POLITECNICO
MILANO 1863

Middleware technologies for distributed systems
Project 5

2020-2021

Authors:
Andrea Falanti
Federico Ferri
Abanoub Faltaous

Professors:
Prof. Alessandro Margara
Prof. Luca Mottola

Contents

1 Introduction

The project demands to create a system composed by multiple IoT devices that when they are within the same broadcast domain, are considered in contact. The contacts among people are reported periodically to the backend on the regular Internet. And whenever a device signals an event of interest, all people that was in contact must be notified. The project is based on the fact that all nodes are constantly reachable from a static IoT device that acts like an IPv6 border Router

- Seven days moving average of new reported cases, for each county and for each day
- Percentage increase (with respect to the day before) of the seven days moving average, for each country and for each day
- Top 10 countries with the highest percentage increase of the seven days moving average, for each day

2 Implementation

The implementation requires to use the COOJA simulator of ContikiNG for the network simulation of the IoT devices. While for the backend we decide to use Akka, that help us for message handling bla bla. We decide to use publish/subscribe architecture style for the communication between the simulator and Akka. The communication happens through MQTT both from Akka either COOJA. We decide also to use the broker Mosquitto, that are (in our case) already on the machine, to finalize the publish/subscribe through MQTT.

For the simulation of the IoT devices we use nodes to publish and subscribe and a rpl border router to allow nodes to connect to the broker on our machine. Nodes implement two main functionalities: one is to send in broadcast to all other visible nodes its own id, this is implemented :

- First, nodes use a state machine to connect with mqtt to the message broker and then publish data periodically(these data represent the events of interest)
- Second, is to send in broadcast to all other visible nodes its own id, this is implemented using udp on an IP address set to the link local all-nodes multi-cast address. When a node receives this kind of message, it calls a callback, that allows that node to publish its own id and the id of the sender of the message if it can
- Third, nodes receive data from their subscription through MQTT

In our implementation there are three topics: one when two people are in contact, one for the event of interest and one for the notification. Our nodes publish on the first two topic, while are subscribed to the third one, vice versa for Akka that is subscribed to the first two topics while for the third one publish. When a node from the simulation send a publish on a topic the broker on our machine takes care to publish it on a broker server, we decide to publish on test.mosquitto.org:1883. Then when a message arrives on Akka, it is handled in different ways based on the topic from which it receive the message. If the message comes is a contact message our server saves the message; if it is a message of interest, our server retrieves the sender of the message and publish a notification for each node that was in cantact with the sender. In the end these notification arrives to all nodes but only nodes that was in contact print something, this print is made looking the data of the message. In Akka we have three main parts: the server, the server-actor and the device-actor. The server creates the MQTT connection and the server-actor ,subscribes to the topics said before and deserializes messages; the server-actor publish data using MQTT through the server and creates a device actor for each node in the COOJA simulation; the device-actor handle the messages and store the information if it needs and tells to send notification.