OS embarqués : projet

Thibaut EHLINGER et Benjamin HERB
Université de Strasbourg

Mardi 13 décembre 2016

Dans ce rapport, nous allons vous présenter notre travail concernant le TP2 d'IoT. Pour rappel, il fallait dans un premier temps modifier certains paramètres (PanID, canal radio) et compiler une version de `Contiki OS` compatible avec les nœuds de la plate-forme FIT/IoT-lab. Ensuite il fallait flasher les six nœuds qui nous étaient attribués avec le binaire obtenu. Ensuite, nous avons dû mettre en œuvre RPL, Foren6 et Stack RIME. Nous allons vous présenter les différentes étapes de ce tp.

1 Contiki OS

Contiki OS est un OS open source spécialisé dans les microcontrôleurs à basse consommation d'énergie reliés à un réseau. Dans un premier temps, nous avons modifié le code de Contiki OS. Pour modifier le canal il fallait modifier `RF2_XX_CHANNEL` dans le fichier `platform/openlab/radiorf2xx.c`. Nous lui avons affecté la valeur **17**. Pour le PanID, nous avons fait un `define IEEE802154_CONF_PANID` et nous avons utilisé l'ID `0x0007`.

Pour compiler Contiki OS, il fallait utiliser la commande `make` avec la bonne cible :

```
make target=iotlab-m3
```

Une fois le binaire généré, nous l'avons téléversé sur nos contrôleurs avec

```
node-cli -i 56476 -l strasbourg,m3,2+30-34 -up tutorial_m3.elf
```

2 RPL

Une fois nos contrôleurs dotés d'un OS, nous avons mis en œuvre RPL, un protocole de routage IPV6 prévu pour les WSN (*Wireless sensor network*). RPL construit un DODAG (*Destination Oriented Directed Acyclic Graph*) orienté vers le BR (*Border router*) de notre réseau.

Pour information, notre sous-réseau a pour adresse IPv6 : `2001:660:4701:f0a6/64`. Notre *border-router* est le nœud numéro **m3-2**. Pour déployer notre réseau nous avons dû :

1.