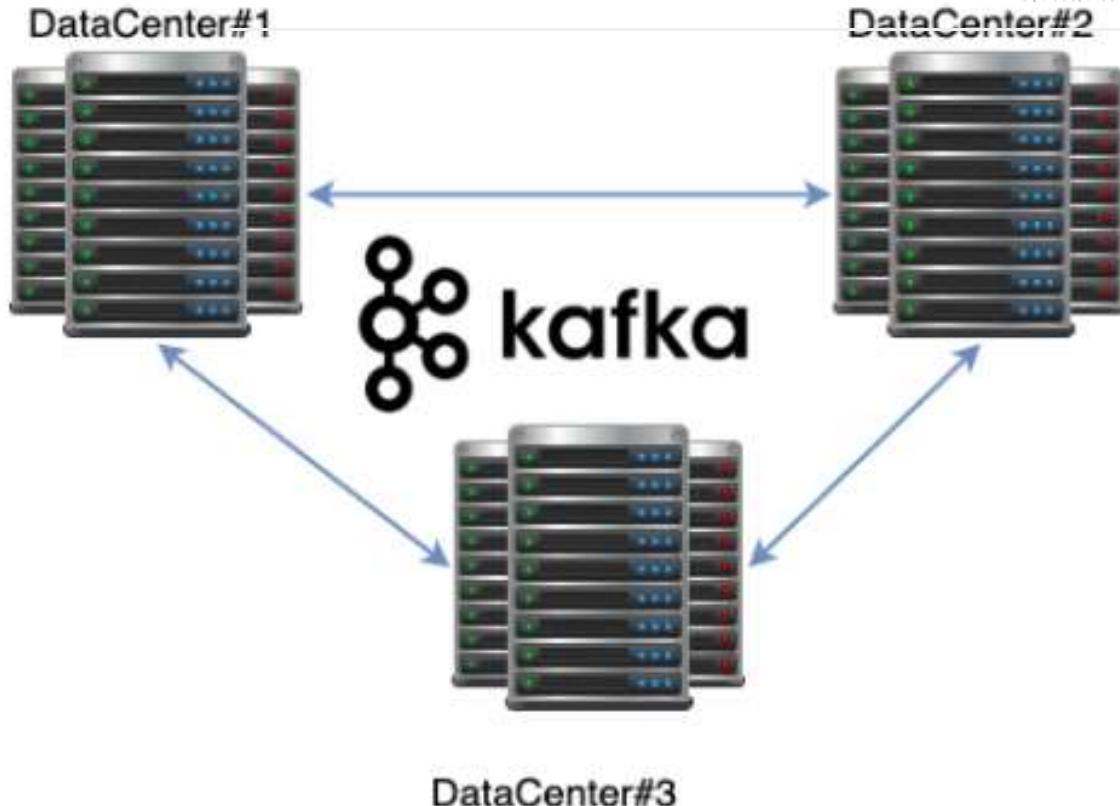


KAFKA (/TAG/KAFKA/) BIGDATA (/TAG/BIGDATA/)
MODERN ARCHITECTURE (/TAG/MODERN-ARCHITECTURE-2/)



Kafka dans un environnement multi-datacenter

👤 Mohamed HILIA (/author/mohamed/) 📅 26 Jun 2020

🗨 0 Commentaires (/2020/06/26/kafka-dans-un-environnement-multi-datacenter/#disqus_thread)

Cet article présente les motivations de mise en place d'une plateforme Kafka dans un contexte multi-datacenter. Il donne un aperçu des choix architecturaux en mettant en avant leurs avantages et inconvénients ainsi que des préconisations.

Contexte

Dans le paysage technologique de la data, la collecte, le stockage et le traitement des données en temps réel gagnent de plus en plus de terrain (p. ex. IoT, industrie 4.0) par rapport à des traitements des données par lots. Les données sont souvent générées dans des localisations géographiquement distribuées (i.e. Régions Cloud), à cela s'ajoute le besoin de traitements et d'analyses de l'ensemble des données en flux à un niveau fédéré ou à un niveau global. D'où le besoin d'avoir des données hautement disponibles, d'avoir une réPLICATION inter-datacenter sans oublier les plans pour faire face à la reprise sur défaillance d'un datacenter et ou d'une région.

Cet article répond à ces exigences de distribution et de communication des données à un niveau global et plus particulièrement sur trois régions, à savoir : l'Europe, l'US et l'APAC. Ces données sont collectées, traitées et mises à disposition pour réaliser les futurs cas d'utilisation.



Motivation

L'organisation de la plateforme globale en plusieurs régions peut être motivée par plusieurs raisons organisationnelles, opérationnelles ou légales. Avoir une plateforme sur plusieurs régions permet aux différentes régions une autonomie opérationnelle et une isolation des ressources. Par ailleurs, voici d'autres contextes où une architecture en multi-datacenter est une solution à envisager :

- Certaines contraintes techniques et organisationnelles imposent une séparation physique en plusieurs clusters.
- Reprise après défaillance (DR = Disaster Recovery).
- Séparation des données/applications critiques avec des fortes contraintes de performance.
- Personnalisation et tuning des configurations au niveau des clusters pour répondre à plusieurs SLA.
- Structure d'organisation des équipes et commodités.
- Données qui ne doivent pas quitter une région.
- "Data geo-locality" des données qui doivent rester proches des utilisateurs à des fins d'analytiques régionaux.
- Raisons légales :
- Différentes lois de stockage des données par pays/région,
- Partage des données,
- Conformité,
- Loi sur les méthodes de chiffrements des données par pays/région.

Le cluster Kafka se compose d'une liste de Brokers. Dans la plupart des cas, le cluster est déployé dans un datacenter ou sur plusieurs datacenter, d'où le déploiement multi-datacenter. Par ailleurs, ces datacenter peuvent être dans la même région au sens Provider Cloud ou sur plusieurs régions.

Le choix de partir sur un déploiement géo-distribué n'est que le début de l'aventure !

Comme on l'a vu, dans certains scénarios nous avons besoin de plusieurs clusters. La réponse à ces exigences peut être simple, à savoir : plusieurs clusters gérés de manière complètement séparée et indépendante. Mais, n'oublions pas notre contrainte forte d'avoir une plateforme globale en temps-réel, ce qui nécessite la réPLICATION inter-cluster des données. Comme la réPLICATION d'un cluster source vers un cluster destination est appelée, "*Mirroring*", alors "*Replication*" désigne la copie des données intra-cluster.

Les configurations par défaut des clusters Kafka sont optimisées pour un fonctionnement à faible latence, et une bande passante élevée (ex. 1 Gbps) entre les clients et les brokers. Le coût de transfert des données entre clusters est à prendre en compte, en particulier dans un contexte Cloud où il peut rapidement être élevé.

Cette section présente plusieurs modèles d'architectures pour le déploiement de Kafka en multi-datacenter, dont celle que nous avons choisie dans le cadre de notre projet. Nous irons même occasion les notions suivantes :



- Mirroring et prévention de la répétition cyclique de messages,
- Gestion des offsets,
- Centralisation du schéma de données.

Modèles d'architectures

Nous sommes tous conscients des problématiques de latence et des coupures réseaux possibles dans un contexte multi-région, et ce que cela peut induire pour les équipes opérationnelles.

Stretch Clusters

Le premier modèle d'architecture "Stretch Cluster" (Fig. 1) n'est pas multi-cluster, mais un seul cluster Kafka déployé sur plusieurs datacenters. La réPLICATION native inter-cluster joue le rôle du « Mirroring ». Cette approche ne nécessite donc aucun composant supplémentaire pour répliquer les données sur plusieurs sites géographiques.

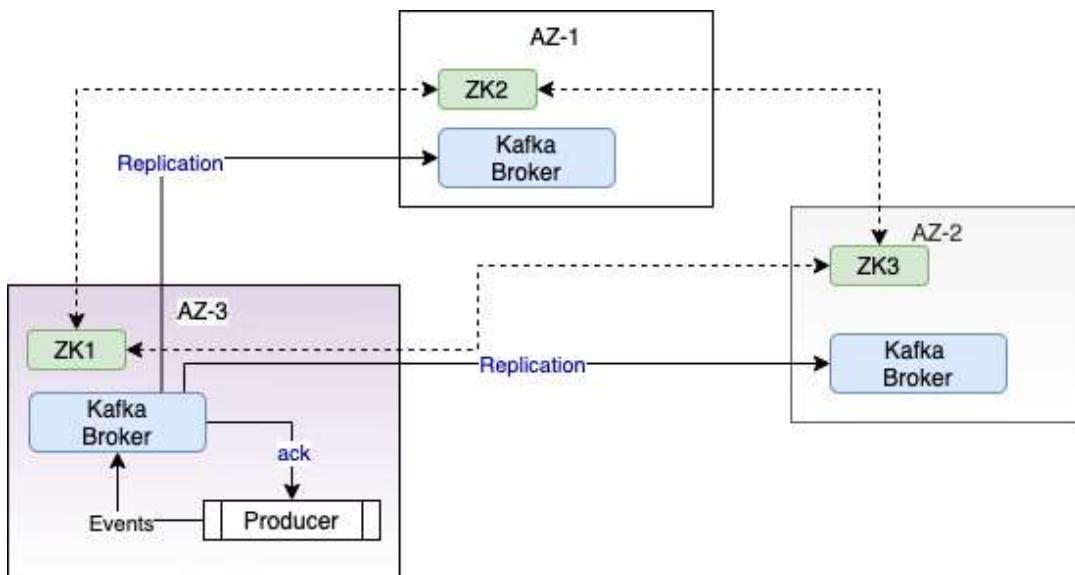


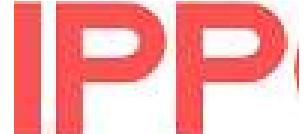
Fig 1 : Stretch model

Cette architecture nécessite trois datacenters proches (p. ex. Availability Zones d'un fournisseur Cloud, et non pas Région.) avec une latence à un chiffre (<10ms). Ce modèle a besoin, pour assurer un bon fonctionnement, de déployer au moins un Zookeeper et un broker par datacenter, un "acks=all" et "min.isr=2" ainsi que la définition de chaque datacenter comme un "rack".

Avantages :

- Cohérence forte des données, la réPLICATION peut être synchrone entre les différents clusters. Le message sera commité avec succès sur au moins deux datacenters.
- Le DR assure toujours une synchronisation 100 % d'au moins deux datacenters.

- L'utilisation de l'ensemble des ressources allouées contrairement au modèle «active-standby».
- Facile à déployer.



Discovery t

Limitations :

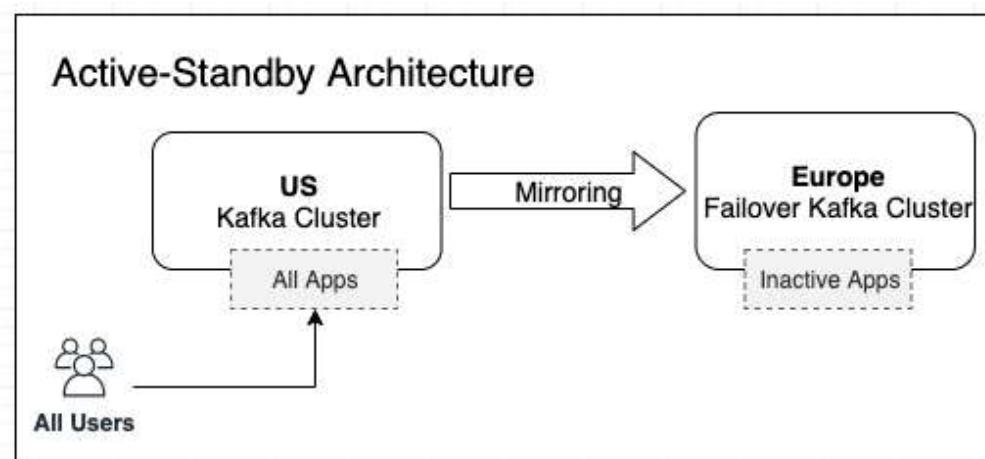
- Ce modèle ne protège que contre les défaillances des datacenters, et non contre la défaillance des applications ou de Kafka.
- Nécessité de trois datacenters proches (p. ex. Des AZ dans la même région cloud).
- Avoir une latence à un chiffre entre les datacenters.
- Une panne du cluster est un désastre.
- Haute latence et débit réduit par rapport à un cluster classique.
- Le trafic entre les datacenter peut présenter un goulot d'étranglement.

Dans notre situation, ce n'est pas la solution la plus optimale. Nos datacenters de déploiement sont sur plusieurs régions Cloud. Avoir un seul cluster n'est pas rassurant et les communications peuvent souffrir d'une grande latence.

Rappelons que nous avons besoin de répartir nos clusters sur les trois régions EU, US et APAC. Un mécanisme de "Mirroring" est nécessaire pour la réPLICATION d'événements entre les clusters déployés dans chacun des régions. Le Mirroring est assuré en mode asynchrone. Kafka dispose d'un outil permettant le Mirroring des événements inter-cluster, à savoir, le "Mirror Maker". Vous pouvez aussi utiliser la version proposée par Confluent nommé le Replicator (<https://docs.confluent.io/current/connect/kafka-connect-replicator/index.html>) qui apporte certaines fonctionnalités supplémentaires par rapport à MirrorMaker comme la propagation des metadata des topics.

Active-Passive

Le modèle "active-passive" dit aussi "active-standby" consiste en une paire de cluster. Le premier cluster est "actif", il gère l'ensemble des requêtes et les applications actives. Les données sont ensuite répliquées vers un cluster "passif" de manière asynchrone. Cette réPLICATION est mono-directionnelle, les applications clientes basculent vers le cluster passif uniquement en cas de défaillance du cluster "actif".



- Simple mirroring unidirectionnel
- La bande passante entre les clusters n'affecte pas la performance
- Besoin de moins de trafic réseaux.
- Nécessite moins d'effort de supervision

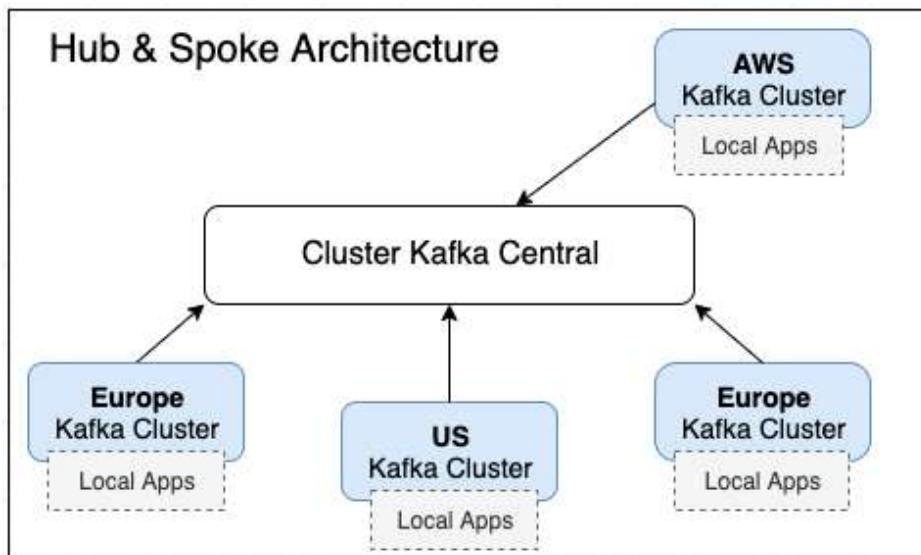


Limitations :

- Cohérence à terme suite au mirroring asynchrone.
- Durée de reprise (Recovery Time Objective)
- Les ressources passives ne sont utilisables qu'en cas de panne
- Possibilité de perte de données à cause de défaillance avant le mirroring asynchrone.

Hub & Spoke

Ce modèle permet d'avoir une instance de cluster agrégée pour la réPLICATION de données des clusters indépendants. Ce modèle prévoit un cluster central pour une synchronisation entrante de l'ensemble des autres clusters. Le cluster central représente un point d'agrégation des données partielles. Les applications qui ont besoin d'un accès à l'ensemble des données peuvent être hébergées dans le cluster central. Les données sont donc produites dans les différents clusters.



Avantages :

- Les données sont produites localement avec une seule direction de réPLICATION vers le cluster central.
- Séparation des applications locales et des applications globales
- Simplicité de déploiement et supervision

Limitations :

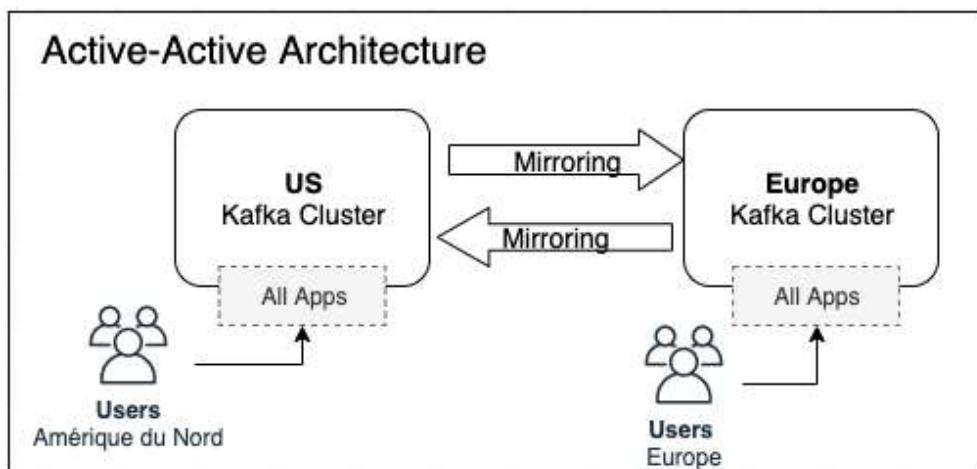
- Visibilité partielle des données pour les applications qui nécessitent des données d'une autre région pour des applications à forte valeur ajoutée globale. (exemple, affiner l'expérience utilisateur entre deux plateformes géographiques)
- Synchronisation et partage d'information sur les données répliquées comme les topics, ou prévoir un préfixe pour ne pas dupliquer les données.

NOTE : Si le même topic existe dans plusieurs datacenters, vous pouvez écrire tous les événements de ce topic sur un seul topic avec le même nom dans le cluster central, ou écrire les événements dans chaque datacenter dans un topic distinct.



Active-Active

L'objectif de ce modèle d'architecture est de maintenir une connectivité et une cohérence entre les données des deux clusters, ce modèle permet une utilisation de l'ensemble des ressources allouées et une réPLICATION asynchrone bidirectionnelle. Les données sont répliquées dans les deux directions, les applications peuvent passer d'un cluster à l'autre en cas de défaillance. Contrairement au modèle Actif-passif, les requêtes des clients sont traitées par les deux clusters. Chacun avec ses propres clients, avec une autonomie opérationnelle. Des contraintes sur les données répliquées peuvent être appliquées, comme le filtrage sur les données à répliquer.



Le mirroring est obligatoire dans les deux directions, de préférence via un lien sécurisé.

Avantages :

- Les ressources sont pleinement utilisées dans les deux clusters.
- Aucun temps d'arrêt en cas de défaillance d'un seul cluster.
- La bande passante du réseau entre les clusters n'affecte pas les performances.
- Les clusters sont quasi-identiques.
- Des bonnes performances avec la proximité des clients aux datacenters.
- Redondance et résilience de la solution étant donné que chacun des clusters sont quasi-identiques.
- Transparence de la reprise sur défaillance.

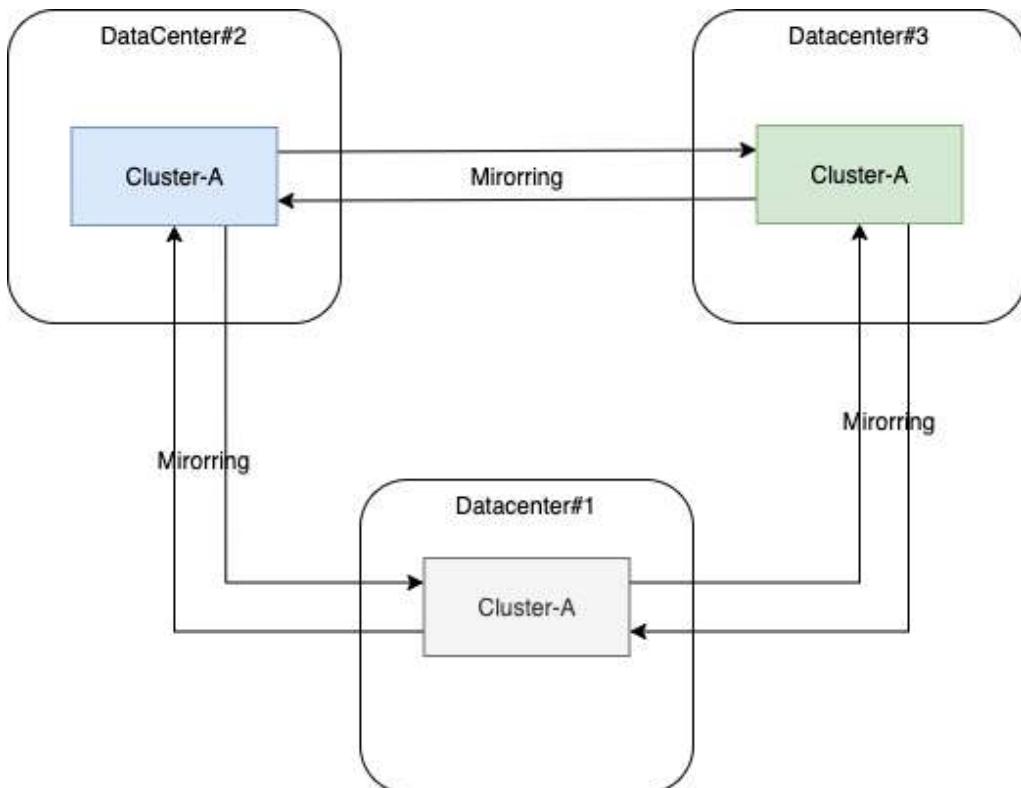
Limitations :

- Cohérence à terme due au mirroring asynchrone entre les clusters.
- Complexité de la mise en réPLICATION bidirectionnelle entre les clusters
- Perte de données possible en cas de défaillance d'un cluster en raison d'un mirroring asynchrone en retard.

Le modèle actif-actif s'appuie sur un composant de mirroring qui permet la configuration des données à répliquer et le choix de la direction. Voici les principales options pour ce composant de "mirroring" :^

- uReplicator (<https://github.com/uber/uReplicator>) : Uber
- MirrorMaker (https://kafka.apache.org/documentation/#basic_ops_mirror_maker) : Apache Kafka
- Confluent Replicator (<https://docs.confluent.io/current/connect/kafka-connect-replicator/index.html>): Confluent Enterprise edition.

Ce dernier modèle permet la réalisation de modèles avancés d'architecture en multi-datacenter comme la topologie en anneaux de plusieurs clusters déployés sur différents datacenters. Dans ce modèle en anneau, chaque cluster est connecté à exactement deux clusters.

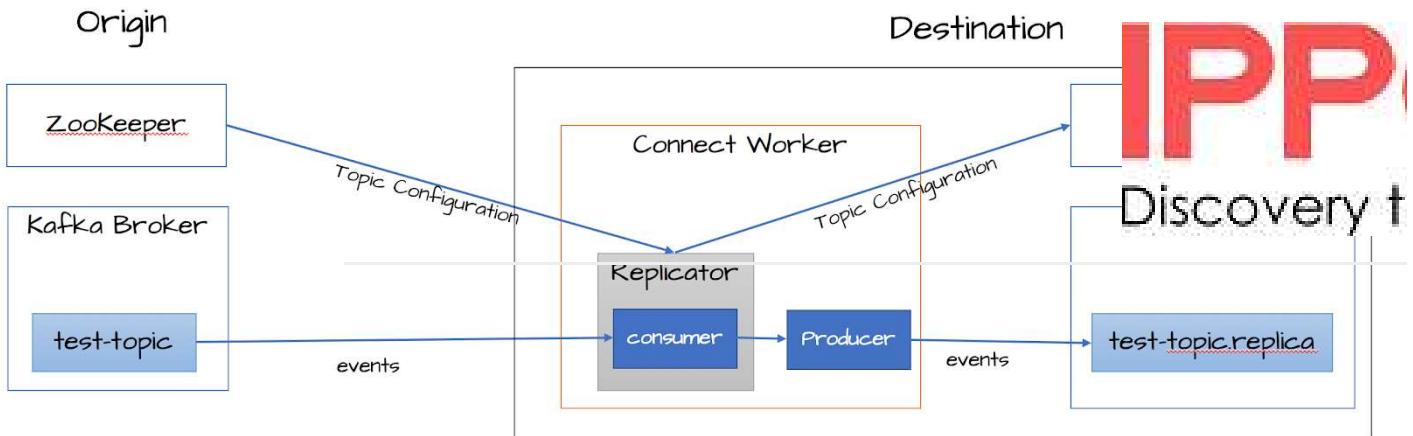


Le composant de Mirroring n'est autre qu'un ensemble de consommateurs et un producteur. Les consommateurs récupèrent les données d'un cluster Kafka source et le producteur se charge d'envoyer ces données vers un cluster Kafka cible.

Kafka dispose d'un MirrorMaker built-in pour la réPLICATION cross-cluster des données. Par ailleurs, ce dernier présente des limitations et une solution a été apportée par le composant Replicator de Confluent, par exemple :

- La propagation des mises à jour des topics
- La prévention de la réPLICATION cyclique
- La scalabilité horizontale en cas de pic de charge

D'autres éléments différenciants se trouvent dans ce comparatif : MirrorMaker vs Replicator (<https://docs.confluent.io/4.1.2/multi-dc-replicator/mirrormaker.html>)



Le Replicator permet d'empêcher la réPLICATION cyclique des topics par l'ajout automatique des informations sur le Header par la configuration de la clé "provenance.header.enable=true".

Il permet aussi de maintenir une liste de topics à répliquer et d'autres à écarter de ce processus de réPLICATION inter-clusters. Concrètement, le Replicator est un simple connecteur Kafka Connect.

Les configurations `topic.regex`, `topic.whitelist` et `topic.blacklist` s'appliquent dans l'ordre suivant :

1. Tous les topics spécifiés dans `topic.blacklist` ne seront pas répliqués.
2. Tous les topics spécifiés dans `topic.whitelist` ou correspondant à `topic.regex` seront répliqués (les topics de la list "whitelist" ne doivent pas nécessairement correspondre à l'expression "regex").

Si les configurations par défaut sont appliquées, aucun topic ne sera donc répliqué.

Replicator prend en charge la communication sécurisée avec Kafka via SSL pour les clusters sources et destinations. Il prend également en charge SSL ou SASL pour l'authentification. Vous pouvez configurer les connexions de Replicator à la source et à la destination de Kafka avec :

- SSL Authentication
(https://docs.confluent.io/current/kafka/authentication_ssl.html#authentication-ssl-replicator)
- SASL/SCRAM
(https://docs.confluent.io/current/kafka/authentication_sasl/authentication_sasl_scram.html#sasl-scram-replicator)
- SASL/GSSAPI
(https://docs.confluent.io/current/kafka/authentication_sasl/authentication_sasl_gssapi.html#sasl-gssapi-replicator)
- SASL/PLAIN
(https://docs.confluent.io/current/kafka/authentication_sasl/authentication_sasl_plain.html#sasl-plain-replicator)

Parmi les données répliquées, on trouve les événements envoyés dans des topics et également les métadonnées sur les topics comme le nombre de partitions d'un topic. Le replicator se charge de propager les métadonnées sur un topic et leurs changements du cluster source vers le cluster cible. Il ^

Partagez cet article:



f (<https://www.facebook.com/sharer/sharer.php?u=http://blog.ippon.fr/2020/06/26/kafka-dans-un-environnement-multi-datacenter/>)

twitter (<https://twitter.com/share?text=Kafka%20dans%20un%20environnement%20multi-datacenter&url=http://blog.ippon.fr/2020/06/26/kafka-dans-un-environnement-multi-datacenter/>)

g+ (<https://plus.google.com/share?url=http://blog.ippon.fr/2020/06/26/kafka-dans-un-environnement-multi-datacenter/>)

digg (<http://www.digg.com/submit?url=http://blog.ippon.fr/2020/06/26/kafka-dans-un-environnement-multi-datacenter/>)

reddit (<http://reddit.com/submit?url=http://blog.ippon.fr/2020/06/26/kafka-dans-un-environnement-multi-datacenter/&title=Kafka%20dans%20un%20environnement%20multi-datacenter>)

in (<http://www.linkedin.com/shareArticle?mini=true&url=http://blog.ippon.fr/2020/06/26/kafka-dans-un-environnement-multi-datacenter/>)



stumbleupon (<http://www.stumbleupon.com/submit?url=http://blog.ippon.fr/2020/06/26/kafka-dans-un-environnement-multi-datacenter&title=Kafka%20dans%20un%20environnement%20multi-datacenter>)



(/author/mohamed/)

Mohamed HILIA (/author/mohamed/)

Ippon

Ippon est un cabinet de conseil en technologies, créé en 2002 par un sportif de Haut Niveau et un polytechnicien, avec pour ambition de devenir leader sur les solutions Digitales, Cloud et BigData.

Ippon accompagne les entreprises dans le développement et la transformation de leur système d'information avec des applications performantes et des solutions robustes.

Ippon propose une offre de services à 360° pour répondre à l'ensemble des besoins en innovation technologique : Conseil, Design, Développement, Hébergement et Formation.

maintient la cohérence de configuration des topics entre les clusters source et cible. Si vous modifiez le nombre de partitions d'un topic dans un cluster source, le replicator réalise cette opération sur le cluster cible. Le MirrorMaker ne dispose pas de cette fonctionnalité de propagation.



Conclusion

Dans cet article, nous avons présenté les différents modèles d'architecture pour la mise en place de Kafka dans un contexte multi-datacenter et partagé les motivations pour choisir tel ou tel type de déploiement.

Nous avons retenu le modèle d'architecture active-active car il répond aux exigences de notre projet, à savoir, la haute disponibilité, la résilience, la performance, des traitements spécifiques aux régions et de la gouvernance régionale. Nous montrerons dans la suite de cette série d'articles, la gestion des schémas et des offsets dans ce cas précis de déploiement.

You have found this publication useful? Click here

LIVRE BLANC

Explore Big Data.

Ce livre blanc prend "un peu" de hauteur sur les technologies Big Data afin d'aider les décideurs à faire les bons choix techniques et à y voir plus clair sur les opportunités offertes par les différentes technologies.

[JE TÉLÉCHARGE \(HTTPS://FR.IPPON.TECH/EXPLORE-BIG-DATA/\)](https://fr.ippon.tech/explore-big-data/)

VIDÉO MÉTIER

Architecte Data @Ippon c'est ... (avec Arnaud Col)



REX CLIENTS

Ippon & ENGIE Digital : Accélération du produit eCare



IPP
Discovery t

POST RÉCENTS

Cucumber et RestTemplate

Aug 31, 2020

(<http://blog.ippon.fr/2020/08/31/cucumber-et-resttemplate/>)

REX, prise en main de Selenium WebDriver

Aug 20, 2020

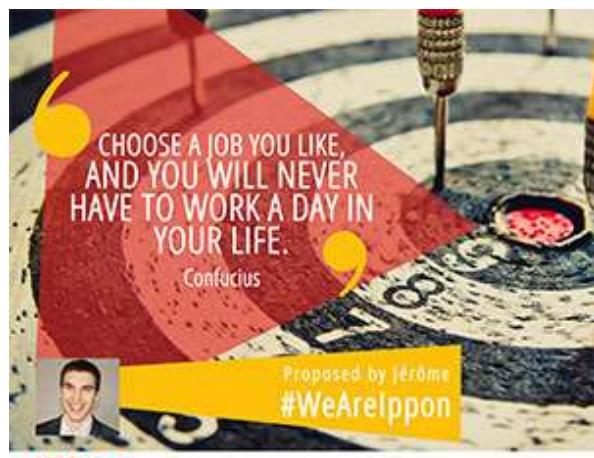
(<http://blog.ippon.fr/2020/08/20/rex-sur-la-prise-en-main-de-selenium-webdriver/>)

Construire la CI d'un monorepo: les parent-child pipelines de Gitlab-ci

Jul 31, 2020

(<http://blog.ippon.fr/2020/07/31/construire-la-ci-dun-monorepo-les-parent-child-pipelines-de-gitlab-ci/>)

WE ARE IPPON



(<http://www.ippon.fr/nous-rejoindre/>)



Ippon est un cabinet de conseil qui accélère les projets innovants des ses clients
page blanche au Cloud.

Nos équipes dans le monde accompagnent les organisations dans la transformation
d'idées innovantes en solutions logicielles de haute qualité avec un focus particulier sur
le Time To Market.

© 2020 IPPON (<http://blog.ippon.fr>). Tous les droits sont réservés.



Nous avons réalisé, en 2019, un chiffre d'affaires de 42 M€. Nous sommes aujourd'hui un groupe de plus de 400 consultants répartis en France, aux USA, en Australie et en Russie.



📍 FRANCE 🌐 WEBSITE (HTTP://WWW.IPPON.FR)
LinkedIn (HTTPS://WWW.LINKEDIN.COM/COMPANY/IPPON-TECHNOLOGIES)

Post précédent

Gestion automatisée de certificats TLS avec Let's Encrypt via Terraform et Ansible sur AWS

(/2020/06/29/gestion-automatisee-de-certificats-tls-avec-lets-encrypt-via-terraform-et-ansible-sur-aws/)

Poste suivant

O2 - Une app offline-first déclinable

(/2020/06/24/o2-une-app-offline-first-declinable/)

ÉGALEMENT SUR BLOG IPPON TECHNOLOGIES

CD : Snowflake, Sqitch et Gitlab il y a un an • 1 commentaire La construction d'un Data Warehouse est assez similaire au développement	80 ou 90% de couverture de tests il y a un an • 2 commentaires S'il est une question qui revient très souvent lorsqu'on parle de tests c'est le taux ...	Ceph - Technologie de stockage Haute ... il y a 7 mois • 2 commentaires Contexte Notre cluster de stockage Ceph ...	Nouvelles fonctionnalités de Spar il y a 1 mois • 1 commentaire La toute nouvelle fonctionnalité de Spar ...
---	---	--	---

0 Commentaires [Blog Ippon Technologies](#) [Règles de confidentialité de Disqus](#)

1 [S'identifier](#) ▾

[Recommander](#)

[Tweet](#)

[Partager](#)

[Les meilleurs](#) ▾