Percona Server features for OpenStack and Trove Ops

George O. Lorch III
Software Developer – Percona

Vipul Sabhaya
Lead Software Engineer - HP

Overview

- Discuss Percona Server features that will help operators manage various aspects of server instances
- Discuss ongoing research on limiting UNDO space and techniques that may be used to prevent XtraDB from getting into an unrecoverable state due to ENOSPC.
- Discuss useful integration points between Percona Server and Percona XtraBackup.
- Discuss XtraBackup features that will help operators secure and store their backup data including parallel compression, encryption and the upcoming parallel streaming to Swift.
- Describe the intersection of all of these things with Openstack Trove

Utility user

- Feature introduced to allow a special type of MySQL user that would have a very limited and well defined amount of control over the system.
- The utility user has a special scoping of abilities, visibility and protection from other MySQL users.

Utility user - specifying

- The utility users credentials, schema access and privileges may be specified either on the mysqld command line or within a my.cnf file, but not via system variable
  - **utility-user-name='user' @ 'hostname'** : specifies a fully qualified user name in the form of 'user' @ 'hostname' which the system will recognize as the utility user
  - **utility-user-password=<password>** : specifies the password for the utility user
  - **utility-user-schema-access=<schema>,<schema>,...** : specifies the name(s) of the schema(s) that the utility user will have full access to read write and modify
  - **utility-user-privileges=<privilege>,<privilege>,...** : specifies the name(s) of the standard MySQL user privilege(s) that the utility user will be granted

Utility user - behaviors

- Can not be seen or modified by any other user, including root
- Will not appear in USER, CLIENT or THREAD statistics
- Any queries executed by this user may appear in the general and slow query logs
- Will not have the ability create, modify, delete or see any schemas not explicitly specified unless given MySQL privileges
- May modify all visible, non-read only system variables allowed by privileges granted (limited by options modifiers)
- May see, create, modify and delete other system users if granted access to the mysql schema, primarily for the purpose of password reset/recovery
- May control higher level system functions if provided privilege set allows it to such as replication

Utility user - behaviors

- The utility user must not be the same as any other user that exists in the mysql.user table.
- If a client attempts to create a MySQL user that matches the utility user specification, the creation attempt will fail with an error

Utility user - behaviors

- There is an interesting interaction between **utility-user-schema-access** and certain **utility-user-privileges**:
  - If the utility user is granted full access to a specific schema via **utility-user-schema-access**, the utility user will already have CREATE, INSERT, DELETE, ALTER and DROP on that schema and not need that access specified via the **utility-user-privileges** option
  - Conversely, the utility user will NOT need explicit schema access via **utility-user-schema-access** in order to execute any privileges granted via the **utility-user-privileges** option

Utility user – questions?

Options modifiers

- MySQL has the concept of options modifiers which is a simple way to modify either the way that MySQL interprets an option or the way the option behaves
- Option modifiers may be specified either on the mysqld command line or within the my.cnf file
- Option modifiers are used by simply pre-pending the name of the modifier and a dash "-" before the actual configuration option name:
  - For example specifying –maximum-query_cache_size=4M on the mysqld command line or specifying maximum-query_cache_size=4M in the my.cnf will prevent any client from setting the query_cache_size value larger than 4MB.

Options modifiers

- Currently MySQL has five option modifiers:
  - **disable [disable-<option_name>]** - disables or ignores option_name
  - **enable [enable-<option_name>]** - enables option_name
  - **loose [loose-<option_name>]** - mysqld will not exit with an error if it does not recognize option_name, but instead it will issue only a warning
  - **maximum [maximum-<option_name>=<value>]** - indicates that a client can not set the value of option_name greater than the limit specified. If the client does attempt to set the value of option_name greater than the limit, the option_name will simply be set to the defined limit
  - **skip [skip-<option_name>]** - skips or ignores option_name

Options modifiers

- Percona Server has the following new option modifiers to offer more control over option visibility, access and range limits:
  - **minimum [minimum-<option_name>=<value>]** - indicates that clients can not set the value of option_name to less than the limit specified. If the client does attempt to set the value of option_name lesser than the limit, the option_name will simply be set to the defined limit
  - **hidden [hidden-<option_name>=<TRUE/FALSE>]** - indicates that clients can not see or modify the value of option_name
  - **readonly [readonly-<option_name>=<TRUE/FALSE>]** - indicates that clients can see the value of option_name but can not modify the value

Options modifiers

- With this full set of option modifiers, operators can now set reasonable limits on all dynamic options and help prevent tenants from shooting themselves in the foot
- Sensitive static values can be hidden from the tenant if desired

Options modifiers – questions?

Enforcing a storage engine

- Percona Server has a new option that allows the ability to enforce the use of a particular storage engine
- **enforce-storage-engine=<engine>** : specifies the name of a storage engine which must be used for all new CREATE and ALTER table statements

Enforcing a storage engine

- When this option is specified and a user tries to create a table using an explicit storage engine that is not the specified enforced engine, they will get either an error if the NO_ENGINE_SUBSTITUTION SQL mode is enabled or a warning if NO_ENGINE_SUBSTITUTION is disabled and the table will be created anyway using the enforced engine (this is consistent with the default MySQL way of creating the default storage engine if other engines aren't available unless NO_ENGINE_SUBSTITUTION is set)
- In the case that an operator tries to use **enforce-storage-engine** with engine that isn't available, system will not start

Enforce storage engine – questions?

Preventing LOAD DATA INFILE and SELECT INTO OUTFILE

- These operations can now be completely disabled by an operator to prevent a user from trying to either load or dump data directly to a local mount point. Since the user will generally not have ssh or any file system access to the instance, there is no reason for them to attempt to execute these commands.
- Percona Server has extended the upstream MySQL option –secure-file-priv that allows you to restrict these operations to certain locations on disk. Now, by specifying –secure-file-priv without an argument, you will completely disable the LOAD DATA INFILE and SELECT INTO OUTFILE functionality.
- LOAD DATA LOCAL INFILE is not affected as the source of the datafile is local to the client making the call.

Preventing LOAD DATA INFILE and SELECT INTO OUTFILE – questions?

Binary and slow log size and rotation

- MySQL has the option **max-binlog-size** that allows you to limit the size of an individual binlog
- Percona Server extends this to add the **max-binlog-files** option. When this option is set to non-zero value, the server will remove the oldest binlog file(s) whenever their number exceeds the value of this option
- When **max-binlog-files** is combined with **max-binlog-size**, you have a very effective means of limiting the total size that binlogs will occupy
  - Ex: if **max-binlog-size** is set to 1G and **max-binlog-files** to 20 this will limit the maximum size of the binlogs on disk to around 20G

Binary and slow log size and rotation

- Percona Server has duplicated the **max-binlog-size** and **max-binlog-files** options for use with the slow query log
- These new options are **max-slowlog-size** and **max-slowlog-files** and behave exactly the same way as the binlog options, thus allowing you to limit the maximum size occupied by the slow query logs

Binary and slow log rotation – questions?

UNDO space

- UNDO records are the ‘before’ values of an INSERT or UPDATE
- UNDO records are stored in the system tablespace in 5.5 and can be relocated to their own tablespace(s) in 5.6 via the innodb-undo-dir, inodb-undo-tablespaces, and innodb-undo-logs options
- There is currently no built in mechanism to control or limit the growth of UNDO space
- There is currently no mechanism to automatically reclaim UNDO space

UNDO space

- The 5.6 options which allow you to relocate the UNDO records into their own tablespace offer some ways to try to control UNDO records
- UNDO growth can be controlled by placing UNDO files/tablespaces on a separate partition of the desired size or by applying a disk quota to the UNDO directory. In most situations, when the UNDO limit has been hit by a transaction, a rollback will then be performed and an error returned to the client

UNDO space

- InnoDB and XtraDB have some ‘soft’ spots with regards to UNDO space and ENOSPC:
  - MySQL Bug #43665 - innodb crashes when tablespace reaches maximum configured size
  - MySQL Bug #67606 - MySQL crashes with segmentation fault when disk quota is reached
  - Percona Server Bug #1079596 - Percona Server crashes with segmentation fault when disk quota is reached

UNDO space

- The core of this issue is that innodb does not properly reserve enough space for both the transaction data and potential rollback space at the same time, nor does it hold back or reserve any space to be used exclusively for rollback
- If this issue is hit, the server will assert and will continue to assert on restart/recovery until enough free space is available to complete the recovery
- A simple preventative workaround is to reserve some small portion of disk space with a dummy file that can act as a 'high water mark'. Then, if you encounter this situation, you simply remove the dummy file and restart the server allowing it to recover and giving you some time to reclaim some space

UNDO space

- There are some discussions on what can be done to help limit UNDO space but all ideas come at a cost:
  - Provide a 'hard limit' of pages that may be consumed by UNDO which when hit, would cause the rollback of the transaction attempting to exceed the limit and return an error to the client.
  - Provide a 'soft limit' of pages that may be consumed by UNDO which when hit:
    - Would trigger an analysis of purge lag and kill any old transactions holding read only (REPEATABLE-READ) view and blocking purge thread.
    - Would trigger an analysis of transactions performing only DML and kill old/idle transactions.

UNDO space

- UNDO space and purge lag resources:
  - http://dimitrik.free.fr/blog/archives/06-01-2009_06-30-2009.html#61
  - http://www.mysqlperformanceblog.com/2010/06/10/reasons-for-run-away-main-innodb-tablespace
  - http://www.mysqlperformanceblog.com/2010/06/10/purge-thread-spira-of-death

UNDO space – questions?

Percona Server and XtraBackup together

Percona Server 5.6 and XtraBackup 2.2.x now have several interesting and useful integration points that are not available with other MySQL variants:

- Changed page tracking – Percona Server can now track changed XtraDB pages in a special series of bitmap files. When performing incremental backups, this allows XtraBackup to read only the XtraDB pages that are known to have changed since the last backup instead of a full file scan.
- Backup locks – Percona Server has implemented a lightweight alternative to FLUSH TABLES WITH READ LOCKS for both physical and logical backups. Unlike FLUSH TABLES WITH READ LOCK, these new locks do not flush tables, i.e. storage engines are not forced to close tables and tables are not expelled from the table cache. As a result, the new lock only waits for conflicting statements to complete (i.e. DDL and updates to non-transactional tables). It never waits for SELECTs, or UPDATEs to InnoDB tables to complete. Percona XtraBackup will automatically detect if these locks are available and use them.
- For more information on these features, please see the Percona Server and XtraBackup documentation.

Percona XtraBackup

Percona XtraBackup has several interesting features that can help operators tune the backup process and secure your customers data:

- Parallel, lightweight compression of the backup data (quicklz)
- Parallel, symmetric AES 64-256 bit encryption of the backup data (gcrypt)
- Maintain a backup history on the source server and perform incrementals based on a simple backup series name or specific backup UUID. Eliminates the need for lots of backup output parsing and scripting.

Coming soon!

- Asymmetric/public key encryption.
- Parallel streaming of a backup to Swift Object Store into static large objects.

For more information on these features, please contact me or see the Percona XtraBackup documentation.

Questions and discussion

THANK YOU!

- We sincerely appreciate you taking the time to participate and attend our talk.
- Slides will be available shortly after the conference.
- If you have any questions related to this talk, Percona or HPs products and services and service in general, please feel free to contact us.

George O. Lorch III
Software Developer – Percona

Vipul Sabhaya
Lead Software Engineer – HP
vipul.sabhaya@hp.com