INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES HAUTS-DE-FRANCE

Compte-rendu
Travaux pratiques

Izaak AUBERT-MECIBAH

Table des matières

I. Etude : Présentation de la sécurité SSL appliquée à KAFKA

Pour cette étude vous pouvez vous informer avec les documentations de ces sites ci-dessous :
https://kafka.apache.org/documentation/#security
https://www.conduktor.io/kafka/kafka-security/
https://jaehyeon.me/blog/2023-06-29-kafka-development-with-docker-part-8/
https://jaehyeon.me/blog/2023-07-06-kafka-development-with-docker-part-9/

1. Quelles sont les principales caractéristiques de la sécurisation via SSL d'un broker Kafka ?

La sécurisation via SSL (Secure Sockets Layer) pour un broker Kafka implique les éléments suivants :

- **Chiffrement des données en transit** : SSL/TLS assure que les données échangées entre les clients et les brokers Kafka sont chiffrées, empêchant ainsi toute interception ou modification par des tiers.
- **Authentification mutuelle** : Les clients et les brokers peuvent être configurés pour s'authentifier mutuellement via des certificats SSL, garantissant que les deux parties de la communication sont bien celles qu'elles prétendent être.
- **Intégrité des données** : L'utilisation de certificats et de clés privées assure que les données transmises n'ont pas été altérées pendant le transit.

2. Quel est l'apport de ce principe sur la sécurité d'accès à Kafka ?

L'utilisation de SSL pour sécuriser l'accès à Kafka améliore beaucoup la sécurité en :

- **Protégeant la confidentialité des données** échangées en chiffrant les communications.
- **Vérifiant l'identité des parties** impliquées dans la communication (clients, brokers) grâce à des certificats, ce qui réduit le risque d'attaques de type "man-in-the-middle".
- **Assurant l'intégrité des messages** en garantissant que les données n'ont pas été modifiées en route.

3. Expliquer la sécurisation Producer vers broker, broker vers consumer, broker vers broker et comment cela se passe dans un mode cluster

Producer vers broker : Les producteurs utilisent des certificats SSL pour établir une connexion sécurisée avec les brokers. Cela inclut la configuration des propriétés SSL telles que `ssl.keystore.location` et `ssl.truststore.location` pour stocker les certificats et les clés.
Broker vers consumer : Les consommateurs établissent des connexions SSL avec les brokers de la même manière, utilisant les propriétés SSL pour configurer les certificats nécessaires.
Broker vers broker : Dans un cluster Kafka, les brokers communiquent entre eux en utilisant SSL pour sécuriser les échanges de données répliquées et autres métadonnées. Cela nécessite la configuration du protocole de sécurité inter-brokers (`security.inter.broker.protocol`).
Mode cluster : Dans un cluster, tous les composants doivent être configurés pour utiliser SSL, y compris les connexions entre les brokers et Zookeeper (s'il est utilisé). Chaque broker doit avoir ses propres certificats et clés configurés pour assurer des communications sécurisées au sein du cluster.

4. Quand est-il de Zookeeper, faut-il également le sécuriser ?

Zookeeper, utilisé pour la coordination dans Kafka, doit également être sécurisé pour empêcher les accès non autorisés. Cela peut être fait en utilisant SSL/TLS pour chiffrer les communications entre Kafka et Zookeeper, et en configurant l'authentification SASL pour sécuriser les connexions des clients Zookeeper.

5. Peut ont sécurisé également un cluster KRAFT avec SSL ? comment ?

Avec KRaft (Kafka Raft), la sécurisation via SSL est également possible et recommandée. Les propriétés SSL similaires à celles utilisées pour les brokers classiques sont appliquées, telles que `ssl.keystore.location` et `ssl.truststore.location`, pour assurer que toutes les communications internes et externes au cluster sont chiffrées et authentifiées.

Pour plus de détails, vous pouvez consulter les documentations officielles sur Kafka Security et les guides pratiques sur Conduktor et les blogs spécialisés sur Jaehyeon.

II. Mise en pratique : Implémentation de la sécurité via SSL à partir d'un projet

Enoncé

Vous utiliserez le projet démo vu en TD que vous modifierez ; celui-ci se trouve dans le dossier TD_KAFKA pour vous inspirer dans ce projet et démarrer avec celui-ci sans trop d'effort de codage.

Nb : Ce projet est basé sur Maven et Java, comme on l'a vu il est assez basique, il met en jeux un producteur et un consommateur dans le même programme qui sont appelé à tour de rôle dans cet ordre, il a pour principe de démontrer le bon fonctionnement de notre broker Kafka et la programmation d'un producteur et d'un consommateur pour Kafka.
Vous pourrez utiliser également le fichier "docker-compose.yml" ou "docker-compose_kafka.yml" de ce même dossier afin de paramétrer votre container docker pour qu'il soit sécurisé via SSL.

Dans le principe il est demandé de modifier le projet (fichiers de configurations respectifs du client Java KAFKA de ce programme et l'un dés fichier docker-compose) qui permettent les actions sécurisées suivantes :

- Forcer le broker a se sécurisé via une couche SSL et utilisation d'un certificat autosigné (voir documentation chapitre 7 d'Apache Kafka).
- Sécuriser les accès par le producteur avec un niveau de sécurité SSL (Certificat), envoyer des éléments via le Producer java Kafka vers ce Broker sécurisé par cette couche SSL.
- Sécuriser les accès du consommateur et la lecture du topic avec un niveau de sécurité SSL (certificats), lire les messages du broker via cet accès sécurisé par la couche SSL.

Vue simplifié du projet

Vous pouvez vous référer à la documentation suivante :

- https://medium.com/jinternals/kafka-ssl-setup-with-self-signed-certificate-part-1-c2679a57e16c
- https://medium.com/jinternals/kafka-ssl-client-authentication-part-2-82c211d64eb6

Conseils

Pour réaliser ce projet, je vous conseille dans un premier temps de faire fonctionner votre broker et votre client en mode non sécurisé, une fois la preuve de concept réalisée, placer le paramétrage de sécurisation de votre broker puis de de votre client (Producer & consumer) en place.

Mise en place

Pour chiffrer les communication broker - broker / broker - producer / broker - consumer, il a été sécurisé un cluster SSL :

- Création d'une autorité de certification avec OPEN SSL.
- Génération d'une paire de clé publique/privée pour chaque broker
- Création d'une requête de signature de certificat pour chaque keystore de broker
- Signer les certificats pour chaque brokers et clients
- Importation du certificat signé dans le keystore de chaque broker
- Génération du trustore

Ensuite il a fallu créer le cluster Kafka. Les brokers ont été configurés pour utiliser SSL pour la communication et pour l'authentification. Concernant les propriété :

- *KAFKA_SSL_** permettent de configurer les fichiers de certificats et keystore pour chaque broker
- *KAFKA_SSL_CLIENT_AUTH* est rendu obligatoire
- *KAFKA_LISTENER_SECURITY_PROTOCOL_MAP & KAFKA_ADVERTISED_LISTENERS* permettent de configurer les protocoles et adresses de communication.