TP1- 6LoWPAN
Introduction à l'émulateur Cooja

Important : Un rapport doit être rendu au plus tard 5 jours après le TP. Le compte rendu doit contenir : un rapport avec des captures d'écrans bien expliquées et argumentées. Le dépôt doit être sur la plateforme « elearning » et il faut respecter les groupes « apprentis » et « initiaux »
https://elearning.u-pem.fr/

Objectif :

- ☐ Prendre en main l'émulateur Cooja du système d'exploitation Contiki
- ☐ Comprendre le mécanisme de collecte des données dans le réseau de capteurs
- ☐ Apprendre à créer des simulations avec l'émulateur Cooja

Prérequis :

- ☐ Protocole IP et adressage IPv4
- ☐ Routage statique et dynamique

L'objectif de ce TP est d'apprendre à utiliser l'émulateur Cooja pour pouvoir simuler un réseau 6LoWPAN.

Introduction

Cooja est un émulateur réseau basé sur Contiki. Contiki est un système d'exploitation pour les réseaux de capteurs qui supporte la technologie 6LoWPAN. Cooja permet d'exécuter des programmes sur Contiki sans avoir besoin du matériel.

Préparation de l'environnement de développement

Pour éviter d'installer tout l'environnement de développement, nous allons utiliser une machine virtuelle (VM) nommée « Instant Contiki ». Les étapes à suivre pour mettre en place la VM :

1- Télécharger la machine virtuelle nommée : « ***Instant Contiki 2.7*** ».
Le site : https://sourceforge.net/projects/contiki/files/Instant%20Contiki/
2- Pour faire tourner cette machine virtuelle, il faut télécharger un lecteur des machines virtuelles comme *VirtualBox* ou *VMPlayer*.
3- Une fois que vous avez installé le lecteur des machines virtuelles avec la machine Instant Contiki, vous allez avoir besoin du login et mot de passe dans ce cas vous utilisez login : **user** et le mot de passe : **user**.
4- Installer à l'aide de la commande « ***apt-get install*** » les packages suivants : « ***ant*** », « ***gcc-msp430*** », « ***msp430-libc*** », « ***gcc-avr*** », et « ***avr-libc*** »
5- Lancer l'émulateur Cooja : Pour lancer l'émulateur "Cooja", il suffit d'aller dans le répertoire "**contiki/tools/cooja**", ensuite d'exécuter la commande : « ***ant run*** »

II) Prise en main

1) Le contenu du répertoire « Contiki » et leurs usages

- **apps** : contient des applications comme webbrowser, telnet etc.
- **core** : les codes sources du cœur de Contiki
- **core/dev** : les codes sources des périphériques comme LED, batterie, capteur, bouton, etc.
- **core/net**: contient l'implémentation des protocoles MAC, et routage comme RPL, IPv6 et IPv4, les files d'attentes des paquets, et les buffers etc.

- **cpu**: contient les fichiers source liés à l'unité de calcul de modules (objets/nœuds)
- **examples**: contient l'implémentation des applications
- **tools**: contient les outils pour tester des applications

2) Exécuter un premier exemple « Hello World » avec Cooja

Cet exemple vous montre comment ouvrir et charger un code Contiki existant. Le plus simple est de commencer avec la compilation du code le plus simple nommé "Hello, Word". Cet exemple est basé sur un réseau formé de deux nœuds qui affichent uniquement le message « Hello, Word ».

Pour ouvrir un fichier de simulation existant (l'extension du fichier de simulation est : « csc »), il faut aller dans le menu « File → Open simulation → Browse ». Ensuite, il faut aller dans le répertoire : « ***contiki/examples/hello-world*** » pour charger le fichier « hello-world-example.csc ». Ou bien il suffit d'exécuter la commande suivante à partir du répertoire « ***contiki/examples/hello-world » : make TARGET=cooja hello-world.csc***

Une fois la compilation terminée, cliquez sur le bouton « Start » pour lancer la simulation.

Q1- Analyser les traces qui se trouvent dans la fenêtre « ***Mote output***».

2) Exécuter l'exemple de collection de données nommée « data collection»

La collection des données est un processus commun de l'ensemble des applications de réseau de capteurs. Ce processus consiste à remonter les informations captées vers le nœud collecteur nommé le « *Sink node*». La distribution de Contiki contient un exemple qui permet d'illustrer le fonctionnement du mécanisme de collecte des données. Pour exécuter ce programme, il faut aller dans le répertoire « contiki/examples/rime» et sélectionner le fichier «*example-collect*». Une fois la compilation terminée, cliquez sur le bouton « Start » pour lancer la simulation.

Vérifier au niveau de la fenêtre "***Network***" et en particulier dans l'anglet « ***View*** » que l'identificateur des nœuds, et la partie « Radio » sont sélectionnés.

Q1- Lors que vous sélectionnez un nœud, vous pouvez observer trois zones (verte, grise et blanche). A quoi correspondent-elles ?

Q2 - Faites correspondre les paquets observés dans « Radio messages » avec ceux affichés dans les fenêtres « Mote output » et « Timeline ». Il suffit de faire un clic droit sur une trame et choisir une option dans le menu « Show in », ou sinon appuyer sur « espace » après avoir sélectionné une trame. Que constatez vous ?

- Quel est le protocole MAC utilisé par les motes? Quel est son Channel check rate?
- Quelle est la fréquence de réveil de la couche radio d'un mote d'après la Timeline?
- Observer dans la fenêre PowerTracker les valeurs de duty cycle de chaque nœud (Mote Radio duty Cycle).

Q3- Analyser les traces qui se trouvent dans la fenêtre « ***Mote output***».
Q4- Quel est l'identificateur du nœud collecteur (***Sink***) ? Donner la topologie de ce réseau.

2) Création d'une nouvelle simulation sous Cooja avec le programme « *Collect View* »

Le programme "**CollectView**" est une application Java qui peut être utilisée avec les capteurs Tmote Sky ou bien avec l'émulateur réseau Cooja. Pour pouvoir utiliser ce programme avec l'émulateur Cooja, il faut suivre les étapes ci-dessous :

- Créer une nouvelle simulation : **File →New simulation**
- Compiler le programme « collect-view » et créer 8 Motes : **Motes →Add motes→Create new mote type→Sky mote**. Dans le champ « contiki process » sélectionner le programme à compiler « **contiki/examples/collect/collect-view-shell.c**». Ensuite, créer 8 motes à l'aide du bouton Create.
- Activer les options **Mote ID** et **Radio environment (UDGM)** dans la fenêtre **« Network»**
- Lancer la simulation avec le bouton « ***Start*** »
- Lancer le programme « ***Collect View*** » avec un clic droit sur le nœud collecteur
- Lancer le processus de collection avec le bouton "***Start collect***" et envoyer la commande à tous les nœuds du réseau avec le bouton "***Send command to nodes***" pour récupérer les données de manière périodique.

Q5- Donner la topologie du réseau.

Q6- Analyser la consommation d'énergie des nœuds. Comment la consommation d'énergie est-elle répartie ? Quelle est l'opération la plus coûteuse en termes de consommation d'énergie ?

Q7- Quel est l'impact de la position des nœuds dans la topologie sur la consommation d'énergie ?

Q8- Quel est l'impact de la fréquence de récolte de données sur la consommation d'énergie dans le réseau ?

Le monde uIP

Une des caractéristiques du système Contiki est de supporter nativement le protocole Internet (IP). Cette partie montre comment configurer un réseau IPv6 à faible consommation d'énergie avec Contiki. Vous utilisez l'émulateur réseau Cooja, mais le principe est le même avec un vrai matériel.

- Diffusion en mode UDP sous IPv6

Créer une nouvelle simulation nommée « udp-com ». Créer et configurer 8 capteurs afin qu'ils exécutent le programme « broadcast-example.c » qui se trouve dans le répertoire « contiki/examples/ipv6/simple-udp-rpl/ ».

Q9- Analyser les données échangées entre les capteurs.

- Lancer la simulation de l'exemple existant "rpl-collect"

- Lancer l'émulateur "Cooja" si vous ne l'avez pas encore fait voilà la commande:

```
cd contiki/tools/cooja
ant run
```

- Dans Cooja, charger et compiler le programme « ***rpl-collect*** » dans le répertoire « ***contiki/examples/ipv6/rpl-collect***» et en particulier le fichier « ***collect-tree-dense-noloss.csc*** »
- Lancer la simulation avec le bouton ***« Start*** ».
- Un réseau formé de 25 nœuds qui exécutent IPv6 avec le protocole de routage RPLapparait dans la fenêtre « Simulation Visualizer ». Parmi les 25 nœuds, un est le nœud collecteur (Sink) et les autres sont des nœuds qui envoient périodiquement des données vers le collecteur.
- Une fois la simulation lancée, le temps de simulation atteint les 70000 ms environ, alors les données commencent à apparaitre au niveau du nœud collecteur. En réalité, le collecteur affiche les données

reçues via son port série. Le programme CollectView écoute ce port sérié (émulé) et il affiche les données reçues.
- Afficher la structure du réseau à l'aide de l'onglet « Network Graph » du programme CollectView.

Q10- Quelle est l'adresse du nœud collecteur ? Ce dernier est-il mobile ou fixe ? S'il est mobile fixer ce nœud à une position fixe.

- Analyser l'ensemble des paramètres réseau à l'aide du programme CollectView.

Q11- Analyser la courbe de l'historique de la consommation d'énergie du réseau. Quel est le nœud du réseau dont la consommation d'énergie est la plus importante ? Pourquoi ?

Créer et configurer votre simulation

Dans cette partie, vous utilisez le même code source C que la simulation précédente, mais vous allez configurer vous-même.

- Créer une nouvelle simulation : File-> New simulation
- Donner un nom à cette simulation par exemple : « ***ExRPLSimulation*** ».
- Sélectionner le modèle de propagation radio : « ***Unit Disck Graph Medium (UDGM) : Distance Loss*** ». Ensuite appuyez sur le bouton « Create »
- Créer le nœud collecteur : ***Mote Types → Create mote type → Sky Mote Type***
- Donner le nom **« *Sink* »** au nœud. Dans la section **« *Contiki process / Firm..* »,** sélectionner le fichier qui contient le code pour le nœud Sink **« *udp-sink.c* »** dans le répertoire **« contiki/examples/ipv6/rpl-collect »**
- Compiler le code à l'aide du bouton « Compile »
- Créer un seul nœud « Sink » à partir de la boîte de dialogue « Add motes (Sink) »
- Créer les nœuds émetteurs : ***Mote Types -> Create mote type -> Sky Mote Type***
- Donner le nom **« *Sender* »** au nœud. Dans la section **« *Contiki process / Firm..* »,** sélectionner le fichier qui contient le code pour le nœud Sink **« *udp-sender.c* »** dans le répertoire **« contiki/examples/ipv6/rpl-collect »**
- Compiler le code à l'aide du bouton « Compile »
- Créer 9 nœuds « Sender» à partir de la boîte de dialogue « Add motes (Sender) »
- Dans la fenêtre «**Simulation Visualizer**», ouvrir le menu « Visualize skin », ensuite sélectionner les paramètres : Mote IDs et Mote Type. Cela va permettre d'afficher l'identificateur du nœud ainsi que la différence entre les types de nœuds : collecteur (Sink) ou capteur classique (Sender).
- Lancer le programme CollectView : Clic droit sur le nœud Sink (ID 1) → Cliquer sur le menu Open mote plugin → Collect View
- Lancer la simulation et commencer la collecte des données.

Q12- Analyser l'ensemble des paramètres réseau à l'aide du programme CollectView après 10000ms.

Développer une première application avec Contiki

Dans cette partie, vous allez apprendre à développer un simple programme Contiki. De plus, le processus de développement de la création du projet, l'écriture du programme en langage Contiki, la compilation et l'exécution avec l'émulateur MSPsim et/ou le Mote Tmote Sky.

Dans cet exemple, le projet Contiki est basé sur code source C et le Makefile. Le fichier source C contient le programme et le Makefile contient les règles de compilation du programme pour générer un exécutable Contiki.

1- Création du répertoire du projet

Créer deux répertoires : un pour les projets Contiki et l'autre pour le projet Hello, Word.

```
mkdir projects
cd projects
mkdir hello-world-project
cd hello-world-project
```

2- Création du fichier Makefile

Makefile décrit la procédure de compilation du programme. Une fois que le fichier Makefile est créé avec votre éditeur préféré (ex. gedit, vi, emacs, …), vous commencez à remplir ce fichier comme suit :

```
CONTIKI=/home/user/contiki
include $(CONTIKI)/Makefile.include
```

Sauvegarder le fichier Makefile et commencer à écrire votre premier programme.

3- Créer le fichier du programme

Créer un fichier programme nommé « hello-world.c », taper le code ci-dessous :

```
#include "contiki.h"

PROCESS(hello_world_process, "Hello world process");
AUTOSTART_PROCESSES(&hello_world_process);

PROCESS_THREAD(hello_world_process, ev, data)
{
  PROCESS_BEGIN();
  printf("Hello, world!\n");
  PROCESS_END();
}
```

Commentaire du code :

- La première ligne inclut les en-têtes (header) C du Contiki. Cette ligne est nécessaire pour inclure les librairies et les définitions du Contiki.
- La deuxième ligne déclare le processus Contiki. Ce processus possède une variable nommée (hello_world_process) et une chaine de caractère nommée («Hello world process). La chaine de caractère est utilisée pour le débogage et comme une commande pour le Shell de Contiki.
- La troisième ligne informe le système Contiki que le programme hello_world_process doit se lancer automatiquement lors du démarrage du système.
- La quatrième ligne définie le processus hello_world_process. Les arguments ev et data dans la macro « PROCESS_THREAD() » sont des variables du nombre d'évènement et de données respectivement que le processus peut recevoir.
- Le processus lui-même esr définit entre les deux macros : PROCESS_BEGIN() et PROCESS_END(). Dans ce cas le processus se limite à afficher le message « hello, world ».

Sauvegarder le fichier « hello-world.c ».

4- Configurer la variable TARGET

Dans la fenêtre terminale, il faut indiquer la plateforme par défaut. Pour ce projet, on peut définir la plateforme « sky » avec cette commande :

```
make TARGET=sky savetarget
```

5- Compilation du projet

A ce stade vous pouvez compiler le projet pour la première fois avec la commande suivante :

```
make hello-world
```

Cette commande entraine à la compilation de l'ensemble du système Contiki.

6- Tester le projet dans MSPsim

Le programme Hello, world peut être testé sans matériel avec l'utilisation du simulateur MSPsim.

```
make hello-world.mspsim
```

La fenêtre « USART1 Port Output » montre la sortie du port série du capteur. Normalement le message « Hello, world » est affiché juste après les messages du boot du Contiki.
La fenêtre SkyGui montre le capteur Tmote Sky avec le Botton « reset » qui nous permet de réinitialiser le capteur. Réinitialiser le capteur à nouveau.
Arrêter la simulation avec Ctrl+C ou de fermer simplement la fenêtre.

7- Exécuter le projet sur une plateforme matériel Tmote Sky

- Insérer le capteur Tmote Sky (TelosB) sur le port USB du PC.
- Il faut configurer le VMware Player ou le VirtualBox pour accéder au capteur. Cette procédure peut être effectuée dans le menu de la machine virtuelle. Dans le cas de VMware Player, il faut aller dans (Virtual Machine -> Removable Devices-> future devices mote sky).
- Une fois cette procédure est terminée, le capteur Tmote Sky devient accessible par Instant Cintiki. Vous pouvez tester cette partie par la commande : **`make sky-motelist`**
- Une fois que le capteur est visible via Instant Contiki, il est possible de charger le programme « Hello, World » sur le capteur avec la commande suivante : **`make hello-world.upload`**
  Remarque : si vous avez une erreur pour charger le code sur le nœud