Manuel utilisateur de la solution Archibook

Aymeric BOCQUET
Adrien FERRER
Clément LANDOUAR
Benoît PERROT

I) Architecture globale

Archibook est un réseau social où l'on se connecte pour accéder à la liste des élèves de son école. Le schéma suivant présente l'architecture générale d'Archibook.

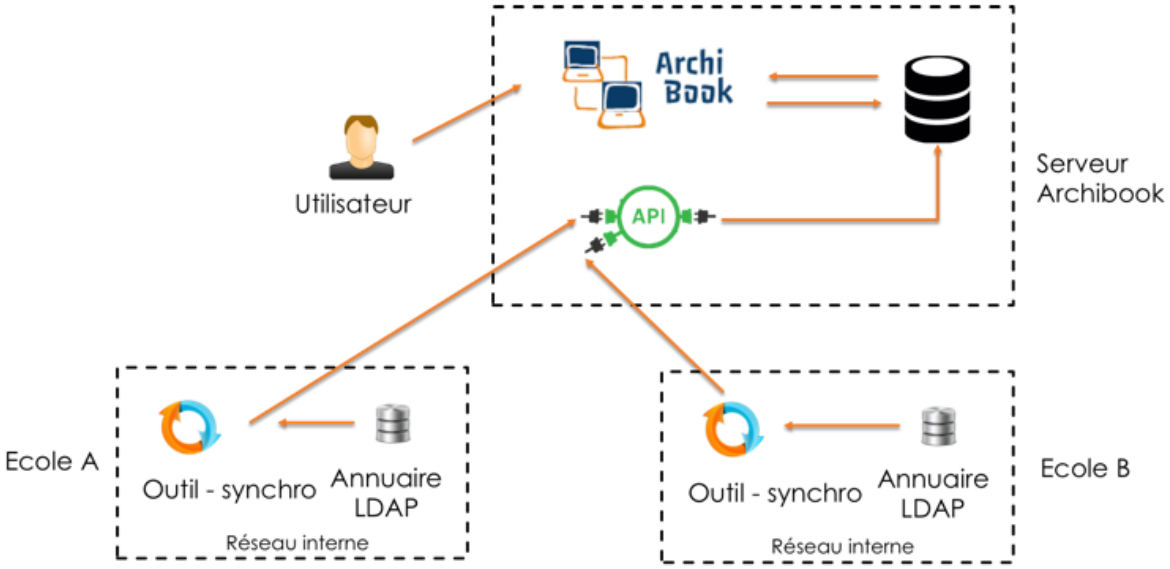

Figure 1 : Schéma de l'architecture d'Archibook

Archibook repose sur une architecture « 3-tiers » ce qui permet une optimisation de son fonctionnement.

Le logiciel à installer sur votre réseau est à configurer (cf partie II). Ensuite, le script se connecte à votre LDAP. Il transfère la donnée à notre API en HTTPS.

L'API REST, récupère toutes les données de la requête sous forme JSON. Elle les transfère ensuite dans la base de données du réseau social. Cette API permet d'éviter qu'un tiers malveillant se connecte directement sur la BDD.

Les mots de passes sont hachés dans la BDD pour assurer un niveau de sécurité maximum.

Lorsqu'un étudiant souhaite se connecter sur notre réseau social, le réseau va interroger la base de données pour savoir si son identifiant et son mot de passe correspondent. Ensuite il va lui afficher toutes les informations de son école.

II) Configuration du script de synchronisation

a) Installation des dépendances requises

Vérifiez que *pip* est bien installé sur votre serveur. Pour cela exécutez la commande suivante :

```
$ pip -V
```

Si pip n'est pas installé, procédez à son installation :

```
$ curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py
```

Puis :

```
$ python get-pip.py
```

Ensuite, exécutez la commande suivante en vous plaçant dans le dossier du fichier requirements.txt :

```
$ pip freeze > requirements.txt
```

En exécutant cette commande, vous installerez les dépendances suivantes :

- python-ldap
- pyOpenSSL
- requests
- simplejson
- parsel
- urllib3

b) Connexion à votre annuaire LDAP

Il est nécessaire d'établir la connexion à votre annuaire LDAP en fournissant l'url et le port de connexion à cet endroit.

```
con = ldap.initialize('url_ldap:port')
```

```
#Initialisation de la connexion au LDAP de test (url et port)
con = ldap.initialize('ldap://localhost:10389')
```

Vous êtes alors connecté à l'annuaire LDAP comme un utilisateur anonyme.

c) Protocole BIND

Pour associer votre compte, vous pouvez vous connecter en liant vos informations de connexion. L'échange de données se fait au format LDIF. Il faut donc fournir vos informations de connexion dans ce format.

```
#Envoi d'une requête BIND au serveur LDAP
con.simple_bind_s("uid=admin,ou=system", "secret")
```

Nous utilisons la commande `con.simple_bind_s('username', 'password')` liant un compte à la connexion à l'annuaire LDAP.

cn : common name - La variable cn fait référence à l'objet individuel (nom de la personne, titre du poste, etc.) que vous souhaitez interroger dans votre requête.

dc : domain component - La variable dc fait référence à chaque composant du domaine. Par exemple www.archibook.com s'écrira dc = www, dc = archibook, dc = com

d) La requête

Dans un premier temps, nous créons une variable *ldap_base* contenant les composant du domaine que nous allons explorer.

Puis nous créons la variable *result* contenant le résultat de la recherche. Celle-ci s'effectue grâce à la commande suivante : `con.search_s(ldap_base, ldap.SCOPE_SUBTREE)`. L'argument SUBTREE de la fonction *ldap.SCOPE* permet d'effectuer la recherche dans l'intégralité de l'arbre de l'annuaire LDAP.

```
#Création de la variable ldap_base contenant le DomainComponent du LDAP
ldap_base = "dc=example,dc=com"

#Création de la variable result contenant le résultat de la recherche
result = con.search_s(ldap_base, ldap.SCOPE_SUBTREE)
```

e) Mise en place d'une tâche CRON

Mettre en place une tâche CRON permet d'exécuter le script à intervalle régulier. Ainsi, la synchronisation de modifications/nouvelles entrées sera effectuée tous les jours.

Pour initialiser la tâche CRON, placez-vous dans le dossier du script de synchronisation et exécutez la commande suivante :

```
* 0 * * 0 Synchro.py
```

III) Reste du script

Le reste du script effectue un parsing du résultat de la requête. Il renvoie ensuite les données récupérées sous la forme d'un objet JSON à l'API d'ARCHIBOOK.
CETTE PARTIE DU SCRIPT N'EST PAS A CONFIGURER PAR L'UTILISATEUR !