# TP Bluetooth Classique

Objets Communicants
Dino LOPEZ

## 1. Introduction

La suite BlueZ fournit le support des principales couches et protocoles de la technologie Bluetooth 2.0 et Bluetooth Low Energy (BLE) pour les systèmes d'exploitation Linux. BlueZ est aujourd'hui très stable, même si le support de la technologie Bluetooth 3.0 +HS est sous développement.

En plus d'une API flexible et modulaire qui nous permet de développer des applications Bluetooth, BlueZ nous offre également un ensemble de commandes qui nous permettent de manipuler, tester une interface et debugger des applications.

## 2. Utilisation des outils BlueZ

Avant de programmer une application, nous allons commencer par réaffirmer par la pratique les connaissances acquises en cours. Les prochains exercices ont donc pour objectif de vous familiariser avec le fonctionnement des équipements Bluetooth 2.0 par le biais des commandes fournies par Bluez.

1) Commencez tout d'abord par lister toutes les interfaces Bluetooth disponible dans votre système avec la commande hciconfig. Donnez l'identifiant de ces dispositifs, leur adresses Bluetooth et les différents états (disponibles dans la ligne 3 de la sortie de la commande hciconfig) qui ont été activé pour ces dispositifs.

2) Toujours avec la commande hciconfig, désactivez et activez à nouveau votre dispositif Bluetooth. Quelle est la commande pour activer / désactiver un dispositif Bluetooth ?

3) Lorsqu'un dispositif Bluetooth cherche des voisins pour former un piconet (le master), il doit exécuter la procédure inquiry. Le dispositif qui écoute pour un inquiry (le slave) doit être en mode inquiry scan. Désactivez les modes « page scan » et « inquiry scan » de votre interface Bluetooth. Ensuite, configurez uniquement un des dispositifs Bluetooth pour répondre uniquement aux « inquiries ». Quelles sont les commandes nécessaires ?

4) Pour connaître les services qu'un équipement avec une interface Bluetooth peut offrir, il doit implémenter le protocole SDP. La commande sdptool (voir le manuel d'aide avec la commande « man sdptool ») permet d'interagir avec le protocole SDP de BlueZ. Indiquez la commande qui vous permet d'obtenir la liste de services disponible dans l'un de vos ordinateurs (ou de votre téléphone portable s'il dispose d'une interface Bluetooth).

## 3. Création d'un serveur RFCOMM en Python

Note : Avant de continuer, vous devez arrêter le daemon bluetoothd (e.g. systemctl stop bluetooth), puis le relancer en mode compatibilité avec « # bluetoothd -C ».

Pour programmer un serveur RFCOMM, et pour vous faciliter la tâche, vous utiliserez la librairie python-bluez, qui est un wrapper au tour de l'implémentation Bluez en C.

5) A l'aide de l'exemple disponible en https://github.com/karulis/pybluez/blob/master/examples/simple/rfcomm-server.py écrivez un serveur d'écho RFCOMM. Générez un identifiant de service UUID à 128 bits. à l'aide de la commande uuidgen. Fixez également le port d'écoute au numéro 2 (à la place de PORT_ANY). Votre serveur devra lire les messages envoyés par un client Bluetooth et lorsque le client part, votre serveur devra se mettre en attente de la connexion suivante.

6) Expliquez les lignes 9, 10, 11, 17, 26, 31 et 32 de l'exemple. Vous trouverez la documentation de l'API PyBluez (i.e. de la librairie python-bluez) à l'adresse http://pybluez.googlecode.com/svn/www/docs-0.7/index.html

7) Expliquez en quoi un serveur RFCOMM est similaire à un serveur TCP.

8) Avec l'interface Bluetooth USB connecté à votre ordinateur (et en étant sûr que sa configuration permet une connexion avec un périphérique distant), exécutez (debuggez si nécessaire) votre serveur RFCOMM.

## 4. Création d'un client RFCOMM en Python

9) Programmez maintenant votre client RFCOMM en Python grâce à la librairie python-bluez et à l'aide de l'exemple disponible à l'adresse https://github.com/karulis/pybluez/blob/master/examples/simple/rfcomm-client.py. Bien évidemment, votre client devra chercher l'UUID et se connecter au port que vous avez configuré sur votre serveur.

10) Expliquez les lignes de l'exemple. Vous trouverez la documentation de l'API PyBluez (i.e. de la librairie python-bluez) à l'adresse http://pybluez.googlecode.com/svn/www/docs-0.7/index.html

11) Avec l'interface Bluetooth USB connecté à votre ordinateur (et en étant sûr que sa configuration permet une connexion avec un périphérique distant), exécutez (debuggez si nécessaire) votre client RFCOMM.

12) Vérifiez que votre client et votre serveur peuvent communiquer et donnent le comportement attendu.

## 5. Stream vidéo par le biais d'une liaison Bluetooth

13) Sachant qu'on peut envoyer un stream audio par le réseau avec « `mplayer -ao pcm:file=/dev/stdout -really-quiet http://listen.livestreamingservice.com:8062 | nc -u 10.0.1.10 1234` »

et jouer le stream de la manière suivante avec VLC « `nc -u -l 1234 | vlc -` » écrivez un script python

a. capable de recevoir le stream audio via STDIN et la renvoyer via une connexion RFCOMM vers le lecteur VLC.

b. capable de recevoir l'audio via la liaison RFCOMM et la transmettre vers le lecteur VLC

```
mplayer | (RFCOMM) -----> (RFCOMM) | vlc
           Python          Python
```