Guide de Migration : De l'Authentification par Login/Mot de Passe au Client ID/Secret

Ce guide vous accompagnera étape par étape dans la transition de l'ancienne méthode d'authentification basée sur le **login et le mot de passe** vers la nouvelle méthode utilisant **Client ID et Client Secret**. Voici comment procéder pour migrer en toute sécurité.

Ancienne Méthode : Authentification par Login et Mot de Passe

Requête HTTP :

- **URL :** `https://api.infolocale.fr/v2/auth/signin`
- **Méthode :** POST

Paramètres de formulaire :

- **login (texte)** - Obligatoire. E-mail utilisé lors de l'inscription.
- **password (texte)** - Obligatoire. Mot de passe utilisé lors de l'inscription.

Nouvelle Méthode : Authentification par Client ID et Client Secret

Requête HTTP :

- **URL :** `https://auth.ouest-france.fr/auth/realms/sipa/protocol/openid-connect/token`
- **Méthode :** POST

Paramètres de formulaire :

- **grant_type (texte)** - Obligatoire. Doit être défini à `client_credentials` pour indiquer une authentification par les credentials du client.
- **client_id (texte)** - Obligatoire. Identifiant unique de l'application cliente, défini lors de l'enregistrement.
- **client_secret (texte)** - Obligatoire. Secret partagé utilisé pour authentifier l'application cliente auprès du serveur d'autorisation.

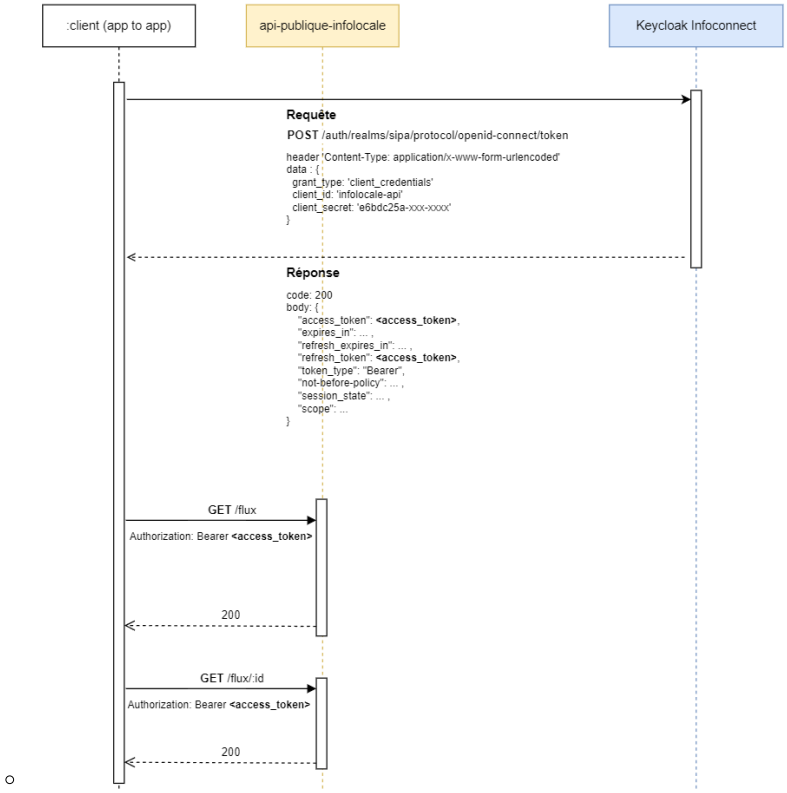
Headers de la Requête :

- **Content-Type:** `application/x-www-form-urlencoded`
  - Il est essentiel d'inclure ce header dans la requête pour indiquer que les données du formulaire sont encodées en URL (comme lors de la soumission d'un formulaire web).

Procédure de Migration

Bien que la méthode d'obtention du jeton d'accès change, l'utilisation de ce jeton pour accéder aux données de l'API reste inchangée. Il est essentiel de continuer à inclure le jeton d'accès dans toutes les requêtes envoyées à l'API après la migration.

1. **Préparation :**
   - **Obtention des Credentials :** Le `client_id` et le `client_secret` nécessaires pour la nouvelle méthode d'authentification seront fournis par Infolocale. Assurez-vous de les demander et de les recevoir avant de commencer la migration.
   - Assurez-vous que ces credentials sont stockés de manière sécurisée et ne sont accessibles qu'aux parties autorisées de votre système.
2. **Modification des Requêtes :**
   - Mettez à jour les requêtes dans votre code source pour remplacer l'URL de l'ancienne méthode d'authentification par la nouvelle URL.
   - Remplacez les paramètres `login` et `password` par `grant_type`, `client_id`, et `client_secret`.
   - Voici un diagramme de séquence qui permettra une meilleure vue d'ensemble :



3. ## Renouvellement du « jeton d'accès » :
   - **NOTA** : Lorsque vous utilisez notre système d'authentification, il n'est pas nécessaire de gérer manuellement la fin de session via un processus de déconnexion. Nos jetons d'accès sont conçus pour expirer automatiquement après une durée déterminée pour assurer la sécurité de vos données.
   - **Automatisation du renouvellement** : Afin de maintenir une expérience utilisateur fluide et sécurisée, nous recommandons d'automatiser le renouvellement de votre jeton d'accès. Plutôt que de surveiller l'expiration du jeton actuel, il est conseillé de demander un nouveau jeton à chaque fois que vous avez besoin d'accéder à des ressources protégées. Cette pratique garantit que vous disposez toujours d'un jeton valide sans nécessiter de calculs complexes ou de mécanismes de gestion de session.
   - **Pas de déconnexion nécessaire** : Dans notre système, le processus de déconnexion au sens traditionnel n'est pas applicable. Comme nos jetons d'accès expirent de manière autonome, il n'y a pas de session utilisateur à fermer explicitement. Cela simplifie la gestion de l'authentification et réduit le risque d'erreur humaine dans le processus de sécurisation des accès.
   - **Procédure de renouvellement** : Lorsque vous avez besoin d'un nouvel accès, simplement soumettez une nouvelle demande d'authentification selon la procédure standard décrite ci-dessus. Vous recevrez un nouveau jeton d'accès à utiliser pour vos requêtes subséquentes.
4. ## Exemple d'Implémentation en PHP :
   - Voici un exemple de code PHP utilisant la bibliothèque GuzzleHttp pour récupérer un jeton d'accès en utilisant la méthode Client ID et Client Secret. Cet exemple illustre comment soumettre une requête POST avec le header approprié et les paramètres requis.

```php
<?php

require 'vendor/autoload.php';

use GuzzleHttp\Client;

// Création d'un client Guzzle
$client = new Client();

// Tentative d'obtention du token d'accès avec le header Content-Type spécifié
try {
    $response = $client->request('POST', 'https://auth.ouest-france.fr/auth/realms/sipa/protocol
/openid-connect/token', [
        'headers' => [
            'Content-Type' => 'application/x-www-form-urlencoded',
        ],
        'form_params' => [
            'grant_type' => 'client_credentials',
            'client_id' => 'VOTRE_CLIENT_ID',
            'client_secret' => 'VOTRE_CLIENT_SECRET',
        ]
    ]);

    $data = json_decode($response->getBody(), true);
    echo "Access Token: " . $data['access_token'];
} catch (\GuzzleHttp\Exception\GuzzleException $e) {
    echo "Erreur lors de la requête : " . $e->getMessage();
}

?>
```