

# Galera : comprendre le Flow Control

Sylvain ARBAUDIE · 28 octobre 2024

GALERA MARIADB FLOW-CONTROL CLUSTERING TUNING

## GALERA FLOW CONTROL — THE CLUSTER HANDBRAKE

recv queue > fc\_limit → FC\_PAUSE → cluster writes blocked



### EXPERT TUNING RULES

`slave_threads = 2 x CPU`  
Drain recv queue faster

`fc_limit = 5 x threads`  
Enough room for parallelism

`fc_factor = 0.8`  
Progressive resume, no oscillation

`wsrep_flow_control_paused > 0.10 = serious problem — investigate immediately`

Flow Control is the guardian of Galera consistency — understand it, tune it, monitor it

## Le problème fondamental

Dans un cluster Galera, tous les nœuds doivent appliquer les mêmes writesets dans le même ordre pour maintenir la cohérence. Mais tous les nœuds ne sont pas égaux : certains sont plus rapides (hardware récent, charge légère), d'autres plus lents (hardware ancien, requêtes lourdes).

Que se passe-t-il quand un nœud lent ne peut pas suivre le rythme des écritures ? Les writesets s'accumulent dans sa file d'attente de réception (recv queue). Si rien n'est fait, cette file grossit indéfiniment, consommant toute la mémoire, et le nœud finit par crasher ou par diverger du cluster.

Le Flow Control est le mécanisme qui empêche cette situation. C'est le frein à main du cluster : quand un nœud est débordé, il demande aux autres de ralentir.

## Comment fonctionne le Flow Control

Le Flow Control repose sur un seuil simple : `gcs.fc_limit`.

Chaque nœud Galera maintient une file d'attente de réception (recv queue) qui stocke les writesets en attente d'application. Quand la taille de cette file dépasse `gcs.fc_limit`, le nœud

envoie un message FC\_PAUSE à tous les autres nœuds du cluster.

À réception de FC\_PAUSE, les autres nœuds cessent d'envoyer de nouveaux writesets au nœud lent. Les écritures sur le cluster entier sont bloquées — c'est le prix de la cohérence synchrone.

Quand la recv queue du nœud lent redescend en dessous de `gcs.fc_limit × gcs.fc_factor`, le nœud envoie un message FC\_CONTINUE et le cluster reprend son fonctionnement normal.

## Les 5 variables wsrep critiques

---

Pour monitorer le Flow Control, cinq variables de statut sont essentielles :

```
SHOW GLOBAL STATUS WHERE Variable_name IN (  
    'wsrep_local_recv_queue',  
    'wsrep_local_recv_queue_avg',  
    'wsrep_flow_control_paused',  
    'wsrep_flow_control_paused_ns',  
    'wsrep_flow_control_sent'  
);
```

### **wsrep\_local\_recv\_queue**

Taille actuelle de la recv queue. En fonctionnement normal, cette valeur devrait être proche de 0. Si elle monte régulièrement au-dessus de `gcs.fc_limit`, le nœud est en difficulté.

### **wsrep\_local\_recv\_queue\_avg**

Moyenne mobile de la recv queue. C'est l'indicateur le plus fiable pour détecter les tendances. Une moyenne au-dessus de 0,5 mérite investigation.

### **wsrep\_flow\_control\_paused**

Fraction du temps passé en Flow Control (entre 0 et 1). Si cette valeur dépasse 0,1 (10% du temps), le cluster a un problème de performance sérieux.

### **wsrep\_flow\_control\_paused\_ns**

Temps total passé en Flow Control en nanosecondes. Utile pour calculer le temps absolu de pause sur une période.

### **wsrep\_flow\_control\_sent**

Nombre de messages FC\_PAUSE envoyés par ce nœud. Si un seul nœud envoie la majorité des FC\_PAUSE, c'est le goulot d'étranglement à traiter.

## Les 6 paramètres de tuning

---

### **gcs.fc\_limit**

Le seuil de déclenchement du Flow Control. Valeur par défaut : 16. Augmenter cette valeur tolère plus de lag avant de déclencher le frein, mais augmente la consommation mémoire.

### **gcs.fc\_factor**

Le coefficient de reprise. Valeur par défaut : 0,5. Quand la recv queue redescend à `fc_limit × fc_factor`, le Flow Control est relâché. Avec un `fc_limit` de 100 et un `fc_factor` de 0,8, le FC se relâche à 80 writesets.

### **wsrep\_slave\_threads**

Nombre de threads d'application des writesets. Plus de threads = application plus rapide des writesets reçus = recv queue qui se vide plus vite. Recommandation : 2 × nombre de cœurs CPU.

### **wsrep\_cert\_deps\_distance**

Distance moyenne de certification entre les transactions. Indique le parallélisme potentiel. Si cette valeur est élevée, augmenter `wsrep_slave_threads` aura un impact positif.

### **gcs.recv\_q\_hard\_limit**

Limite absolue de la recv queue en octets. Si dépassée, le nœud est avorté. C'est le dernier recours pour éviter un OOM. Recommandation : la moitié de la RAM + swap disponible.

### **gcs.max\_throttle**

Débit minimum garanti même en Flow Control (entre 0 et 1). La valeur par défaut de 0,25 signifie que même en FC, 25% du débit normal est maintenu. Mettre à 0 pour un arrêt complet en FC.

## Recommandations d'experts

---

Après des années de gestion de clusters Galera en production, voici les recommandations consolidées :

## Dimensionnement des slave threads

```
wsrep_slave_threads = 2 × CPU_CORES
```

Si votre serveur a 8 cœurs, commencez avec `wsrep_slave_threads = 16`. Surveillez `wsrep_cert_deps_distance` — si elle est inférieure au nombre de slave threads, réduisez.

## fc\_limit en fonction des slave threads

```
gcs.fc_limit = 5 × wsrep_slave_threads
```

Avec 16 slave threads, `gcs.fc_limit = 80`. Cela donne assez de marge aux threads pour travailler en parallèle sans déclencher le FC trop tôt.

## fc\_factor pour une reprise progressive

```
gcs.fc_factor = 0.8
```

Un `fc_factor` de 0,8 (au lieu du défaut 0,5) permet une reprise plus progressive du trafic, évitant les oscillations `FC_PAUSE / FC_CONTINUE`.

## Hard limit pour la sécurité

```
gcs.recv_q_hard_limit = HALF_RAM_PLUS_SWAP
```

Sur un serveur avec 32 Go de RAM et 16 Go de swap, mettez `gcs.recv_q_hard_limit = 24G`. C'est le filet de sécurité contre les OOM.

## Identifier le nœud problématique

Quand le Flow Control se déclenche fréquemment, il faut identifier le nœud lent :

```
-- Sur chaque nœud
SELECT @@hostname,
       VARIABLE_VALUE AS fc_sent
```

```
FROM information_schema.GLOBAL_STATUS
WHERE VARIABLE_NAME = 'wsrep_flow_control_sent';
```

Le nœud qui envoie le plus de FC\_PAUSE est le goulot. Les causes habituelles :

- **Hardware inférieur** : disques plus lents, moins de RAM
- **Requêtes lourdes** : un ALTER TABLE ou un SELECT massif qui monopolise les ressources
- **Sauvegarde en cours** : mariabackup/xtrabackup consomme beaucoup d'I/O
- **Charge applicative déséquilibrée** : trop de lectures sur un nœud qui doit aussi appliquer les writesets

## Conclusion

---

Le Flow Control est le gardien de la cohérence dans Galera. Comprendre son fonctionnement et ses paramètres est essentiel pour maintenir un cluster performant.

Les règles d'or : `slave_threads = 2 × CPU` , `fc_limit = 5 × threads` , `fc_factor = 0.8` , `hard limit = moitié RAM + swap`. Monitorisez `wsrep_flow_control_paused` et réagissez dès que la valeur dépasse 10%.

---

Cet article a été initialement publié sur [Medium](#).