TP Intergiciel et programmation par composants

A réaliser :

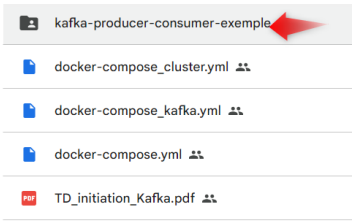
I) [ETUDE] : Présentation de la sécurité SSL appliqué à KAFKA

a) Quelles sont les principales caractéristiques de la sécurisation via SSL d'un broker Kafka ?
b) Quel est l'apport de ce principe sur la sécurité d'accès à Kafka ?
c) Expliquer la sécurisation Producer vers broker, broker vers consumer, broker vers broker et comment cela se passe dans un mode cluster
d) Quand est-il de Zookeeper, faut-il également le sécuriser ?
e) Peut ont sécurisé également un cluster KRAFT avec SSL ? comment ?

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/

II) [MISE EN PRATIQUE] : Implémentation de la sécurité via SSL à partir d'un projet

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.



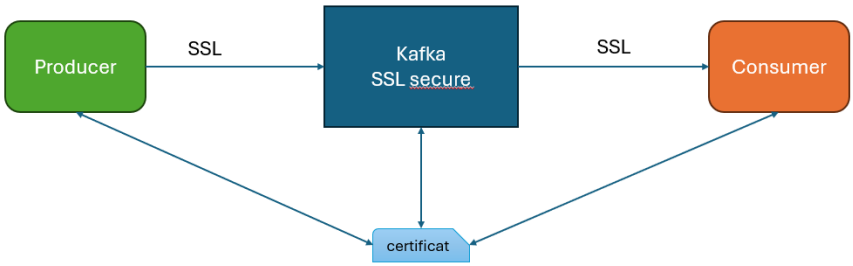
Nb : Ce projet est basé sur Maven et Java, comme on l'a vu il est assez basique, il met en jeux un producteur et un consommateur dans le même programme qui sont appelé à tour de rôle dans cet ordre, il a pour principe de démontrer le bon fonctionnement de notre broker Kafka et la programmation d'un producteur 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

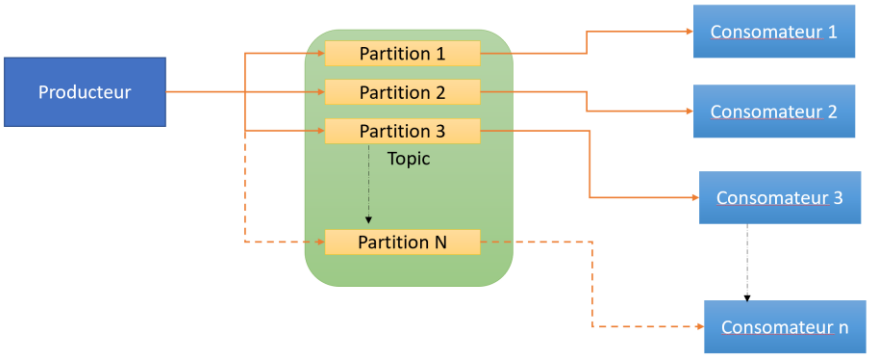
Pour réaliser ce projet, je vous conseille dans un premier temps de faire fonctionner votre broker et votre client en mode non sécurisé, une fois la preuve de concept réalisée, placer le paramétrage de sécurisation de votre broker puis de de votre client (Producer & consumer) en place.

III) [OPTIONS] développement d'un consommateur supplémentaire client et utilisation des partitions

En option, il vous est demandé de développer un consommateur supplémentaire qui pourra consommer les messages de la partition X (ou x est un numéro de partition supérieur à 1) que l'on pourra passer en paramètre à notre application, il sera assez générique pour être instancier autant de fois que nécessaire.

On transformera le producteur pour qu'il puisse alternativement envoyer des messages sur les partitions de votre Broker en utilisant une ventilation par clé ou en écriture directe sur les partitions, on fera en sorte que le producteur écrive équitablement sur l'ensemble des partitions de votre topic.

Vous ajusterez également au niveau votre broker la création du topic pour prendre en charge X partitions. Vous avez le choix du X (tant que > à 1)



A remettre en fin de cycle de TP

Un document comportant les études et synthèses sur les questions I et II ci-dessus.

Un document qui explique comment vous avez fait pour sécuriser votre broker avec un certificat et la couche SSL.

Comment avez-vous modifié votre programme démo afin que le producteur et le consommateur puissent se connecter sur le broker en utilisant la couche d'encryptage SSL et votre certificat.

Vous remettrez vos réponses, scripts, codes sources, fichiers « properties » modifiés, documents via un git (GitHub) au plus tard à la **date du 29 mai 2024**, vous pouvez lancer une invitation a « tondeur-h » sur github pour me partager votre dossier git

Nb : Ce tp fait office de note pour ce module.

Vous enverrez le lien git aux adresses suivantes :

- tondeur.herve@yahoo.fr
- herve.tondeur@uphf.fr

  Aux deux adresses obligatoirement, n'oubliez pas de donner les droits d'accès sur votre repository (invitation).

Nb : Travail en équipe autorisé (3 max).