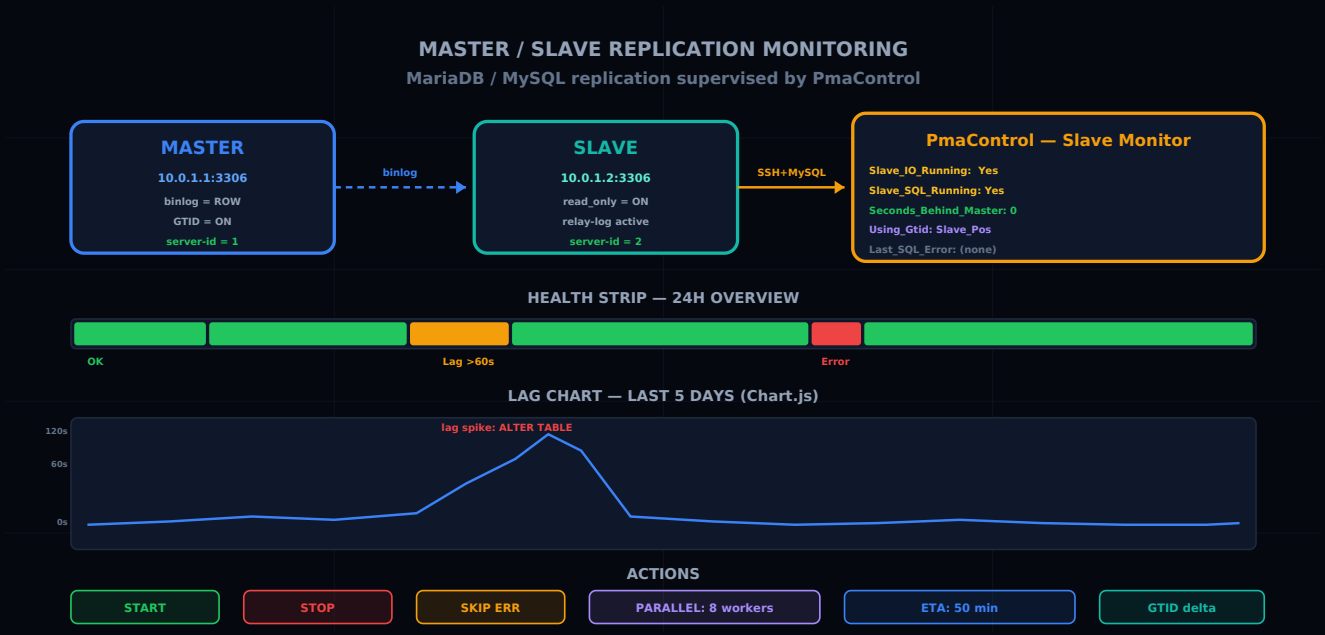
Déployer Master/Slave et superviser la réplication dans PmaControl

Aurélien LEQUOY · 13 avril 2026

MARIADB MYSQL REPLICATION MASTER-SLAVE MONITORING PMACONTROL



La réplication en 2026 : toujours le pilier

La réplication master/slave reste la brique fondamentale de toute infrastructure MariaDB / MySQL en production. Haute disponibilité, lecture distribuée, backups chauds — tout repose sur elle. Et pourtant, la majorité des incidents graves en base de données impliquent une réplication cassée ou un lag non détecté.

Cet article couvre les deux faces de la réplication : comment la mettre en place proprement, puis comment la superviser dans PmaControl pour ne jamais être surpris.

Configurer la réplication

Prérequis côté master

Le master doit avoir le binlog activé et un `server-id` unique :

```
[mysqld]
server-id = 1
log-bin = /var/log/mysql/mysql-bin
binlog-format = ROW
gtid_strict_mode = ON          # MariaDB
# enforce_gtid_consistency = ON  # MySQL
# gtid_mode = ON                 # MySQL
```

Créer l'utilisateur de réplication :

```
CREATE USER 'repl'@'10.0.1.%' IDENTIFIED BY 'secret_replication_password';
GRANT REPLICATION SLAVE ON *.* TO 'repl'@'10.0.1.%';
```

Prérequis côté slave

Le slave a besoin de son propre `server-id` et du relay log :

```
[mysqld]
server-id = 2
relay-log = /var/log/mysql/relay-bin
read-only = ON
log-slave-updates = ON
```

`log-slave-updates` est essentiel si vous prévoyez de chaîner les slaves (slave d'un slave) ou d'utiliser Galera.

Lancer la réplication

Avec GTID (recommandé)

Sur MariaDB :

```
CHANGE MASTER TO
  MASTER_HOST = '10.0.1.1',
  MASTER_USER = 'repl',
  MASTER_PASSWORD = 'secret_replication_password',
  MASTER_USE_GTID = slave_pos;

START SLAVE;
```

Sur MySQL 8.0+ :

```
CHANGE REPLICATION SOURCE TO
  SOURCE_HOST = '10.0.1.1',
  SOURCE_USER = 'repl',
  SOURCE_PASSWORD = 'secret_replication_password',
  SOURCE_AUTO_POSITION = 1;

START REPLICA;
```

Sans GTID (mode classique)

Si GTID n'est pas activé, il faut noter la position binlog du master :

```
-- Sur le master
SHOW MASTER STATUS;
-- File: mysql-bin.000042, Position: 154

-- Sur le slave
CHANGE MASTER TO
  MASTER_HOST = '10.0.1.1',
  MASTER_USER = 'repl',
  MASTER_PASSWORD = 'secret_replication_password',
  MASTER_LOG_FILE = 'mysql-bin.000042',
  MASTER_LOG_POS = 154;

START SLAVE;
```

Vérifier que ça fonctionne

```
SHOW SLAVE STATUS\G
```

Les deux indicateurs clés :

```
Slave_IO_Running: Yes
Slave_SQL_Running: Yes
Seconds_Behind_Master: 0
```

Si `Slave_IO_Running` est `No`, le slave ne peut pas se connecter au master (réseau, credentials, firewall). Si `Slave_SQL_Running` est `No`, le slave ne peut pas appliquer les événements (erreur

SQL, contrainte violée).

Ajouter les serveurs à PmaControl

Via l'interface web

1. Allez dans **Serveurs → Ajouter un serveur**
2. Renseignez l'IP, le port (3306), les credentials SSH et MySQL
3. PmaControl détecte automatiquement le rôle (master ou slave) après le premier cycle de collecte

Via l'API REST

```
# Ajouter le master
curl -X POST -H "Authorization: Bearer $TOKEN" \
  -H "Content-Type: application/json" \
  -d '{"ip": "10.0.1.1", "port": 3306, "name": "db-prod-master", "ssh_key_id": 1}' \
  https://pmacontrol.example.com/api/v1/servers

# Ajouter le slave
curl -X POST -H "Authorization: Bearer $TOKEN" \
  -H "Content-Type: application/json" \
  -d '{"ip": "10.0.1.2", "port": 3306, "name": "db-prod-slave-01", "ssh_key_id": 1}' \
  https://pmacontrol.example.com/api/v1/servers
```

PmaControl lance un cycle de collecte dans la minute qui suit. Le slave apparaît alors avec un badge "Slave" dans le dashboard.

La page Slave dans PmaControl

La supervision de réplication est accessible via la route :

```
/{lang}/slave/show/{id}/{name}/
```

Par exemple : `/fr/slave/show/42/db-prod-slave-01/`

Le contrôleur `Slave.php` expose deux actions principales :

- **`show()`** : la page principale avec l'état courant et le graphique de lag

- `showGraphDay()` : les données AJAX pour charger un jour de graphique supplémentaire

Variables surveillées

PmaControl collecte toutes les variables de `SHOW SLAVE STATUS` et les stocke dans les tables time series. Les plus critiques :

Variable	Signification
`Slave_IO_Running`	Le thread IO est-il actif (connexion au master)
`Slave_SQL_Running`	Le thread SQL est-il actif (application des events)
`Seconds_Behind_Master`	Le lag en secondes
`Using_Gtid`	Mode GTID utilisé (MariaDB)
`Auto_Position`	GTID auto-position (MySQL)
`Last_SQL_Error`	Dernière erreur SQL rencontrée
`Relay_Log_Space`	Taille du relay log en cours

Pour MySQL 8.0+, PmaControl gère aussi les équivalents renommés :

Ancienne variable	Nouvelle variable (MySQL 8.0+)
`Slave_IO_Running`	`Replica_IO_Running`
`Slave_SQL_Running`	`Replica_SQL_Running`
`Seconds_Behind_Master`	`Seconds_Behind_Source`

PmaControl normalise automatiquement les deux noms en interne.

Le graphique de lag

Le graphique de lag est le coeur de la page slave. Il affiche les **5 derniers jours** de `Seconds_Behind_Master` sous forme de courbe Chart.js.

Caractéristiques :

- **Résolution** : un point par minute (1440 points par jour)

- **Chargement progressif** : le jour courant est chargé immédiatement, les jours précédents via AJAX "Load previous day"
- **Échelle automatique** : l'axe Y s'adapte au lag maximum observé
- **Zone verte** : < 10 secondes, fonctionnement normal
- **Zone ambrée** : 10-60 secondes, lag modéré
- **Zone rouge** : > 60 secondes, lag critique

La bande de santé (Health Strip)

Au-dessus du graphique, une bande horizontale résume l'état de chaque minute sur 24h avec un code couleur :

Couleur	Condition
Vert	IO Running + SQL Running + lag < 60s
Ambre	IO Running + SQL Running + lag > 60s
Rouge	IO ou SQL stopped, ou erreur critique

C'est un indicateur visuel instantané : un coup d'oeil suffit pour voir si la dernière journée a été stable ou chaotique.

Actions correctives depuis PmaControl

La page slave ne se limite pas à l'observation. PmaControl permet d'agir directement sur la réplication.

START / STOP SLAVE

Deux boutons pour démarrer ou arrêter la réplication. STOP SLAVE est utile pour :

- Effectuer une maintenance sur le slave (ALTER TABLE lourd)
- Prendre un backup cohérent (snapshot avec réplication arrêtée)
- Diagnostiquer un problème de lag (arrêter pour examiner le binlog)

SKIP error

Quand la réplication s'arrête sur une erreur SQL (contrainte dupliquée, table manquante), PmaControl propose de **sauter l'événement** :

```
SET GLOBAL sql_slave_skip_counter = 1;
START SLAVE;
```

Attention : cette action requiert une confirmation explicite. Sauter un événement signifie accepter une divergence entre master et slave. PmaControl loggue l'action avec l'utilisateur, la date, et l'erreur sautée pour traçabilité.

Parallel workers (réplication parallèle)

PmaControl propose un **slider** pour ajuster le nombre de workers de réplication parallèle :

- **Minimum** : 1 (réplication séquentielle classique)
- **Maximum** : 50 ou CPU × 2 (le plus petit des deux)

```
STOP SLAVE;
SET GLOBAL slave_parallel_workers = 8;
START SLAVE;
```

Augmenter les workers permet de rattraper un lag plus rapidement, car plusieurs transactions sont appliquées en parallèle. Mais attention : cela ne fonctionne bien qu'avec la réplication parallèle par LOGICAL_CLOCK (MariaDB) ou WRITESET (MySQL 8.0+).

Algorithme ETA de rattrapage

Quand un slave a du lag, la question immédiate est : **combien de temps pour rattraper ?**

PmaControl calcule une estimation (ETA) basée sur :

1. Le lag actuel en secondes
2. La vitesse de rattrapage observée (variation du lag sur les 10 dernières minutes)
3. Extrapolation linéaire

```
Si le lag passe de 3600s à 3000s en 10 minutes :
  Vitesse = 600s rattrapées / 10min = 60s/min
  ETA = 3000s / 60s/min = 50 minutes
```

L'ETA est affichée en haut de la page slave avec une barre de progression. Si la vitesse de rattrapage est nulle ou négative (le lag augmente), PmaControl affiche "ETA: divergent" en rouge — signe qu'il faut intervenir.

Support GTID

PmaControl détecte automatiquement le mode GTID :

- **MariaDB** : lecture de `Using_Gtid` dans `SHOW SLAVE STATUS`. Valeurs possibles : `No`, `Slave_Pos`, `Current_Pos`
- **MySQL** : lecture de `Auto_Position` dans `SHOW SLAVE STATUS`. Valeurs possibles : `0`, `1`

Quand GTID est actif, PmaControl affiche des informations supplémentaires :

- Le GTID set du master (`Gtid_Slave_Pos` ou `Executed_Gtid_Set`)
- Le GTID set du slave
- Le delta entre les deux (nombre de transactions de retard)

Le delta GTID est souvent plus parlant que `Seconds_Behind_Master` : il donne le nombre exact de transactions à rattraper, indépendamment de la durée de chaque transaction.

Bonnes pratiques

1. Toujours utiliser GTID

Le mode position binlog (file + position) est fragile : un fichier purgé trop tôt, un failover, et la réplication est cassée. GTID est idempotent et survit aux failovers.

2. Activer `read_only` sur les slaves

```
SET GLOBAL read_only = ON;
SET GLOBAL super_read_only = ON;  -- MySQL 5.7.8+ / MariaDB 10.3.16+
```

Sans `read_only`, une écriture accidentelle sur le slave provoque une divergence silencieuse.

3. Monitorer le lag, pas seulement l'état

`Slave_IO_Running: Yes` et `Slave_SQL_Running: Yes` ne suffisent pas. Un slave peut être "en marche" mais avec 2 heures de lag. PmaControl surveille les deux : l'état ET le lag.

4. Configurer les alertes

Dans PmaControl, configurez des seuils d'alerte :

- **Warning** : lag > 60 secondes pendant 5 minutes
- **Critical** : lag > 300 secondes OU IO/SQL stopped

Les alertes sont envoyées via Telegram avec le nom du serveur, le lag actuel, et le lien direct vers la page slave.

5. Planifier les gros traitements

Les `ALTER TABLE` sur des tables volumineuses génèrent un pic de lag temporaire. Utilisez `pt-online-schema-change` ou `gh-ost` pour les DDL en production, et prévenez PmaControl en mettant le slave en maintenance pour éviter les faux positifs.

Conclusion

La réplication MariaDB / MySQL est simple à configurer mais difficile à maintenir sur la durée. PmaControl comble le gap entre "la réplication tourne" et "la réplication est saine" en fournissant :

- Une vue temps réel du lag avec historique 5 jours
- Des actions correctives intégrées (start/stop, skip, parallel workers)
- Une estimation de rattrapage (ETA) pour anticiper
- La détection GTID automatique
- Des alertes proactives via Telegram

L'objectif n'est pas de remplacer le DBA — c'est de lui donner les outils pour réagir en minutes au lieu d'heures.