Runbook — Arrêt & Démarrage d'un cluster Kafka Confluent (3 brokers)

Procédure détaillée et sécurisée pour arrêter et démarrer un cluster Kafka Confluent sans perte de données ni dégradation.

0. Pré-requis

• Facteur de réplication ≥ 3 et min.insync.replicas ≥ 2
• Compte admin MDS (RBAC)
• Accès aux brokers (systemctl ou scripts)
• Vérifier qu'aucun rebalance n'est en cours
• Connaître le contrôleur actuel

1. Vérification de la santé du cluster

```
kafka-topics --bootstrap-server broker1:9092 --describe | grep -i isr
kafka-metadata-quorum describe --bootstrap-server broker1:9092
```

■ Attendre que toutes les partitions soient en ISR complet.

2. Préparation (facultatif mais recommandé)

```
kafka-preferred-replica-election --bootstrap-server broker1:9092
confluent-rebalancer execute --bootstrap-server broker1:9092 --throttle 100000000
```

■ Cette commande permet de rééquilibrer les leaders entre les brokers.

Explication :
Chaque partition Kafka a un leader et plusieurs followers. Si trop de leaders sont concentrés sur un seul broker, l'arrêt de ce broker provoquera un fort impact (beaucoup de partitions sans leader d'un coup).

Avant élection :
• B1 = 60 leaders
• B2 = 30 leaders
• B3 = 10 leaders
■ Si B1 tombe, 60 leaders doivent basculer = gros impact.

Après élection (équilibré) :
• B1 = 33 leaders
• B2 = 33 leaders
• B3 = 34 leaders
■ Si B1 tombe, seulement 33 leaders doivent basculer = impact limité.

■ En résumé : **répartir mieux les leaderships avant arrêt limite fortement l'impact** lors d'un rolling shutdown.

3. Arrêt progressif (rolling shutdown)

Arrêter les brokers un par un, en commençant par ceux qui ne sont pas contrôleur.

```
sudo systemctl stop confluent-kafka
# ou
bin/kafka-server-stop
```

Puis vérifier la stabilité ISR :

```
kafka-topics --bootstrap-server broker2:9092 --describe | grep -v "isr=.*"
```

■■ Contrôleur toujours en dernier pour limiter les élections.

4. Vérifications après arrêt

• Brokers arrêtés (systemctl status)
• Logs indiquant un shutdown complet
• Données persistées dans /var/lib/kafka/data

5. Démarrage progressif (rolling start)

Redémarrer les brokers un par un, en commençant par ceux qui ne sont pas contrôleur.

```
sudo systemctl start confluent-kafka
# ou
bin/kafka-server-start -daemon config/server.properties
```

Après chaque démarrage :

```
kafka-topics --bootstrap-server broker1:9092 --describe | grep -i isr
kafka-metadata-quorum describe --bootstrap-server broker1:9092
```

■ Vérifier que le broker rejoint l'ISR et que le cluster garde un seul contrôleur.

6. Différence : démarrage initial vs rolling restart

Démarrage initial (cluster totalement arrêté)
• L'ordre des starts est moins critique, car un contrôleur sera élu dès le premier broker démarré.
• Bonne pratique : démarrer un broker à la fois pour stabiliser l'ISR progressivement.

Rolling restart (cluster partiellement en ligne)
• L'ordre est critique : il faut garder le contrôleur pour la fin afin d'éviter plusieurs élections successives.
• Cela garantit moins d'instabilité et une seule élection contrôleur.

7. Étapes complémentaires

• Notifier les utilisateurs / équipes
• Exporter la config (backup)
• Surveiller metrics : UnderReplicatedPartitions=0, ActiveControllerCount=1, OfflinePartitionsCount=0
• Faire un test de production/consommation de message

■ Résumé : équilibrer les leaderships (preferred replica election), arrêter broker par broker (contrôleur en dernier), vérifier ISR entre chaque étape, redémarrer progressivement, et distinguer démarrage initial (ordre souple) de rolling restart (ordre critique).