Documentation – Analyseur de logs Apache vers Kafka

Ce script utilise **PySpark Structured Streaming** pour lire un flux de logs HTTP depuis Kafka, les parser, calculer des statistiques d'erreurs (globales, par URL et par IP), et envoyer des alertes sur un autre topic Kafka si certains seuils sont dépassés.

Fonctionnalités principales

1. **Connexion à Kafka** : Consomme les logs depuis un topic Kafka (`http-logs`).
2. **Parsing des logs** : Utilise une expression régulière pour extraire les champs IP, timestamp, méthode, URL, et code HTTP.
3. **Traitement par batch** :
    - Calcule le taux d'erreur global.
    - Calcule les taux d'erreur par URL et par IP.
    - Génère des alertes si les seuils configurés sont dépassés.
4. **Envoi des alertes** : Les alertes sont envoyées dans un topic Kafka (`alerts`).

Architecture du flux

```
Kafka (http-logs) --> PySpark Streaming --> Parsing logs --> Calculs d'erreurs -->

Génération d'alertes --> Kafka (alerts)
```

Détail des Fonctions

`create_spark_session()`

Crée et configure une session Spark.

`connect_to_kafka(spark, kafka_brokers, topic)`

Se connecte à un topic Kafka pour lire les messages (logs bruts).

`parse_logs(df)`

Parse chaque log HTTP (au format Apache/Nginx) en extrayant les champs pertinents.

`process_batch(batch_df, batch_id, spark, kafka_brokers, alerts_topic, thresholds)`

Analyse les données par batch :

- Taux d'erreur globale
- Taux d'erreur par URL (si ≥ 5 requêtes)
- Taux d'erreur par IP (si ≥ 3 requêtes)
- Génère et envoie les alertes à Kafka.

`main()`

Point d'entrée :

- Initialise Spark
- Configure Kafka (entrée/sortie)
- Démarre le streaming
- Applique `process_batch` à chaque micro-batch.

Kafka Topics

- **Entrée (logs)** : `http-logs`
- **Sortie (alertes)** : `alerts`

Chemins de checkpoint

- Pour la lecture Kafka : /tmp/checkpoints
- Pour la détection d’alertes : /tmp/alerts

Dépendances

- PySpark
- Kafka (brokers configurés sur kafka:9092)
- Python 3.6+
- Structure des logs au format commun Apache