AMIO : Applications mobiles et Internet des Objets
Troisième année

T.P. 6 : Web of things avec COAP/HTTP

L'objectif de ce dernier TP est d'interroger directement les motes depuis le web grâce au protocole COAP (https://tools.ietf.org/html/rfc7252).

Partie 1 : COAP côté client

Plusieurs solutions sont possibles pour la partie client : utiliser un client COAP natif ou utiliser un client HTTP classique, mais il faut alors disposer d'une gateway faisant la traduction entre COAP et HTTP. Nous allons opter pour la première solution.

- Exercice 1 – Client COAP Copper
  - Installer le client COAP textuel fourni par la libcoap:
    - *sudo apt-get install libcoap-1-0-bin* **ou** *sudo apt-get install libcoap2-bin (sur les systèmes récents.*
      (documentation: http://manpages.ubuntu.com/manpages/bionic/man5/coap-client.5.html)
  - Les requêtes à taper seront de la forme:
    - *coap-client -m get coap://[@ipv6_sensor]/path/*
    - exemple : *coap-client -m get coap://[aaaa::212:7402:2:202]/.well-known/core*
    - exemple 2 : *coap-client -m get coap://[aaaa::212:7402:2:202]/sensors/light*

Partie 2 : COAP côté serveur

Il existe plusieurs implantations de serveur COAP pour Contiki.

- Pour la partie 2, utiliser de préférence l'implantation « Erbium », qui implante l'ensemble du protocole COAP (validé lors de plugtests ETSI en 2012) :
  - Installer la version de Contiki incluant Erbium: http://people.inf.ethz.ch/mkovatsc/erbium.php
    git clone --recursive git://github.com/mkovatsc/SmartAppContiki.git
  - Pour information, la version de COAP disponible nativement dans Contiki 2.7 et utilisée dans le tutoriel suivant n'implante pas toutes les spécifications du protocole.
    http://anrg.usc.edu/contiki/index.php/REST_example_running_on_Cooja_and_Sky_motes,

- Exercice 2 – COAP dans Cooja
  - Charger la simulation *SmartAppContiki/examples/er-rest-example/server-only.csc* faisant intervenir un serveur COAP et un routeur RPL. Quelles sont les adresses IPv6 de ces deux nœuds ? Lire le code du serveur *er-example-server.c* . Quelles ressources sont a priori exposées ?
  - Connecter maintenant le routeur RPL au socket via un bridge : *make connect-router-cooja* Quelle est l'adresse IPv6 publique du routeur ? Vérifier la connectivité avec *ping6*.
  - Lancer Wireshark et écouter l'interface du bridge. Accéder depuis Copper aux ressources du capteur émulé via Cooja. L'ensemble des ressources prévues sont-elles accessibles ?
  - Quel est le port par défaut utilisé par COAP ?
  - Tester l'ensemble des ressources disponibles et observer avec Wireshark les messages COAP transitant entre le client et le serveur.

- Exercice 3 – COAP sur les motes
  - Copier et renommer le fichier originel du serveur (cp -old).
  - Modifier le serveur pour activer la capacité d'exposer via COAP la luminosité mesurée par le capteur sous l'arborescence *sensors/light* .
  - Vérifier le bon fonctionnement de votre serveur au sein du simulateur.
    - *make TARGET=cooja server-only.csc* (recompile au passage er-example-server.c)
  - Déployer maintenant votre code sur un mote réel. Vous aurez besoin :
    - de flasher un premier mote exécutant le serveur COAP : *make TARGET=sky er-example-server.upload*
    - de flasher un second mote exécutant une instance de border router (*border-router.c*) servant de passerelle : *cd ../ipv6/rpl-border-router*
      make TARGET=sky border-router.upload
    - connecter le routeur : *make connect-router*
    - note : pour connaître l'adresse IPv6 de lien local d'un mote, il faut le connecter en USB, lancer une commande login : *make login TARGET=sky MOTE=2* , et le redémarrer
  - Vérifier la connectivité (*ping6*) et le bon fonctionnement via Copper du réseau de capteur ainsi déployé.

- Exercice 4 – Ajout d'une ressource COAP

  - Ajouter enfin la possibilité de consulter en plus de la luminosité, la température des motes sous l'arborescence *sensors/temp* et leur niveau de batterie *sensors/battery*. A quelle limitation faites-vous face ?

Partie 3 : HTTP côté serveur

Cet exercice vise à rendre les capteurs directement consultables via une liaison WiFi et HTTP.

- Nous allons faire ici l'équivalent de l'exercice précédent en remplaçant COAP avec le protocole HTTP qui assure une plus grand compatibilité avec les clients (bien que moins adapté).
  - Préparer un capteur exposant un service REST via HTTP où doivent être accessibles les mesures de température, luminosité et humidité.
  - Les instructions suivantes vous seront utiles (penser à faire un *make clean* avant de changer les options de compilation):
    - http://anrg.usc.edu/contiki/index.php/REST_example_running_on_Cooja_and_Sky_motes
  - Tester avec votre navigateur web que votre service web est bien accessible.

- Ensuite, il faut disposer d'une infrastructure réseau permettant aux capteurs et au client web de communiquer.
  - Partager la connexion WiFi du PC
  - Pour faire le lien avec des clients externes il faut exécuter un proxy HTTP à sur l'ordinateur faisant passerelle est disponible à cette adresse:
    - http://thibault.cholez.free.fr/teaching/OCI/proxy_IoT.c :

- Interroger un capteur via le proxy avec votre smartphone. Si les données sont accessibles, elles pourront être affichées sur Hololens dans le module RVA.