**Rapport sur la partie I du TER “traitement du flux de données horodatées” Hadja maimouna BALDÉ**

**Partie I : Etat de l’art sur les systèmes (architectures data / logicielle) de traitement de données horodatées.**

**A) Pourquoi les données horodatées.**

Le traitement de flux de données horodatées est conçu :

* pour le traitement instantané des données (c à d des données qui sont encore en mouvement) et leurs analyses en temps réel.
* pour enregistrer aussi l’instant auquel une opération a été effectuée.

L’objectif principal est de fournir des informations actualisées à la milliseconde sur les événements qui se produisent dans un système et de nous aider à réagir aux événements critiques dès qu’ils se produisent. De ce fait, ce traitement est très avantageux pour les entreprises, car ça nous permet d’avoir des informations en temps réel sur la santé d’un système.

Il existe différents domaines dans lesquels ces données sont utilisés telle que :

* La détection des fraudes : techniques d’analyse des données pour la détection des fraudes
* Intelligence artificielle : les données brutes sont traitées comme un ensemble d'entraînement/apprentissage et un ensemble de tests lors de la construction d'algorithmes d’IA et d'apprentissage automatique.

**B) Les solutions existantes pour traiter les données horodatées.**

**pour traiter ces données parmi les logiciels actuels on peut :**

Utiliser des frameworks car cela réduit le temps et les efforts nécessaires au développement de diverses applications de streaming qui sont :

* la collecte de données, notamment l’intégration multicloud et les données métier de flux et de messages ;
* la livraison à l’échelle de l’entreprise, y compris la distribution des données ainsi que la supervision et la détection des dérives de données ;
* la détection des anomalies en cours de flux ;
* l’agrégation sur le flux ;
* l’enrichissement sur le flux et la détection basée sur des règles ;
* la conformité des données ;
* la détection des fraudes financières et le repérage des activités criminelles en temps réel ;
* la supervision du système, grâce à l’analyse en temps réel et à la maintenance prédictive des serveurs physiques, du réseau, des applications ou des équipements industriels ;
* l’échange de titres algorithmique à haute vitesse ;

Utiliser aussi les systèmes de messagerie traditionnels tels que Kafka, qui a adopté l’approche de colocation du traitement et du stockage de ces données sur le même nœud.

**présentation kafka**

kafka implémente un modèle de messagerie de publication-abonnement qui offre une tolérance aux pannes, un service réputé, rapide, scalable, partitionné, répliqué, et une évolutivité permettant de gérer de gros volumes de données en streaming pour des analyses en temps réel

**Architecture Kafka**

Kafka possède un ensemble de composant assurant son fonctionnement en tant que système de messagerie distribué :

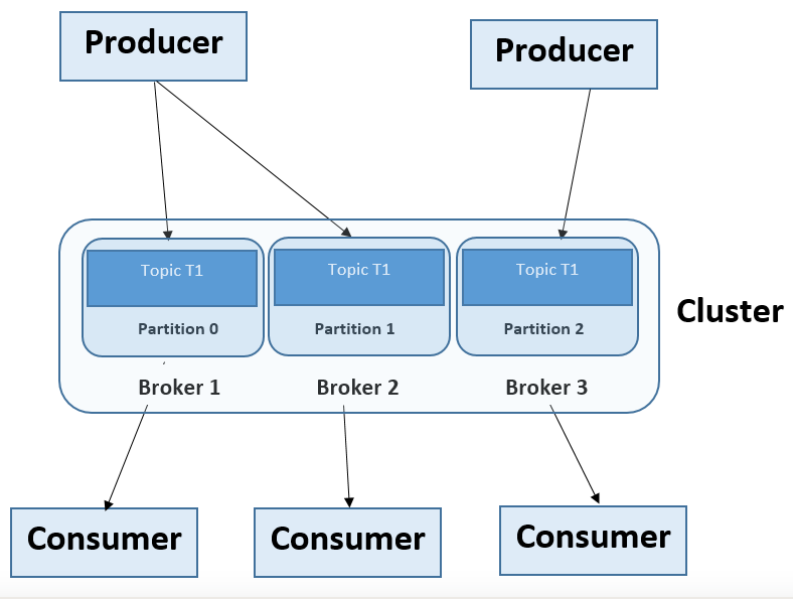


Figure 1 illustration du fonctionnement kafka

En claire :

* Topic : c’est ici où sont stockés ou enregistrés les données dans kafka
* Producteur : c’est le système qui va produire les données dans kafka, publie aussi les messages à un topic
* Broker : est un ensemble de serveur ou noeud formant un cluster kafka, permettant de stocker et gérer les messages publiés
* Consommateurs : peuvent s’abonner à un ou plusieurs topic et consomme les messages publiés à partir du cluster

Dans l’exemple de la figure 1, le topic T1 contenant N message est partitionné à travers 3 brokers du cluster kafka. Les consommateurs consomment toujours les messages à partir d’une partition spécifique, de façon séquentielle. Cela joue un rôle non négligeable dans les bonnes performances des traitements effectués par kafka. Avant toute consommation d’un message, chaque consommateur lance une requête asynchrone de type pull au broker afin que celui-ci lui alloue un tampon d’octets prêts à être consommé.

utiliser aussi influxDB ou KDB comme base de données les plus populaires pour traiter ces série temporelles (données horodaté). Ces bases de données sont conçues pour stocker et analyser des données de capteurs ou des journaux horodatés sur une période donnée. De même des outils de data visualisation comme prometheus ou graphite on vu le jour pour intégrer ces données horodaté au sein de tableau de bord, l’un des principaux objectifs est de détecter une panne sur un objet connecté

**Architecture influxDB**

InfluxDB fournit un agent de serveur piloté par plug-in appelé Telegraf qui collecte et rapporte les métriques de toutes les sources prises en charge et alimente InfluxDB avec ces données ou vice versa.

Telegraf peut être démarré en tant que service sysvinit ou systemd sous linux, ou l'exécuter en tant que commande de terminal.

À partir de la figure ci-dessous on peut voir que telegraf a des plug-ins d’entrée et de sortie, qui doivent être configurés à l’avance avant de le démarrer. Par exemple, il dispose d’un plug-in d’entrée consommateur apache kafka qui permet de lire les messages d’un sujet kafka et d’un plug-in de sortie qui permet d'écrire toutes les données d’entrée dans influxDB.

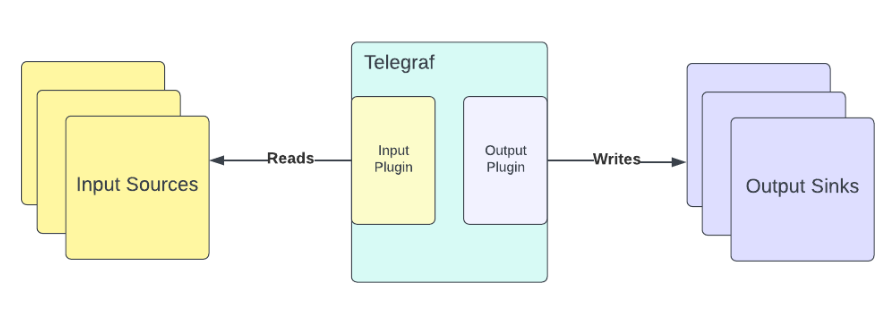


Figure 2 illustration du fonctionnement InfluDB

**Analyse Kafka + influxDB dans notre Cas d'étude**

Si nous prenons un exemple sur les données trafic routier de notre TER qui a des capteurs x, y et z. Dans ce cas on doit collecter des données d’une application de trafic routier, puis envoyer les données à Apache kafka pour qu’elles soient lues par influxDB et affichées sur son interface d’exploitation les données

Architecture global de l’application ressemblera à celui ci :

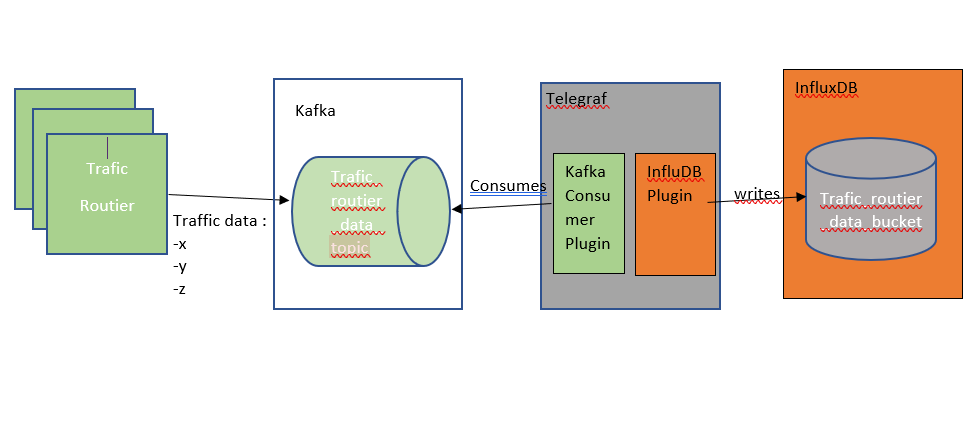


Figure 3 illustration du fonctionnement InfluDB + kafka

Sources utilisées :

<https://mcherif.wordpress.com/2015/08/07/big-data-kafka-concepts-generaux/>

<https://www.splunk.com/fr_fr/data-insider/what-is-stream-processing.html#what-is-stream-processing>

<https://www.journaldunet.fr/web-tech/guide-de-l-entreprise-digitale/1443846-influxdb-une-base-de-donnees-time-series-open-source-sur-optimisee/>

<https://www.ionos.fr/digitalguide/hebergement/aspects-techniques/quest-ce-quinfluxdb/>

<https://www.influxdata.com/blog/getting-started-apache-kafka-influxdb/>

https://www.influxdata.com/blog/getting-started-apache-kafka-influxdb/