



PALOMINO DB

Proven Database Excellence

MySQL Backup and Recovery: Tools and Techniques

Presented by:

René Cannaò @rene_cannaò

Senior Operational DBA

www.palominodb.com

EXPERIENCE WITH BACKUP

- How many of you consider yourself beginners?
- How many are experienced and want to learn more?
- Snapshot?
- Incremental backup?
- Data recovery?



WHY DO I NEED BACKUPS?

- In short: because data is the most valuable asset an organization possesses, and you have to protect it
- It is an essential component too often forgotten when implementing a new system
- plan backup and recovery from the beginning



BACKUPS ARE USEFUL FOR ... ?

- Disaster recovery
- Auditing
- Rollback
- Testing
- Scalability



BACKUPS FOR DISASTER RECOVERY

- **Hardware failures**
 - system crash; corrupted memory; disk failure
- **Software failures**
 - mysqld bug; 3rd parties software failure (even backup software)
- **Human mistakes**
 - accidental DDL/DML statements, or application bug
- **Data compromised by attacker**
 - malicious outsider (security topic)



BACKUPS FOR AUDITING

- Need to know how the data looked in the past
- Need to trace an odd event detected in a late moment
- Involvement in a lawsuit



BACKUPS FOR ROLLBACK

- Rollback to a previous application release
- Reimplementation of a removed feature
- Remove the effect of some features/events



BACKUPS FOR TESTING

- Refresh staging/testing environment
- Test new features
- Run benchmark
- Test scalability



BACKUPS FOR SCALABILITY

- Build slaves (scale out)
- Build remote sites (disaster recovery)
- Hardware upgrade (scale up)
- Integrate the backup process with other operational processes for constant testing



WHAT DO I NEED TO BACKUP?

The real questions are:

- What can I afford to lose?
- How efficient is backup?
- How efficient is recovery?



WHAT CAN I AFFORD TO LOSE?

- Is it ok to lose all the transactions after the last full backup ? (last week)
- Is it ok to lose all the transactions after the last incremental backup ? (last night)
- Is it ok to be able to do point-in-time recovery close enough to when the problem happened?
- Or do I need to do point-in-time recovery at the exact time when the problem happened?

Less data we can afford to lose, more complex our backup become. This makes a difference not only in the procedure, but also in the architecture.



TERMINOLOGIES AND DIFFERENCES

- Offline / Online backup
- Logical / Raw backup
- Full / Incremental backup



OFFLINE BACKUP

Done taking the server offline, and eventually shut it down (depends from the type of backup)

- Pros:
 - _ safe (less risk of corruption and inconsistency)
 - _ no contention with application (is not using the system)
 - _ Fast
- Cons:
 - _ can cause interruption of service if you have only one database server
 - _ expensive, if you have a slave server just for backup
 - _ in case of shutdown/start : this is a slow operation
 - _ If only 2 servers are present and one crashes during backup, MTTR increases

Because the server needs to be taken offline, in production offline backups are possible only on slaves.



ONLINE BACKUP

- Pros:
 - requires minimal interruption of service
 - depending from the load and workload on the server, the interruption of service can be less than a second , or being very disruptive
- Cons:
 - contentions with the application
 - CPU / RAM / disk I/O and network are limited resources

A typical winner is online backup from a replication slave



LOGICAL BACKUP

- Pros:
 - _ Text file : easy to manipulate (not with text editor)
 - _ Easy to restore
 - _ Many options available for mysqldump (including filters)
 - _ Storage engine independent
 - _ Portable
- Cons:
 - _ CPU intensive during dump (but eventually less disk IO)
 - _ Very CPU intensive during restore
 - _ Can cause a lot of disk IO during restore
 - _ Incorrect representation of floating-point values
 - _ Slow on backup and restore (especially on big tables)
 - _ Corruption if incorrect charset is used (ex. Different defaults across versions)



RAW BACKUP

- Pros:
 - Copy of data files
 - Easy to restore a single table (limitations apply)
 - Quite portable (not always)
 - Faster to restore (no processing of SQL statements)
- Cons:
 - Size
 - Not always portable



FULL BACKUP

- Backup of all the data
- Normally very large
- Easy to restore



INCREMENTAL BACKUP

- MySQL binary log
- Generally smaller than full backup
- Point in time recovery
- Skip binlog events
- Slow (single thread, both SBR and RBR)



STORING BACKUP

- On the same server (but different storage)
- Off-site on another server
- Backup media
- Encryption
- Compression
- Retention policy (ask your business user)
- Multiple strategies for different needs
- Think over recovery time



TEST YOUR BACKUPS! (DO IT!)

- Never blindly trust backups
- Backup procedure could be incorrect
- Recovery procedure could be incorrect
- Avoid false sense of security
- Try recovery
- Practice recovery



PRACTICE RECOVERY

- Backups are automated, recovery aren't
- Backups are routine, recovery aren't
- Backups are often better documented than recovery
- Recovery is performed under pressure
- Be aware of recovery time
- Try different recovery scenario (from full backup, PIT, single table)
- Train personnel with recovery
- Integrate recovery into automated processes



Q: WHAT TO BACKUP?

A: WHAT YOU NEED TO RESTORE!!

- What needs to be restored?
 - One or more tables?
 - A single database?
 - The whole dataset?
 - The whole MySQL instance?
 - The whole server?
- How much data can I afford to lose?
- Need point in time recovery?
- Acceptable recovery time?



WHAT TO BACKUP? (CONT)

- Data
- Non data files
 - Binlog , replication files
 - Triggers
 - Tables definition
 - Stored procedures
 - Events
- MySQL configuration
- OS configuration
- Custom script



GOOD PRACTICES

- Raw backup for large dataset
- Logical backup
 - for the whole dataset
 - for a subset of the whole dataset
- Incremental backup for point in time recovery
- Have a good retention policy
- Monitor backups, check functionality
- Test backups, automatic reminders
- Test backups (YES, DO IT!)



BACKUP CONSISTENCY: FILE CONSISTENCY

- Differences between storage engines:
 - MyISAM : FLUSH TABLES WITH READ LOCK
 - InnoDB : background threads
 - MEMORY: no data on disk



BACKUP CONSISTENCY: DATA CONSISTENCY

- Prevent data changes from application
 - All tables need to be back up at the same time
 - LOCK TABLES;
 - SHOW MASTER STATUS;
- Perform snapshot
 - InnoDB MVCC (REPEATABLE READ)



FILE SYSTEM SNAPSHOT: LVM

- Point in time snapshot of a block device creates a new block device
- Raw backup: copy the database files (with any tool)
- Logical backup: start a second mysqld instance using the snapshot



FILE SYSTEM SNAPSHOT: LVM (cont)

- Almost no downtime
 - FLUSH TABLES [WITH READ LOCK]
- Storage engine independent
- Fast backup
- Fast recovery



FILE SYSTEM SNAPSHOT: LVM (cont)

- Safe
- Small IO overhead
 - depending from workload
- All data need to be on the same Logical Volume



FILE SYSTEM SNAPSHOT: LVM (cont)

MyISAM and InnoDB

- `mysql> FLUSH TABLES WITH READ LOCK;`
- `mysql> SHOW MASTER STATUS;`
- `shell> lvcreate -snapshot -size 10G -name backup /dev/VolGroup00/mysql`
- `mysql> UNLOCK TABLES;`
- `shell> mount /dev/VolGroup00/backup /mnt/backup`
- (copy the data over)
- `shell> umount /mnt/backup`
- `shell> lvremove /dev/VolGroup00/backup`



File System Snapshot: LVM (cont)

Only InnoDB (really hot backup):

- ~~mysql> FLUSH TABLES WITH READ LOCK;~~
- ~~mysql> SHOW MASTER STATUS; *~~
- shell> lvcreate -snapshot -size 10G -name backup /dev/VolGroup00/mysql
- ~~mysql> UNLOCK TABLES;~~
- shell> mount /dev/VolGroup00/backup /mnt/backup
- (copy the data over)
- shell> umount /mnt/backup
- shell> lvremove /dev/VolGroup00/backup

* InnoDB reports binlog coordinates in the error log



FILE SYSTEM SNAPSHOT: LVM (cont)

Recovery from LVM snapshot:

- Use the backup to create a datadir
- Configure mysqld to use such datadir, check:
 - Paths
 - Permissions
 - innodb_log_file_size
- Start mysqld
- InnoDB will perform automatic recovery



FILE SYSTEM SNAPSHOT: LVM (cont)

- `service mysql stop`
- `mv /var/lib/mysql /var/lib/mysql_bck`
- `mkdir /var/lib/mysql && cd /var/lib/mysql`
- `tar -zxf /path/to/backup/file.tar.gz`
- `chown -R mysql:mysql /var/lib/mysql`
- `service mysql start`



mylvmbackup

- <http://www.lenzg.net/mylvmbackup/>
- Tool that automatizes LVM snapshot
- Records binlog coordinates
- Records replication coordinates
- Supports compression
- Supports InnoDB recovery before backup



EBS SNAPSHOT ON AWS

- Elastic Block Store (EBS)
- Block level storage volumes
- `mysql> FLUSH TABLES WITH READ LOCK;`
- `mysql> SHOW MASTER STATUS;`
- `shell> xfs_freeze -f /vol *`
- `console> ec2-create-snapshot -v vol-VVVV1111`
- `shell> xfs_freeze -u /vol`
- `mysql> UNLOCK TABLES;`
 - * AWS recommends to unmount the FS before doing the snapshot. This is not convenient, so writes on FS needs to be blocked.



EBS SNAPSHOT ON AWS (cont)

Recovery from EBS snapshot:

- `ec2-create-volume --snapshot snap-SSSS1111`
- `ec2-attach-volume -d /dev/sdh -i i-III2222 vol-VVVV2222`
- `mount /dev/sdh /ebs`
- `service mysql start`
- InnoDB will perform automatic recovery



XtraBackup

- <http://www.percona.com/software/percona-xtrabackup/>
- Copies InnoDB tables without locks
- Records InnoDB REDO logs
- Copies MyISAM tables locking them
- Records binlog coordinates
- Records replication coordinates
- Supports compression



XtraBackup (cont)

- Uses innobackupex as a wrapper to:
 - Copy table definition and other non data object
 - Copy MyISAM table
 - Do point in time recovery
- Consistent at the end of the backup
- Constantly developed and extra features added



XtraBackup (cont)

- Create a full backup:
 - innobackupex /path/to/backup-dir/
- Preparing a full backup:
 - innobackupex –apply-log /path/to/backup-dir/
- Restore a full backup:
 - innobackupex –copy-back /path/to/backup-dir/
- Fix permission:
 - chown -R mysql:mysql /path/to/datadir
- Start mysqld
 - service mysql start



mysqldump

- Logical backup
- Data and non-data stored together
- Single huge file
- SQL statements
- Supports a wide set of options



mysqldump (cont)

Some interesting features/options:

- All databases , list of databases or list of tables
- Filter rows
- Records binary log coordinates
- Records replication coordinates
- Dump only schema , or no schema
- Lock tables
- Single transaction
- Complete insert
- Extended insert



mysqldump (cont)

- `mysqldump --all-databases`
- `mysqldump --databases db1 db2`
- `mysqldump db1 tbl1 tbl2 tbl3`
- `mysqldump --skip-add-drop-table`
- `mysqldump --no-data`
- `mysqldump --master-data`
- `mysqldump --dump-slave`
- `mysqldump --single-transaction` (for InnoDB only)
- `mysqldump --where`



mysqldump (cont)

- `mysqldump --add-locks`
- `mysqldump --skip-add-locks`
- `mysqldump --complete-insert`
- `mysqldump --extended-insert`
- `mysqldump --opt`
 - `--opt` = `--add-drop-table`, `--add-locks`, `--create-options`,
`--disable-keys`, `--extended-insert`, `--lock-tables`, `--quick`
- `--skip--option` disable/invert the option



mysqldump (cont)

- Create a full backup
 - `mysqldump --single-transaction --master-data=2 --all-databases | gzip --fast > dump.sql.gz`
- Create a backup of only the schema definition (no data):
 - `mysqldump --no-data > schema.sql`
- Recover a backup
 - `zcat dump.sql.gz | mysql`
 - `(echo "SET SESSION SQL_LOG_BIN = 0;" ; zcat dump.sql.gz | egrep '^INSERT INTO `cities`') | mysql`



MYSQL BINARY LOG

- Binlog events record data changes
 - SQL statements (SBR)
 - Data being written (RBR)
- Replication
- Point in time recovery
- Incremental backup
- Store them on their own block device
 - Eventually, replicated with DRBD



MYSQL BINARY LOG (cont)

- `log_bin = mysql-bin`
- `log_bin_index = mysql-bin.index`
- `sync_binlog = 1`
- `expire_logs_days`
- `log_slave_updates`



MYSQL BINARY LOG (cont)

- Automate the backup process
- FLUSH LOGS
- rsync / cp



POINT IN TIME RECOVERY

- Restore the last full backup
- Replay the binary log
 - Starting from the time of the last full backup
 - Stop at the point you want to recover
 - to the last incremental backup
 - to the last binlog available
 - to the binlog event you want to skip



POINT IN TIME RECOVERY (cont)

- To apply an entire binlog:
 - `mysqlbinlog mysql-bin.008253 | mysql`
- To analyze the content of a binlog:
 - `mysqlbinlog mysql-bin.008259 | less`
 - `mysqlbinlog mysql-bin.008259 | egrep -B 5 '^DROP'`
- To apply binlog till a certain datetime
 - `mysqlbinlog --stop-datetime="2011-10-20 12:34:56" mysql-bin.00825[3-9] | mysql`
- To apply binlog from a specific position:
 - `mysqlbinlog --start-position=123456 mysql-bin.008259 | mysql`



BUILD A SLAVE

- Perform recovery from last full backup
- Define an unique server-id
- CHANGE MASTER TO
 - Coordinates are recorded during backup
 - from SHOW MASTER STATUS (to clone the master)
 - from SHOW SLAVE STATUS (to clone a slave)
- START SLAVE



Zmanda Recovery Manager for MySQL

- Backup Manager
- Backup Scheduler
- Supports different backup tools
- Full backup and incremental backup
- Logical backup and raw backup
- Compression and/or encryption
- Plugins support
- Retention policies
- Email notification



MySQL ZRM

General config file: /etc/mysql-zrm/mysql-zrm.conf

verbose=1

retention-policy=6W

all-databases=1

destination=/backups

backup-mode=logical

compress-plugin=/bin/gzip

user=backup

password=backup

compress=1

mailto=monitoring@example.com



MySQL ZRM (cont)

Dataset specific config file:

```
/etc/mysql-zrm/DB1/mysql-zrm.conf
```

```
retention-policy=4w
```

```
backup-mode=raw
```

```
snapshot-plugin="/usr/share/mysql-zrm/plugins/lvm-  
snapshot.pl"
```

```
copy-plugin="/usr/share/mysql-zrm/plugins/socket-  
copy.pl"
```



MySQL ZRM (cont)

- snapshot-plugin
 - creates the snapshot
 - removes the snapshot at the end of the backup
- copy-plugin
 - copies the data from the database server
- Communication happens through port 25300 using an xinetd service
- server = `/usr/share/mysql-zrm/plugins/socket-server.pl`



MySQL ZRM (cont)

Plugins at PalominoDB

- innobackupex
- snapshot-plugin="/usr/share/mysql-zrm/plugins/stub-snapshot.pl"
- copy-plugin="/usr/share/mysql-zrm/plugins/xtrabackup-client.pl"
- server = /usr/share/mysql-zrm/plugins/xtrabackup-agent.pl



Ubiquitous We're Hiring Slide

WE'RE HIRING



Need Proactive Operational Database Support?

- * Proactive team integration
- * 24x7 Tier 1 or 2 emergency support
- * MySQL and variants, PostgreSQL, NoSQL and Key/Value stores
- * Data Modeling, Release Process Integration, Proactive SQL Reviews
- * Monitoring and Trending
- * High Availability Solutions
- * Backup, Recovery and Disaster Recovery
- * BI, Data Analysis and Warehousing

