TP Intergiciels - Étude/Synthèse

MOSBAH Karim

I. Présentation des principes de SASL

1) Qu'est-ce que c'est ?

SASL est l'acronyme de "Simple Authentification and Security Layer". C'est un framework d'authentification et d'autorisation standardisé par l'IETF.

2) Définition : définir SASL dans un cadre générique.

SASL sert d'interface entre un protocole d'application comme SMTP ou IMAP et un mécanisme d'authentification comme CRAM-MD5. Grâce à SASL, un service peut assurer l'authentification via une variété de protocoles de sécurité supportés par SASL.

3) Principes de bases ?

SASL effectue une négociation à base de challenges. Ca se présente sous cette forme :

```
250-mail.example.com Hello pc.example.org [192.168.1.42], pleased to meet you
250-AUTH DIGEST-MD5 CRAM-MD5 LOGIN PLAIN
250 HELP
AUTH CRAM-MD5
334 PDk5MDgwNDEzMDUwNTUyMTE1NDQ5LjBAbG9jYWxob3N0Pg==
amFzIDBkZDRkODZkMDVjNjI4ODRkYzc3OTcwODE4ZGI5MGY3
235 2.0.0 OK Authenticated
```

c.f. https://www.gnu.org/software/gsasl/manual/html_node/SASL-Overview.html

D'abord, le client choisit un protocole dans la liste envoyée par le serveur (ici CRAM-MD5). Ensuite, un challenge permet de vérifier l'identité du client et son accès au protocole choisi.

II. Présentation de SASL appliqué à KAFKA

1) SASL/GSSAPI (Kerberos)

GSSAPI est une autre framework comme SASL qui supporte des mécanismes d'authentification, dont Kerberos. Pendant la négociation SASL, le client choisit GSSAPI parmi les mécanismes SASL supportés. Puis il va généralement choisir Kerberos parmi les mécanismes d'authentification supportés par GSSAPI. Ensuite se déroule l'authentification selon le protocole Kerberos, qui est un système à base de tickets.
c.f. https://serverfault.com/questions/139896/what-is-sasl-gssapi

2) SASL/PLAIN

Le mécanisme PLAIN consiste à transmettre les identifiants et mots de passe en clair pour authentifier le client. Cette méthode ne doit être utilisée que si le lien de communication est déjà sécurisé par un autre moyen. Autrement, les identifiants seraient facilement interceptés par eavesdropping. Les identifiants sont stockés dans le broker. L'avantage de cette méthode est qu'elle est très facile à mettre en œuvre.
Avec SASL/PLAIN, les consumers/producers non-identifiés n'ont plus accès aux brokers.

3) SASL/SCRAM-SHA-256 et SASL/SCRAM-SHA-512

C'est un mécanisme qui fonctionne par mot de passe, comme le SASL/PLAIN. Il règle une grande partie des problèmes de sécurité de celui-ci en ajoutant un challenge à la procédure d'authentification. La différence entre scram-sha-256 et scram-sha-512 est dans la taille du hash utilisé pour le mot de passe. Les identifiants sont stockés dans le zookeeper.

4) SASL/OAUTHBEARER

OAUTHBEARER est un mécanisme qui authentifie les clients via des tokens. Ces tokens sont distribués par un serveur OAUTH 2.0 et représentent le droit d'un client d' accéder à des ressources kafka. Cette méthode offre des avantages certains pour la sécurité du service mais elle nécessite d'avoir un disposition un serveur d'authentification oAuth 2.0.

Qu'en est-il de Zookeeper ? Faut-il également le sécuriser ?

Oui, il faut également sécuriser zookeeper. En premier lieu parce que c'est un élément indispensable de l'architecture kafka, ensuite parce qu'il gère les autorisations des différents producers/consumers via les ACLs (Access Control Lists).

Il s'agit essentiellement de sécuriser les connections au zookeeper via ssl ou tls.

Peut-on se passer de Zookeper ? Si oui qui est le grand remplaçant ? et comment fonctionne SASL sur ce remplaçant ?

Oui, on peut se passer de zookeeper. Il est même prévu que zookeeper disparaisse complètement de kafka d'ici quelques années.
Son grand remplaçant est KRaft. KRaft détermine parmi les brokers un "KRaft leader" et des "KRaft controllers" qui font le travail auparavant effectué par zookeeper. Donc si on a configuré SASL pour sécuriser les échanges de données autour des brokers, alors c'est désormais suffisant avec KRaft.

Sitographie

https://medium.com/@stephane.maarek/introduction-to-apache-kafka-security-c8951d410adf
https://kafka.apache.org/documentation/#security
https://datatracker.ietf.org/doc/html/rfc5802
https://www.openlogic.com/blog/apache-kafka-best-practices-security
https://www.lemagit.fr/actualites/366537484/Apache-Kafka-ZooKeeper-fait-de-la-resistance-mais-Confluent-a-un-plan
https://developer.confluent.io/learn/kraft/
https://docs.confluent.io/platform/current/kafka-metadata/config-kraft.html
https://www.conduktor.io/blog/kraft-quorum-run-kafka-without-zookeeper/