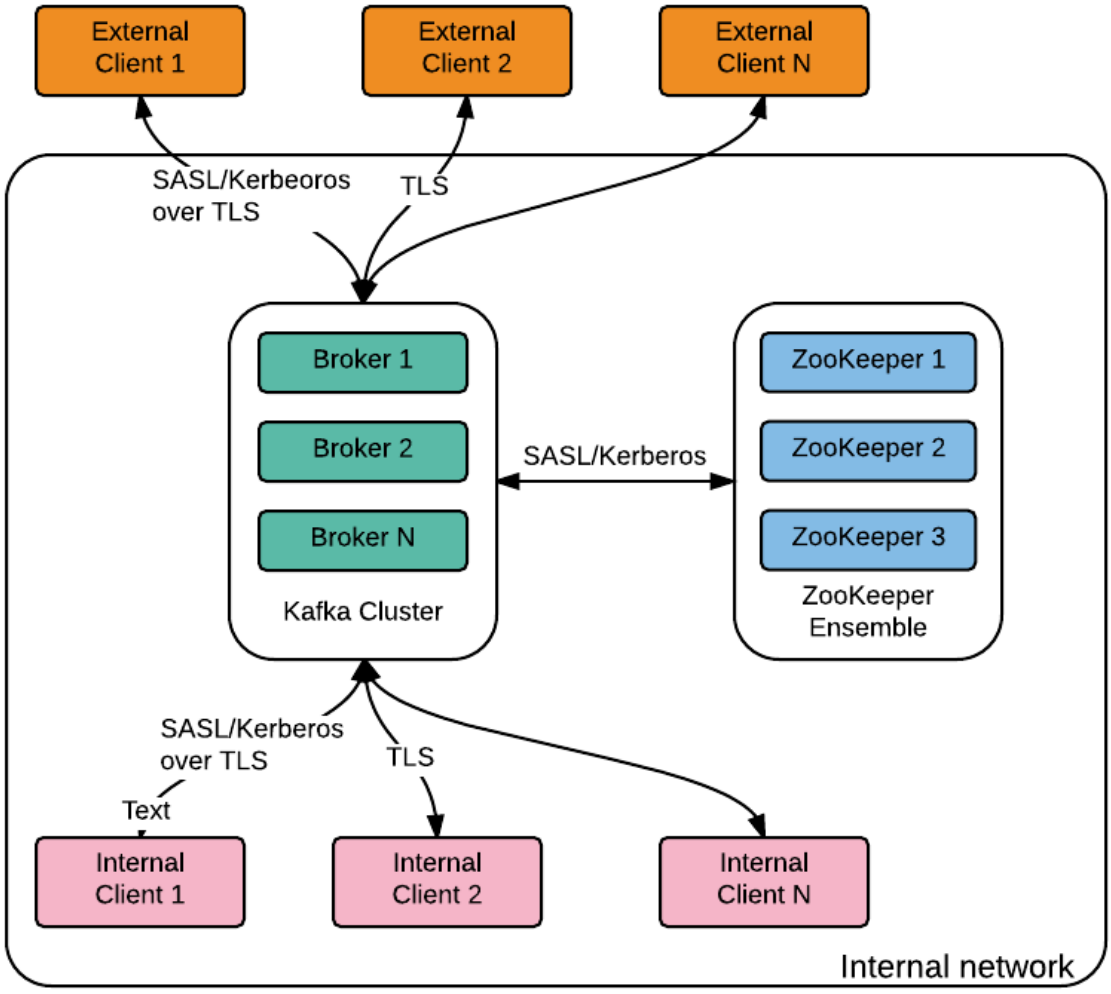
Rendu TP partie I et II: Intergiciel et programmation par composants

Réalisé par: MEFTAHI Naoufal - SEKKOUMI Samir - EL MOUDEN El Mehdi

Présentation des principes de SASL



Qu'est-ce que c'est ?

La couche SASL (Simple Authentication and Security Layer) est une structure fournissant des services d'authentification et de sécurité facultatifs aux protocoles réseau. Une application appelle la bibliothèque SASL, /usr/lib/libsasl.so, qui fournit une couche de collage entre l'application et les divers mécanismes SASL. Les mécanismes sont utilisés dans le processus d'authentification et lors de la fourniture de services de sécurité facultatifs. La version de SASL est dérivée de la couche Cyrus SASL avec quelques changements.

Définition du SASL dans un cadre générique

Dans un cadre générique, SASL est un framework permettant de fournir des services d'authentification et de sécurité des données dans les protocoles orientés connexion via des mécanismes remplaçables pour les protocoles d'application, tels que SMTP ou IMAP. Il fournit une interface structurée entre les protocoles et les mécanismes. SASL prend en charge divers mécanismes d‘authentification, tels que les noms d‘utilisateur/mots de passe, Kerberos, certificats X.509, etc., offrant ainsi une flexibilité pour répondre aux exigences spécifiques de sécurité d‘une application.

Les principes de base ?

Modularité : SASL est conçu pour être modulaire, ce qui signifie qu'il peut prendre en charge différents mécanismes d'authentification en fonction des besoins.

Extensibilité : Il est possible d'ajouter de nouveaux mécanismes d'authentification sans avoir à modifier les protocoles existants.

Sécurité : SASL fournit un moyen sûr de négocier des mécanismes d'authentification et de protéger les échanges de données lors de la communication entre les clients et les serveurs.

Séparation des couches : SASL permet de séparer les mécanismes d'authentification et de sécurité des protocoles de communication sous-jacents. Cela permet aux protocoles de rester indépendants des détails spécifiques liés à l'authentification.

Négociation de mécanismes : SASL offre un mécanisme de négociation entre le client et le serveur pour déterminer le mécanisme d'authentification à utiliser. Les deux parties peuvent proposer une liste de mécanismes pris en charge, et le mécanisme le plus sûr commun à toutes les parties est sélectionné.

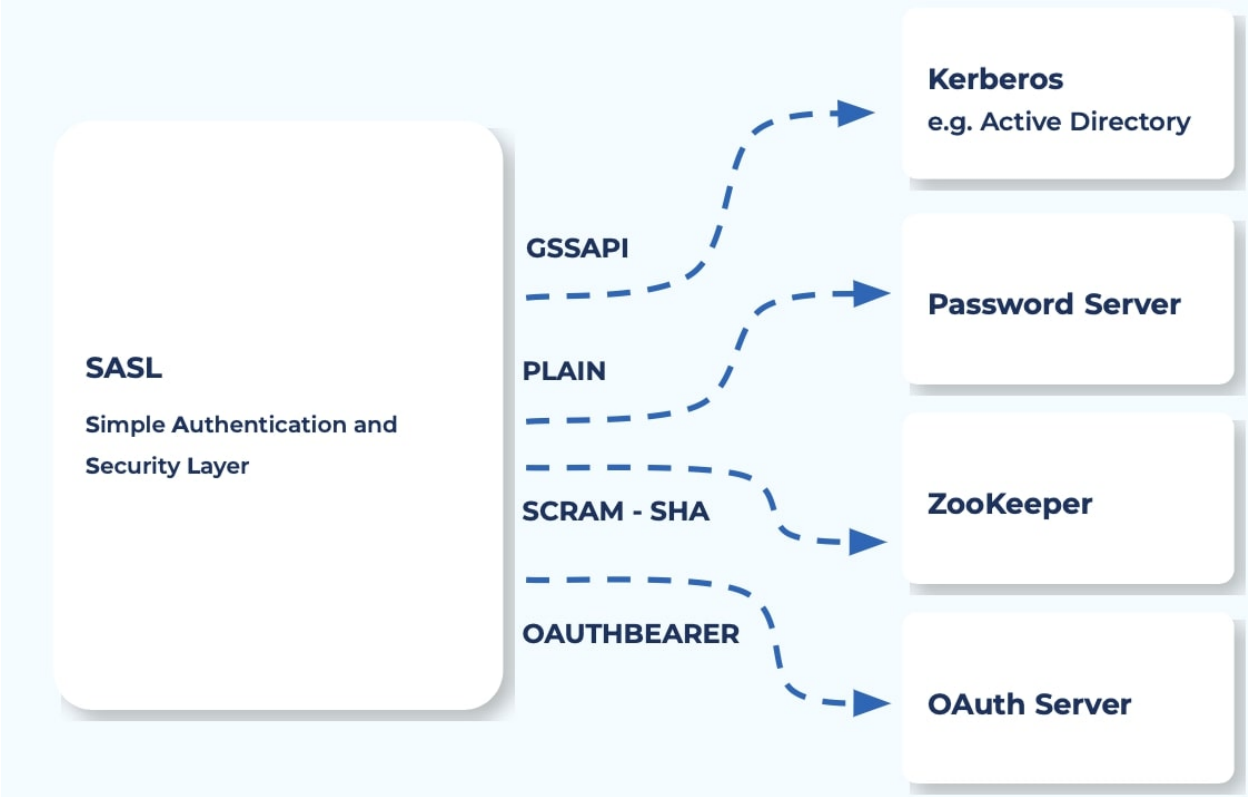
Confidentialité et intégrité des données : SASL prend en charge la confidentialité et l'intégrité des données échangées entre le client et le serveur. Cela peut être réalisé en utilisant des mécanismes de chiffrement et de signature numérique appropriés lors de la communication.

Gestion des identités : SASL permet la gestion des identités des clients et des serveurs. Il offre des mécanismes d'authentification pour vérifier l'identité des utilisateurs et des services, tels que les noms d'utilisateur et les certificats.

Adaptabilité : SASL est conçu pour être adaptable à différents environnements et mécanismes d'authentification. Il peut être utilisé avec divers protocoles de communication tels que LDAP, SMTP, IMAP, etc.

Mécanismes personnalisés : SASL permet également la création de mécanismes d'authentification personnalisés. Cela offre la flexibilité de mettre en œuvre des mécanismes spécifiques aux besoins d'une application ou d'un système.

Présentation de SASL appliqué à KAFKA



SASL : Mécanismes d'authentification

1. SASL / GSSAPI (Kerberos)
2. SASL / PLAIN
3. SASL / SCRAM-SHA-256 et SASL / SCRAM-SHA-512
4. SASL / OAUTHBEARER

Quelles sont les principales caractéristiques de ces quatre principes ci-dessus et quel est l'apport de chacun de ces principes sur la sécurité d'accès à Kafka ?

SASL/GSSAPI (Kerberos) :
Caractéristiques principales : Utilise le protocole Kerberos pour l'authentification et fournit une sécurité basée sur des tickets .
Apport sur la sécurité d'accès à Kafka : Renforce la sécurité en utilisant des tickets Kerberos pour l'authentification des clients Kafka.
Permet une intégration transparente avec les environnements Active Directory et les infrastructures de gestion des identités basées sur kerberos.

SASL/PLAIN :
Caractéristiques principales : Utilise l'authentification par nom d'utilisateur et mot de passe en texte brut.
Apport sur la sécurité d'accès à Kafka : Fournit une méthode simple d'authentification, mais moins sécurisée que d'autres mécanismes basés sur des clés ou des certificats.
Généralement utilisé dans des scénarios où la communication est déjà sécurisée par d'autres moyens(par exemple,sur une connexion SSL/TLS).

SASL/SCRAM-SHA-256 et SASL/SCRAM-SHA-512 :
Caractéristiques principales : Utilisent l'algorithme de hachage HMAC avec des fonctions de hachage SHA-256 et SHA-512 pour l'authentification basée sur SCRAM (Salted Challenge Response Authentification Mechanism)..
Apport sur la sécurité d'accès à Kafka : Offre une sécurité renforcée par rapport à SASL/PLAIN en utilisant des fonctions de hachage sécurisées (SHA-256 ou SHA-512) pour l'authentification. Offre une protection robuste contre les attaques par force brute et les attaques par dictionnaire et assure la confidentialité des données lors des échanges.
SASL/OAUTHBEARER :
Caractéristiques principales : Utilise le protocole OAuth pour l'authentification basée sur les jetons d'accès.
Apport sur la sécurité d'accès à Kafka : Permet l'authentification sécurisée en utilisant des jetons d'accès OAuth, ce qui est particulièrement utile dans les environnements avec des services tiers et des autorisations granulaires.
Principalement utilisé dans les applications qui interagissent avec des services Web et des API.
Fournit un mécanisme d'autorisation granulaire pour accéder aux ressources.

Sécurisation de zookeeper :

Quand il s'agit de ZooKeeper dans le contexte de Kafka, il est fortement recommandé de sécuriser également ZooKeeper. ZooKeeper joue un rôle crucial dans la coordination et la

gestion des métadonnées de Kafka. Une sécurité inadéquate de ZooKeeper peut entraîner des vulnérabilités potentielles et compromettre la sécurité globale du cluster Kafka.

Pour sécuriser ZooKeeper, il faut impérativement prendre les mesures suivantes :

Authentification SASL : On configure ZooKeeper pour utiliser l'authentification SASL, ce qui permet une authentification forte des clients se connectant à ZooKeeper.

Chiffrement SSL/TLS : On active le chiffrement SSL/TLS pour chiffrer les communications entre les clients et ZooKeeper, empêchant ainsi l'interception non autorisée des données.

Autorisations et contrôle d'accès : On essaie de configurer des règles d'autorisation appropriées pour contrôler l'accès aux ressources de ZooKeeper. Limitez les privilèges aux seuls utilisateurs et services nécessaires.

Sécurité du système d'exploitation : On s‘assure que le système d'exploitation sur lequel ZooKeeper s'exécute est sécurisé en appliquant les bonnes pratiques de sécurité, telles que les correctifs réguliers, la configuration appropriée des pare-feux, etc.

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

Zookeeper agit comme un système de gestion de métadonnées externe pour kafka. Cela crée de multiples problèmes tels que la duplication des données, la complexité accrue du système et conduit également à l'utilisation de processus Java supplémentaires.

Afin d'exécuter kafka sans zookeeper, il peut être exécuté en utilisant le mode de métadonnées Kafka Raft (KRaft) . Le KRaft mode permet à Kafka de fonctionner sans dépendre de ZooKeeper pour la gestion des métadonnées. Au lieu de cela, Kafka utilise un protocole de consensus basé sur Raft pour la gestion des métadonnées, ce qui offre une plus grande simplicité et une meilleure résilience.

Ceci est disponible en mode expérimental dans kafka 2.8.0.

À l'heure actuelle, l'utilisation de KRaft est expérimentale et ne devrait pas être utilisée en production.