Advanced Networking And Wireless Systems
Computer Engineering Master Degree
University of Pisa

IoT Project

6LoWPAN/RPL Wireless Sensor Network with CoAP Proxy for resource observing

UNIVERSITÀ DI PISA

Contents

1. Introduction

The aim of this project is to design and implement a 6LoWPAN Wireless Sensor Network that uses RPL routing protocol with the Contiki development environment. Every node is intended to sense the temperature value and to keep it as a CoAP resource. Moreover, with the CoAP protocol, a Proxy is implemented in order to "observe" the temperature values of the nodes and to forward when a Client request them.

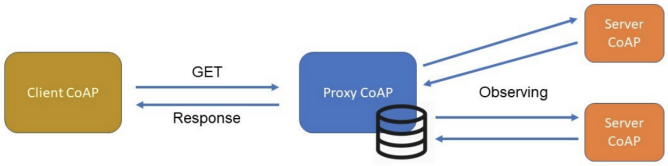

Figure 1.1: Sample of the application

Both the Proxy and the Client are implemented with Californium, the Java library for CoAP. The Proxy, retrieves by observing the values of the temperature from every node deployed in the network and stores them in a private cache. When the Client request a temperature value of a certain node by specifying the ID of the node, to the Proxy by performing a GET operation, the latter one answer with the requested value.

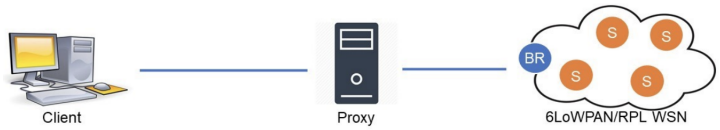

Figure 1.2: Architecture of the application

2. Design

2.1 ClientCoAP

The Client is implemented with JAVA. It allows to perform a request with a GET operation to the Proxy in order to obtain the temperature value of a node. The node of interest can be specified by inserting the numeric ID of it at runtime, the Client prepares the CoAP URI and forward the request to the Proxy.

2.2 ProxyCoAP

Also the Proxy is implemented with JAVA. At the startup it needs as an input the number of nodes deployed in the network in order to start the "observe" operation to every node. As already explained, the Proxy stores the last temperature value of every node in a cache. Hence, when a client makes a request for a value of a certain node, it will receive the last received value. The cache is implemented as an array, every entry of the array correspond to a node in the WSN.

2.3 CoAP Node

A node is implemented with the Contiki environment (C language) using the Z1 mote as model. With the purpose of simulation, the temperature value is randomly generated and sent to the Proxy every time that a variation is detected. The REST paradigm allows to keep the temperature value as a CoAP resource.

3. Execution Tutorial

In order to execute the project it is necessary to open the Cooja simulator on Contiki. Once the simulator is opened, we're now able to add the nodes and deploy the network. First of all the RPL border router is added to the network, then we can insert the other regular nodes. The nodes inserted in the WSN are around 20-30.

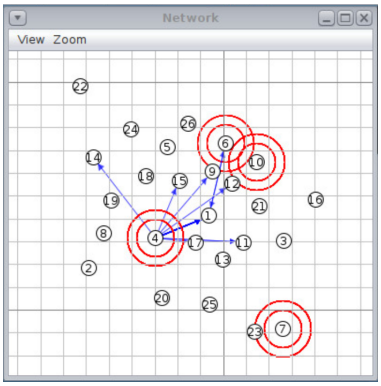

Figure 3.1: WSN deployed with Cooja simulator

It is necessary to define the Serial Socket port for the Border Router, by selecting *Tools*, then *Serial Socket (SERVER)* and finally select the Border Router and the listen port. It is now neccessary to tun *Tunslip6* by inserting the following command in a terminal:

```
make connect-router-cooja PREFIX="abcd::1/64"
```

This is for creating the connection between the mote (border router) and the host. It is reccomended now to start the simulation in Cooja and give some time before proceeding in order to create all the routes.

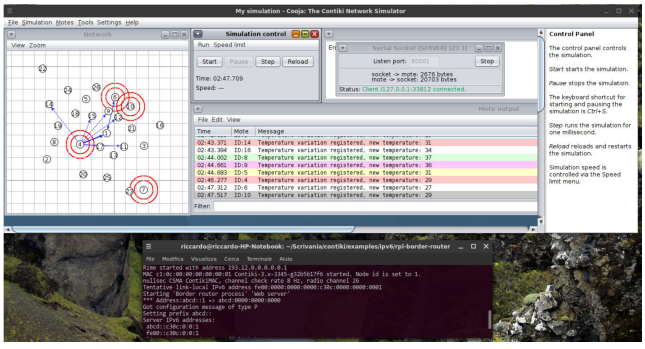

Figure 3.2: The running simulation in Cooja

Now it is possible to run the Proxy and the Client. When the Proxy starts, it asks as an input the number of nodes deployed in the network. Once the number is inserted, it will start with the "observe" operation to the nodes and will report the temperature value of everyone.

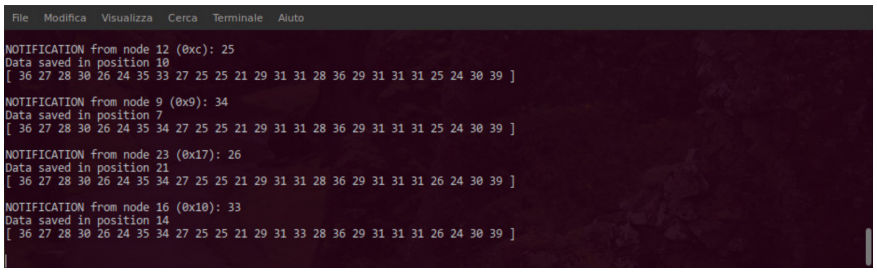

Figure 3.3: Sample of a Proxy execution with 25 nodes deployed

The client, at runtime, can request the temperature value of every node with a GET operation and the Proxy will reply by taking the last received value from the cache.

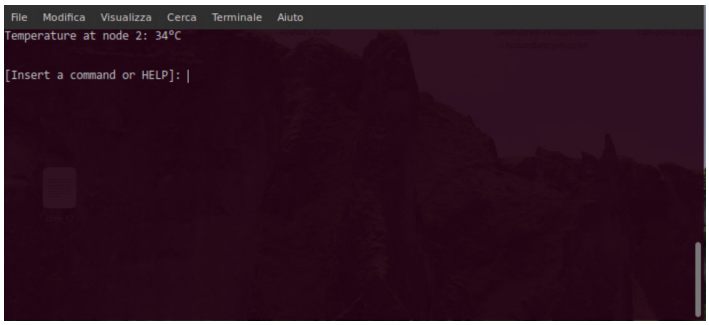

Figure 3.4: Sample of a Client request for a temperature value