describe diferentes herramientas para hacer pruebas de carga, útiles para ajustar el entorno a la mayor capacidad de proceso posible.

Se ha configurado Pandora FMS para soportar una carga de alrededor de 2500 agentes en sistemas donde base de datos, consola y servidor están en la misma máquina. La cifra recomendada está en torno a 2500 agentes por sistema, pero esta cifra varía enormemente en funcion de si son agentes XML, módulos remotos, con intervalos altos o bajos, o con sistemas de mucha capacidad o poca memoria.

Todos factores alteran enormemente el número de agentes que un sistema puede gestionar eficientemente. En pruebas de laboratorio se han logrado ejecutar 10000 agentes en un solo servidor con hardware básico, pero fuertemente optimizado.

Ejemplo de configuración de servidores de alta capacidad

Suponiendo una máquina CentOS con 16GB de RAM y 8 CPU a optimizar para la máxima capacidad de procesamiento del data server (XML).

my.cnf

Solo se muestran los parámetros más significativos.

```
[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
```

Para MySQL versión 8, y versiones posteriores, el equipo de desarrollo de MySQL ha retirado el soporte para *query cache*, si desea obtener más información puede visitar el siguiente enlace web (en inglés): https://dev.mysql.com/blog-archive/mysql-8-0-retiring-support-for-the-query-cache/

pandora_server.conf

Se muestran solo los parámetros relevantes.

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

Aspectos a tener en cuenta:

- El número establecido en el parámetro `verbose` hace referencia a la cantidad de información con se escribe en los *logs*, siendo recomendable no sobrepasar de 10. Cuanto mayor sea el número, menor será el rendimiento de Pandora FMS debido a la gran cantidad de información a escribir en los *logs*.
- Un valor alto (15) del parámetro `server_threshold` hace que la BD se resienta menos, mientras que el incremento en el máximo número de ficheros procesados hace que cada vez que el servidor *busque ficheros* y llene los *buffers*. Estos dos elementos de la configuración están íntimamente ligados. En el caso de optimizar el *network server* sería recomendable bajar el `server_threshold` a 5 ó 10.
- El número de hilos muy alto (más de 5) establecido en dataserver_threads solo beneficia a procesos con largas esperas de E/S, como el *network server* o el *plugin server.* En el caso del *dataserver,* que está todo el tiempo en proceso, puede incluso perjudicar el rendimiento. En sistemas con una BD lenta, debe usar incluso menos hilos: pruebe diferentes combinaciones entre 1 y 10. En el caso de optimizar el sistema para el *networkserver*, el número sería mucho más alto, entre 10 y 30.

- Algunos parámetros de la configuracion pueden afectar mucho al rendimiento de Pandora FMS, tal como el parámetro `agent_access` (configurable desde la Consola).

Herramientas de análisis de capacidad (Capacity)

Pandora FMS dispone de varias herramientas que le ayudarán a dimensionar adecuadamente su hardware y software para el volumen de datos que espera obtener. Una de ellas sirve para *atacar* directamente la base de datos con datos ficticios (dbstress) y la otra genera ficheros XML ficticios (xml_stress)

Pandora FMS XML Stress

Este es un pequeño guión o *script* que genera ficheros de datos XML como los enviados por los agentes de Pandora FMS. Está ubicado en :

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

Los *scripts* leen los nombres de los agentes desde un fichero de texto y generan ficheros de datos XML para cada agente acorde con fichero de configuración, donde los módulos están definidos como plantillas.

Los módulos se rellenan con datos al azar. Se pueden especificar el valor inicial y la probabilidad de cambio de los datos de un módulo mediante valores escritos en un archivo.

Puede ejecutar el guión de este modo:

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

Ejemplo de configuración de un fichero (*configuration file*) llamado `pandora_xml_stress.conf`:

```
# 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
```

Enviar y recibir la configuración local del agente

Activando en su `pandora_xml_stress.conf` el valor de configuración `get_and_send_agent_conf` a 1, puede hacer que los agentes de prueba de carga se comporten como agentes normales, ya que envían su fichero de configuración y el *hash* MD5.

(E) Desde Pandora Console Enterprise puede modificar la configuración remota para que en sucesivas ejecuciones del `pandora_xml_stress` use la configuración personalizada desde Pandora Console Enterprise en vez de a través de la definición de `pandora_xml_stress.conf`.

A parte de eso puede configurar dónde guardar de forma local los `.conf` de tus agentes de prueba con el token de configuración `directory_confs` en el fichero `pandora_xml_stress.conf`.

Fichero de configuración

- max_threads: Número de hilos (*threads*) en la que se ejecutará el *script*, esto mejora la E/S.
- agent_file: Ruta del fichero de lista de nombres, separados por nueva línea.
- temporal: Ruta del directorio donde se generarán los ficheros de datos XML ficticios.
- log_file: Ruta del fichero de *log* donde informará el *script* de su ejecución.
- xml_version: Versión del ficheros de datos XML (por defecto 1.0).
- encoding Codificación de los ficheros de datos XML (por defecto ISO-8859-1).
- os_name: Nombre del sistema operativo de los agentes ficticios (por defecto GNU/Linux).
- os_version: Versión del sistema operativo de los agentes ficticios (por defecto 2.6).
- agent_interval: Intervalo de los agentes ficticios en segundos (por defecto 300).
- time_from: Fecha de comienzo a generar los ficheros de datos XML ficticios, en formato “AÑO-MES-DIA HORA:MIN:SEC”
- time_to: Fecha de fin a generar los ficheros de datos XML ficticios, en formato “AÑO-MES-DIA HORA:MIN:SEC”

- get_and_send_agent_conf: Valor *booleano* 0 ó 1, cuando está activo los agentes ficticios intentarán descargar por configuración remota una versión actual del fichero configuración estándar de un agente. Si tiene la Consola Pandora FMS Enterprise podrá editarlos de manera remota y masiva.
- startup_delay: Valor numérico de tiempo en segundos antes de comience cada agente a generar los ficheros, se usa para evitar *condiciones de carrera*.
- timezone_offset: Valor numérico del *offset* de *time zone*.
- timezone_offset_range: Valor numérico que sirve para generar dentro de este rango los *timezone* de forma aleatoria.
- latitude_base: Valor numérico, es la latitud donde apareceran los agentes ficticios.
- longitude_base: Valor numérico, es la longitud donde apareceran los agentes ficticios.
- altitude_base: Valor numérico, es la altitud donde apareceran los agentes ficticios.
- position_radius: Valor numérico, es el rango alrededor, la circunferencia con este radio en que aparece el agente ficticio de forma aleatoria.

Definición de los módulos

La definición de un módulo dentro del fichero de configuración *script* y si tiene activada la configuración remota también será igual:

```
 module_begin
 module_name <nombre_del_módulo>
 module_type <tipo_de_dato_módulo>
 module_description <descripción>
 module_exec type
=<tipo_generación_xml_stress>;<otras_opciones_separadas_por_punto_y_coma>
 module_unit <unidades>
 module_min_critical <value>
 module_max_critical <value>
 module_min_warning <value>
 module_max_warning <value>
 module_end
```

Y cada uno lo puede configurar como:

- tipo_generación_xml_stress: Puede tomar los valores RANDOM , SCATTER y CURVE.
- module_attenuation <value>: El valor generado se multiplica por el valor dado, normalmente entre 0.1 y 0.9.
- module_attenuation_wdays <value> <value> ... <value>: El valor del módulo se atenúa únicamente los días dados, que van de domingo (0) a sábado (6). Por ejemplo, el siguiente módulo simula una disminución del 50% en el tráfico de red los sábados y domingos:

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

```
module_end
```

- module_incremental <value>: Si se configura a uno, el valor previo del módulo se suma siempre a un nuevo valor, lo que da lugar a una función creciente.
- otras: Ver más abajo que opciones tiene, en función del tipo de generación del módulo.

Aleatorios (RANDOM)

Los cuales tienen las siguientes opciones:

- variation: Probabilidad en porcentaje de que varíe con respecto al valor anterior.
- min: Valor mínimo que puede tener el valor.
- max: Valor máximo que puede tener el valor.

Numéricos

Genera valores numéricos aleatorios entre el rango valor min y el valor max.

Boleanos

Genera valores entre 0 y 1.

Cadena

Genera una cadena de longitud entre valor min y el valor max, los caracteres son aleatorios entre A y Z incluídas mayúsculas y minúsculas y cifras numéricas.

Fuente externa de datos (SOURCE)

Permite escoger un archivo de texto plano como fuente de datos. Opciones:

- src: archivo fuente para los datos.

El archivo contiene un dato por línea, no hay límite de líneas. Por ejemplo:

```
4
5
6
10
```

Admite todo tipo de valores (numéricos y cadenas de texto). Este tipo de módulos usará cada unos de los datos del archivo para generar los datos de los módulos en Pandora FMS, los datos se toman de forma secuencial. Por ejemplo, los datos del módulo anterior se verían en Pandora FMS de esta manera:

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

Dispersión (SCATTER)

Solo vale para datos numéricos y la gráficas generadas son parecidas a las de un latido de corazón, es decir un valor normal y de vez en cuando un *latido*.

Y tiene las siguientes opciones:

- min: Valor mínimo que puede tener el valor.
- max: Valor máximo que puede tener el valor.
- prob: Probabilidad en porcentaje de que genere un *latido*.
- avg: Valor medio que debe mostrar por defecto si no hay ningún *latido*.

Curva (CURVE)

Genera datos de módulo siguiendo una curva trigonométrica. Los cuales tienen las siguientes opciones:

- min: Valor mínimo que puede tener el valor.
- max: Valor máximo que puede tener el valor.
- time_wave_length: Valor numérico en segundos de la duración de la *cresta* de la onda.
- time_offset: Valor numérico en segundos de comienzo de la onda desde el tiempo cero con valor cero del módulo (similar a la gráfica de seno).

Notas de interés:

- Esta herramienta está preconfigurada para buscar, en todos los agentes, los módulos de nombre «*random*», «*curve*» o «*boolean*», y que usen un intervalo entre 300 segundos y 30 días.

Como medir la capacidad de proceso del dataserver

Existe un pequeño guión llamado `pandora_count.sh` que está en el directorio `/usr/share/pandora_server/util/` del servidor de Pandora FMS. Este *script* se usa para medir la tasa de procesamiento de ficheros XML por el *data server*, y utiliza como referencia el total de ficheros pendientes de procesar en `/var/spool/pandora/data_in`, de forma que para poder usarlo se necesita tener pendiente de procesar varios miles de paquetes (o generarlos con la herramienta anteriormente mencionada). Una vez en ejecución, puede detenerlo pulsando las teclas CTRL+C

Este *script* simplemente cuenta los paquetes existentes ahora y los resta de los paquetes que había hace 10 segundos, luego divide el resultado por 10 y esos son los fiheros que se han procesado en los últimos 10 segundos, mostrando la tasa por segundo. Es una medida algo burda pero sirve para ajustar la configuración del servidor.

Pandora FMS DB Stress

Esta es una pequeña herramienta para probar el rendimiento de su base de datos. También se puede usar para *pregenerar* datos periódicos o aleatorios (usando funciones trigonométricas) y rellenar módulos ficticios.

Se debe crear un agente y asignarle módulos para inyección de datos automática con esta herramienta. Los nombres se deben llamar con la notación siguiente:

- *random*: para generar datos aleatorios.
- *curve*: para generar una curva de coincidencias usando funciones trigonométricas. Útil para ver el trabajo de interpolación con diferentes intervalos, etc.
- *boolean*: generar datos booleanos aleatorios.

De tal forma que se podría usar cualquier nombre que contenga las palabras *«random»*, *«curve»* y/o *«boolean»*, por ejemplo:

- random_1
- curve_other

Sólo se podrá elegir el tipo de módulo «data_server».

Ajuste fino de la herramienta Pandora FMS DB Stress

Esta herramienta está preconfigurada para buscar, en todos los agentes, los módulos de nombre *«random»*, *«curve»* o *«boolean»*, y que usen un intervalo entre 300 segundos y 30 días.

Si se quiere modificar este comportamiento, se debe editar el *script* `pandora_dbstress` y modificar algunas variables al inicio del fichero:

```
# 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. La primera línea de variable correspondiente con `target_module`, se debe establecer para un módulo fijo o `-1` para procesar todos los objetivos que coincidan.
2. La segunda línea de variable corresponde con `target_agent`, para un agente específico.
3. La tercera línea corresponde con `target_interval`, definida en segundos y que representa el intervalo de periodicidad predeterminada del módulo.
4. La cuarta línea es *`target_days`* y representa el número de días en el pasado desde la fecha, en *timestamp*, actual.

Herramientas de Diagnóstico en Pandora FMS

Algunas veces los usuarios tienes problemas y los desarrolladores de Pandora FMS no le pueden

ayudar sin tener más información acerca de sus sistemas. En la versión 3.0 se desarrollaron algunas herramientas con el fin de ayudarle a resolver algunos problemas.

Diagnostic Info

La funcionalidad para poder obtener información de diagnóstico de la instalación de Pandora FMS se encuentra dentro de la sección de Management → Admin tools → Diagnostic Info. En esta ventana podrá observar la información acerca de la configuración de Pandora FMS y MySQL, así como gráficas del sistema de automonitorización.

pandora_diagnostic.sh

Es una herramienta está localizada en `/usr/share/pandora_server/util/pandora_diagnostic.sh` y proporciona mucha información acerca del sistema:

- Información de la CPU.
- *Uptime* y *avgload* (tiempo de funcionamiento y promedio de carga de trabajo) de la CPU.
- Información sobre la memoria.
- Información sobre Kernel/liberación.
- Un fichero *dump* de configuración MysQL.
- Un fichero *dump* (filtrando contraseñas) de configuración del servidor de Pandora FMS.
- Los *logs* de información de Pandora FMS (*¡pero no el log completo!*).
- Información del disco.
- Información sobre procesos de Pandora FMS.
- Una información completa del *log* del kernel (dmesg).

Toda la información se genera en un fichero `.txt`, con lo que los usuarios pueden enviar esta información a cualquiera que quiera ayudarles, por ejemplo, en el foro de usuarios de Pandora FMS o en las listas de correo públicas de Pandora FMS. Este archivo no deberá contener ningún tipo de información confidencial. Tenga en cuenta que posiblemente usted quiera correr con privilegios de *root* si quiere obtener los ficheros `pandora_server.conf` y `my.cnf` .

Este es un ejemplo de ejecución:

```
$ ./pandora_diagnostic.sh

Pandora FMS Diagnostic Script v1.0 (c) 2009
https://pandorafms.com/ . 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'
```

Volver al Índice de Documentación Pandora FMS