Cookbook Oauth Taillefer

Taillefer Jordan

Octobre 2016

1 Résumé

De nos jours, le nombre de sites proposant des API d'authentification ne cesse de grandir. Facebook, Twitter, Google ne sont que quelques-uns parmi des milliers de Provider. Pour un programmeur, il est impossible de pouvoir gérer rapidement et facilement l'accès à chacun d'entre eux.

Blaine Cook et Chris Messina, créateur d'Oauth, ont donc réalisé un service gratuit permettant à un site web de pouvoir obtenir via une API Key, une autorisation d'accès à l'API d'un autre site web.

Ce cookbook aura donc pour objectif de présenter comment cette technologie arrive à connecter un site web à plusieurs API simplement en passant par une unique authentification.

2 La recette de Oauth

2.1 Les caractéristiques de Oauth

Pour commencer, OAuth a été rendu disponible dans plusieurs langages.

Les langages disponibles pour Oauth sont :

- Javascript
- Phonegap/Cordova
- iOS
- Android
- PHP
- Node.js
- Java
- Go

Dans notre cas, nous utiliserons le Javascript.

Pour l'installation, l'outil propose d'utiliser simplement le gestionnaire de dépendance bower.

```
bower install oauth-js
<script src="bower_components/oauth-js/dist/oauth.js"></script>
```

Figure 1: Installation Oauth

2.1.1 Plusieurs goûts possibles

Contrairement à la majorité des services d'authentification, Oauth n'est pas un protocole, mais plutôt une autorité qui permet d'authentifier un utilisateur à un autre service, on délègue en quelque sorte le travail d'authentification à Oauth.

Le service propose la possibilité d'être connecté à plus de 120 providers, allant des plus connus comme Facebook ou Twitter mais aussi à certains bien moins influents tels que toutes les branches de la firme Google. Le service offre notamment un accès aux blogs skyrock, aux musiques de Spotify ou encore aux widgets de Wordpress.

2.1.2 Étape : Authentification

Comme nous avons vu précédemment, on va donc déléguer nos authentifications au service. Pour cela, nous allons devoir faire une simple et unique identification auprès de Oauth, puis c'est lui qui va gérer derrière les liens.

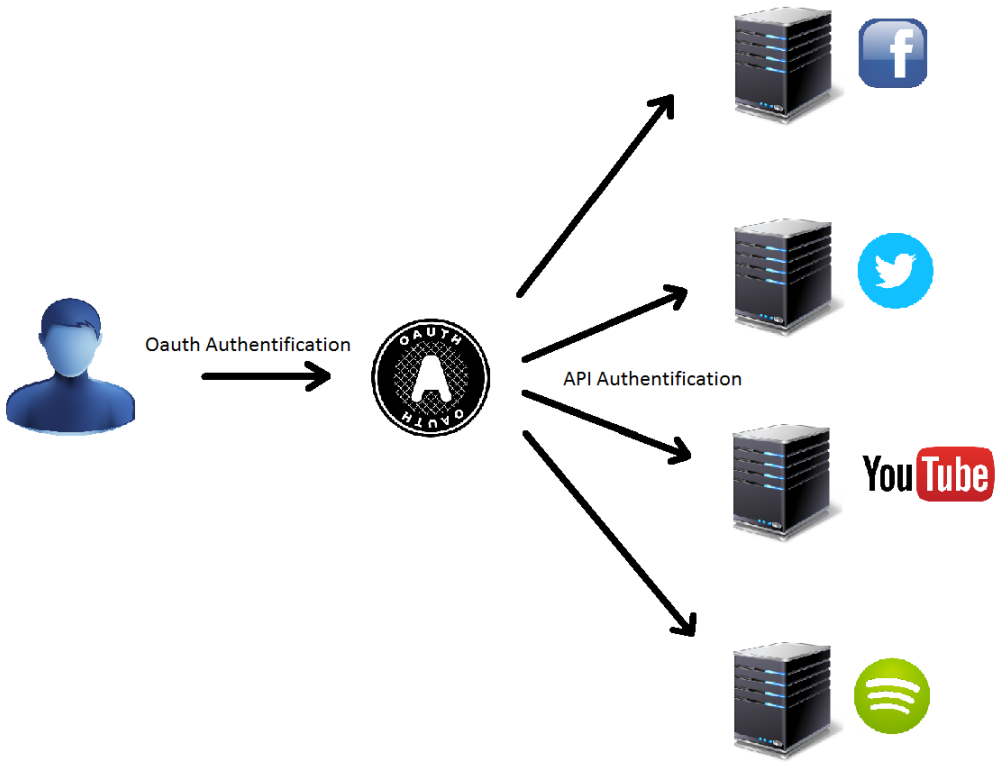

Figure 2: Schema Oauth

2.1.3 Étape : Authentification du client à Oauth

Pour commencer l'étape d'authentification du client vers Oauth, il faut tout d'abord s'enregistrer sur le site. Une fois cela fait, une interface de gestion de son compte est proposée afin de gérer les accès aux providers.

Pour initialiser le SDK, nous avons besoin de récupérer la clé publique de notre compte.

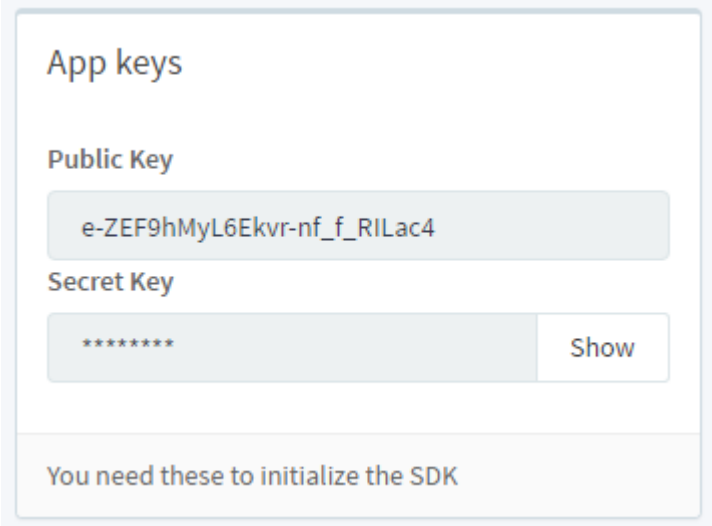

Figure 3: Public key Oauth

Puis de l'insérer dans le code fourni

```
// On initialise oauth avec la clé public
OAuth.initialize( 'e-ZEF9hMyL6Ekvr-nf_f_RILac4' )
```

Figure 4: Initialisation Oauth

Une fois ce code ajouté, l'application client sera authentifiée auprès de Oauth. Il reste plus qu'a authentifier les providers auprès de Oauth avant de pouvoir les utiliser

2.1.4 Étape : Authentification de Oauth aux Providers

Cette étape est simple, mais reste la plus longue, il va falloir authentifier chaque provider souhaité.

Pour cela, comme vu précédemment, un site développeur est généralement proposé, pour l'exemple, nous allons reprendre le cas de Facebook.

Nous allons donc demander à Oauth d'authentifier un nouveau provider, dans notre cas Facebook.

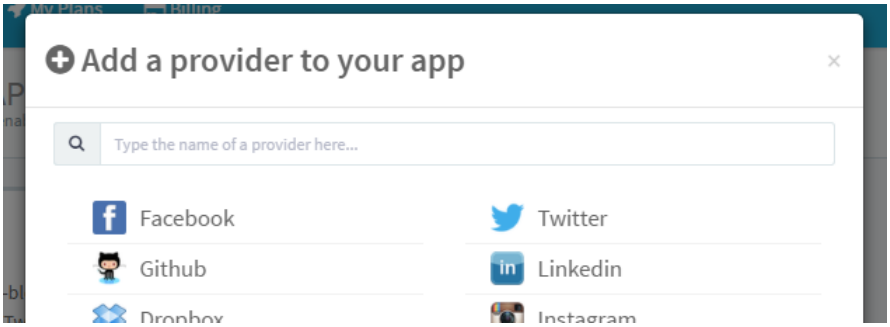

Figure 5: Selection d'un nouveau provider

Il nous est demandé des informations concernant le compte développeur Facebook. Dans notre cas, nous avons donc

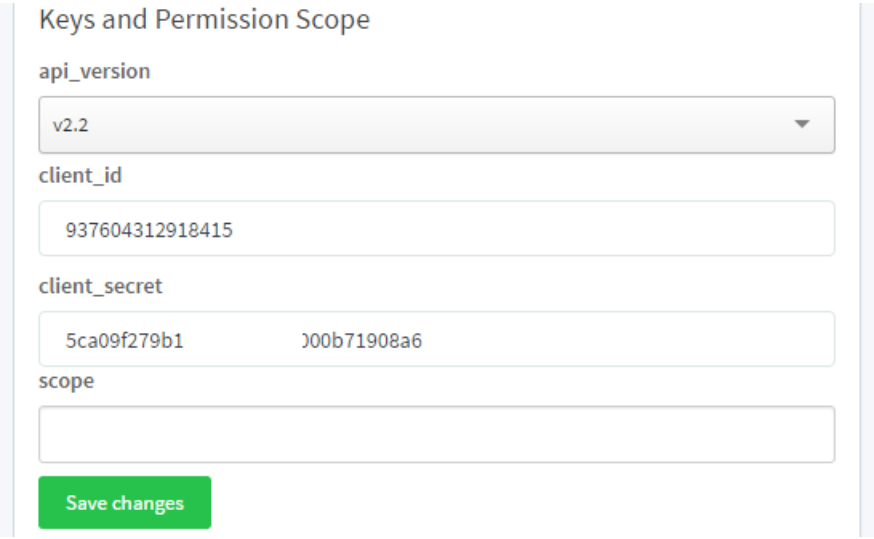

Figure 6: Information Facebook

Une fois que tout est rempli, il nous reste plus qu'à valider et à tester directement sur Oauth si toutes nos informations sont correctes.

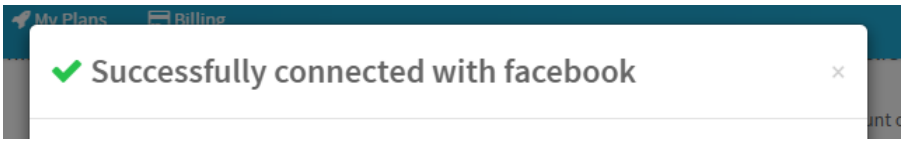

Figure 7: Connexion réussie Facebook

2.1.5 2 façons de servir OAuth

Il existe 2 possibilités d'exploitation des services Oauth.

La première est d'ouvrir une fenêtre pop-up, l'utilisateur va alors charger que la partie authentification sans avoir à changer de page. Une fois validé, le pop-up va renvoyer un code de succès avec les données demandées.

```
//Exemple de connexion avec Facebook via une popup
OAuth.popup('facebook').done( result => console.log(result)
  //Fonction executé en cas de réussite de l'authenfication
).fail( error => console.log( error )
  //Fonction executé en cas d'echec d'authenfication
)
```

Figure 8: Connexion Popup Fb

La seconde possibilité est de rediriger l'utilisateur directement sur le site afin qu'il s'identifie, le code nécessite de préciser la page de retour afin de revenir sur le site après un succès ou un échec.

Une fois que l'opération est terminée, avec succès ou échec, le provider redirige l'utilisateur vers le site d'origine fourni précédemment.

```
//Exemple de connexion avec Facebook via une redirection
//Oauth va rediriger l'utilisateur sur la page d'authenfication de facebook
//http://localhost/callback sera l'adresse de retour une fois l'identification finie
OAuth.redirect( 'facebook', 'http://localhost/callback.html' )

//in your callback page (can be the same page)
OAuth.callback('facebook').done( result => console.log( result )
  //Fonction executée en cas de réussite de l'authenfication
).fail( error => console.log( error )
  //Fonction executée en cas d'echec d'authenfication
)
```

Figure 9: Connexion Redirect Fb

Étant donné que l'utilisateur doit charger entièrement 2 pages internet, comparé à une simple pop-up, la seconde possibilité serait plutôt faite pour un site léger ayant peu de contenu à charger et nécessitant une connexion internet moins importante que la première solution.

2.2 Exemple de présentation

L'exemple suivant montre le résultat de la recette. Afin de se connecter, l'utilisateur possède un bouton explicite qui va lui ouvrir un pop up lui demandant ses informations de compte ainsi qu'une confirmation des informations qui vont être envoyés à notre application.

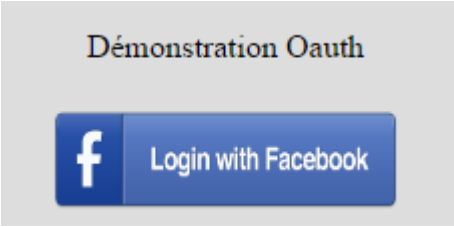

Figure 10: Connexion en attente Facebook

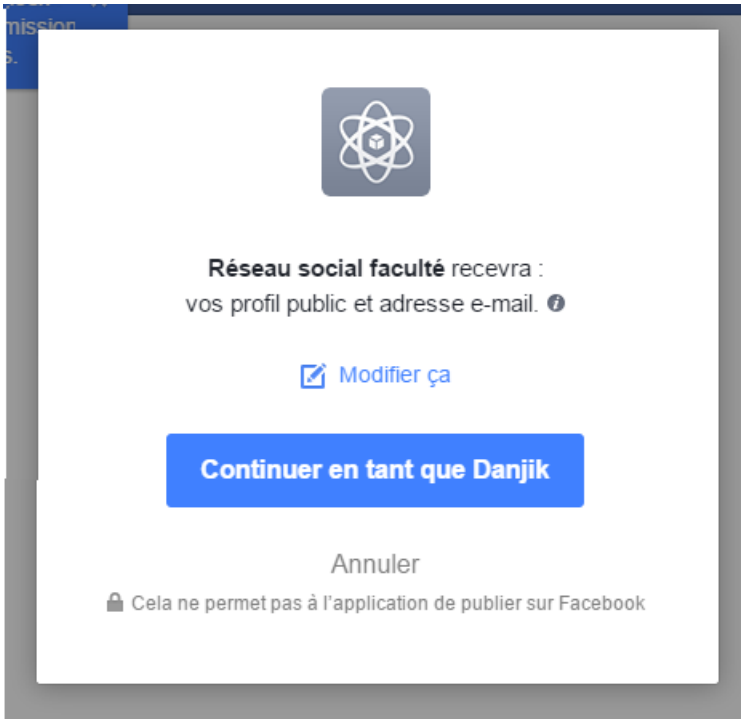

Figure 11: Connexion demande de droit

Une fois connectée, l'application a donc accès aux données envoyées par l'api.

Figure 12: Connexion réussie Facebook

3 Conclusion

Nous avons donc pu voir l'utilisation globale de Oauth, cet outil est vraiment puissant pour pouvoir se connecter rapidement un ensemble de providers.

Nous avons accès à une gestion des accès regroupés en un seul point. De plus le site offre un tableau de bord indiquant le nombre ou encore la fréquence d'utilisation de chaque provider.

Cependant, il reste certains inconvénients.

Nous avons vu que nous regroupons l'accès à tous les providers en un seul point, une personne malveillante n'a donc qu'à s'introduire dans le compte Oauth pour pouvoir avoir accès à tous les providers enregistrés.

De plus, le client est dépendant du fonctionnement du site pour chacun des providers, si celui-ci tombe, le client perd tous les moyens de sa connexion.

Enfin, l'outil permet de gérer le scope d'utilisation, mais il faudrait se pencher sur le fait de savoir si on a accès à tout ce que propose chaque API de base. [3]

References

[1] Facebook. Site développeur facebook. `https://developers.facebook.com/`.

[2] Oauth. Documentation du projet oauth. `http://docs.oauth.io/`.

[3] Oauth. Github du projet oauth. `https://github.com/oauth-io/oauth-js`.