Group project

Ilias Boulif, Jérémy Bricq, George Rusu

4 mai 2019

Table des matières

1 Introduction

In the course of Mobile and Embedded Computing we have seen two ways to send sensor data from IoT devices to MQTT subscribers. The first way is with MQTT on TCP and the last way is with MQTT-SN on UDP. However we have seen during the exercises sessions that with an IP network based on 6LoWPAN and RPL it is difficult to obtain good network performance.

The aim of this project is to propose an implementation of an IoT network using *Rime* where the sensor data is published through an MQTT-Rime gateway to normal MQTT subscribers. To achieve that we have implemented as requested by the project's statement, a tree-based routing protocol using Rime and an MQTT-Rime gateway.

-expliqué stucture

The source code of this project implemenation is available at [1].

2 General structure of our system

-le réseau basé sur un root , transmet les info au subsc -des sensors node qui recup les info et les envoi au root -les sensors se connecte entre par un réseau sans fil (rime) il comminuque entre en s'envoyant diff msg pour s'assurer du bon fct du reseau -le réseau est pensé pour pouvoir envoyé des info au root meme si il y a des sensors qui sont déplacé ou suprimé - le root est connecté au gatesay qui permet d'envoyer vers subscriber - le root recoit mode depuis le gateway , il est envoyé dans les alives responses packet jusqu'en bas

Our system is a network composed of one root node and a number of sensor nodes. The network depends on the root which is the node responsible for transmitting the information to the subscribers. The sensor nodes for their part, collect the information for the root and forward them to him. use rime ! The nodes communicate by sending different kind of messages (see Section 2.1.

2.1 Message format

In our implementation we have 3 packets structures. The node interprets the packet according to his type. We will explain each kind of packets and discuss the different types used by nodes to communicate between them.

Description of *packet*

1. Type : [DISCOVERY_REQUEST, DISCOVERY_RESPONSE, ALIVE_REQUEST, ALIVE_RESPONSE]
2. Rank : the rank of the emission node
3. Mode : the mode of sent [DATA_ON_CHANGE, DATA_PERIODICALLY]
4. HaveSubscriber : Boolean that indicates if someone is subscribed to some topics

Description of *data_packet*

1. Type : [SENSOR_DATA]
2. NodeSrc : The node where the information come from
3. NodeRank : The rank of the node source
4. DataTemp : The temperature sent
5. DataOthers : A place used for another kind of data

Description of *data_packet_aggregate*

1. Type : [SENSOR_DATA_AGGREGATE]
2. numberPacket : The number of packets
3. packet1 : The first packet
4. packet2 : The second packet

DISCOVERY_REQUEST When a node starts, he is alone and has a rank of 0. To connect to a network, Each sensor node need to send in broadcast a discovery request to find a parent connected to that network. Even when connected, the node will still keep sending discovery request in order to find a better parent, if exists. The lower is the rank (except a rank of 0) and the nearest is the node from the root node.

DISCOVERY_RESPONSE When a node receives a discovery request packet he replies with a discovery response with his rank. The node that has made the request receive now the response and assign the his rank the value of the rank of his parent + 1.

ALIVE_REQUEST Each node has to keep sending an alive request in order to know if his parent is still available in the network. After 4 alive requests without any response, the child considers that he has no parent. If he has no more parent, his rank return back to 0 and he has to research for a new one.

ALIVE_RESPONSE When a node receives an alive request, he has to replies with an alive response to his child to validate his presence. Each alive response is accompanied with the rank of the parent. In this way, if the parent is disconnected from the network (rank = 0), the child can also put his rank to 0 and can disconnect.

SENSOR_DATA A sensor data packet is a packet that contains some information collected by the sensor. This packet travels from his sensor to the subscribers by going forwarded by the nodes on the path to the root.

SENSOR_DATA_AGGREGATE A sensor data aggregate packet is a packet that contains many sensors data packets in one. This packet will be explained in details in the Section 4 : Optimizations.

3 Network organization and routing

4 Optimizations

5 Conclusion

Références

[1] Github repository, *MobileEmbeddedComputing*, Available at : `https://github.com/georgesrusu/MobileEmbeddedComputing`.