Keep your distance Report

Alessandro Dangelo, 10524044
Tancredi Covioli, 10498705

July 21, 2021

1 Project proposal

Design and implement a software prototype for a social distancing application using TinyOS and Node-Red and test it with Cooja. The application is meant to understand and alert you when two people (motes) are close to each other.

- Each mote broadcasts its presence every 500ms with a message containing the ID number.
- When a mote is in the proximity area of another mote and receives 10 consecutive messages from that mote, it triggers an alarm. Such alarm contains the ID number of the two motes. It is shown in Cooja and forwarded toNode-Red via socket.
- Upon the reception of the alert, Node-Red sends a notification through IFTTT to your mobile phone.

2 Assumptions and implementation choices

- **Distance range**: the distance range corresponds to the Transmission range of the mote. It is used the default value in Cooja, since it works on a different order of magnitude with respect to a real social distancing, but still scalable in the order of decimeters. Transmission ranges are equal for each mote.
- **Population size**: the population size is set to 6 (MAX_NODES), since a greater number would have been not helpful for testing purposes. It can be increased by editing the header file.
- **Alarm message**: the alarm message is sent each time a mote receives 10 consecutive messages from the same corresponding mote. This approach may bring to an incoming message flooding, expecially in crowded use cases in which social distancing may not be easily met. It is energy consuming, but it is a simple way to force social distancing between two or more motes.
- **Testing**: the system is tested with Cooja, since it gives the possibility to move motes in real time. Some tests have been made also in TOSSIM, but they are not added to the documentation because they have been easily replicated in Cooja.
- **IFTTT notification**: the alarm message is displayed on the user's smartphone by a notification that appears in the notification menu. Also the Telegram message notification is tested, but the notification results to be less reliable than the classic notification.

3 Mote's overwiew

All motes are equal and they implement the same code:
a mote has its own id, a personal counter and stores the status of the counters received from other motes around.
When a mote starts its radio, the counter is set to 0. Once the message containing its own Id and the value of the counter is boadcasted, then the mote increments the counter.
Upon receipt of a new message, the mote updates its status by checking the sender ID, such that the

corresponding value in the status is updated or reset, depending on the value stored and the new value received.
If one of the counters in the status becomes grater or equal than 10, then the alarm procedure is triggered and a message is sent. This message contains the IDs of the two motes (the one of the neighobring mote and its own) and the time they are not meeting the social distancing rule (as number of consecutive messages).
As soon as the two neighboring motes goes away from each other, the alarm procedure is stopped. The update of the status never stops, even during the alarm procedure.

4 Node-Red flow overview

The flow begins with a TCP input node. This node is listening to a specific port, that corresponds to the port which a Cooja's mote is connected to. Each mote may potentially be conncected to a specific port, but in this case only one mote is set up this way for testing and debugging purposes.
All incoming messages are filtered in the next step, this way only the alarm messages go forward. All these messages are then modified without loosing information in order to make them more verbose.
As last step, a IFTTT service is triggered by sending a URL request to the IFTTT API, containing the verbose message.

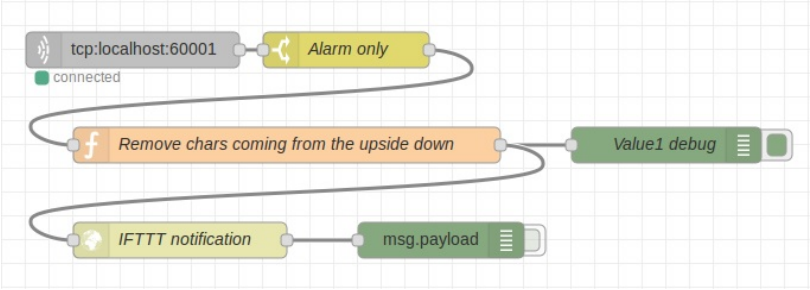

Figure 1: porco

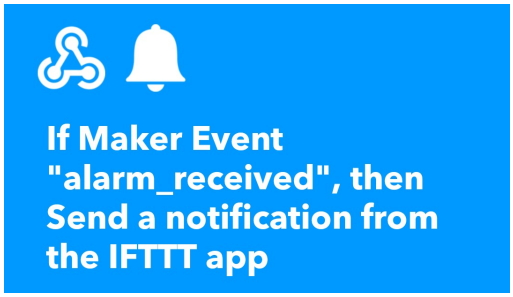

Figure 2: cane

5 IFTTT overview

The IFTTT works this way:
IF the event "alarm_received" is triggered
THEN a notification is sent from the app.

6 Simulation and testing

The project is tested with several topologies and evolutions of them as shown in the following images. Other tests (with more motes) were made, but they were not reported due to lack of specific peculiarities.

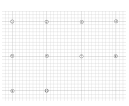

Figure 3: porco

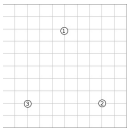

Figure 4: cane

conclusions and others.