LINFO2146

MOBILE AND EMBEDDED COMPUTING

Smart Greenhouses:

Project Report

Group Members:

Antoine TACQ - 22191800
Cyril LAPIERRE - 55922300
Hadrien ALLEGAERT - 07991800

Academic year 2023-2024

1 Introduction

In this project we were asked to design a smart green houses network. To achieve this we are using Z1 motes on contiki cooja and NULLNET (a protocol to send packet over radio waves).

In this report, we will describe our implementation design and the choices we made to create our functional network protocol

2 Message Format in the sensor Network

The packet/message structure is defined as follows:

Field	**Size (bytes)**	**Description**
Version	1	Indicates the version of the packet format
Source Rank	1	Used for reliability (similar to TCP)
Flags	1	8 bits where each bit can represent a specific flag (e.g., SYN, ACK, etc.)
Checksum	1	Used to verify the integrity of the packet
Source IP	8	The IP address of the source (type `linkaddr_t`)
Destination IP	8	The IP address of the destination (type `linkaddr_t`)
Payload	12	Contains the actual data being transmitted, limited to 12 bytes

We chose to use the default 8-byte IP addresses of contiki present as the `linkaddr_t` types. Note that the dhecksum field is not really used in the current version of our protocol. Figure 1 represents the visual structure of our packets.

```
 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
+---------------+---------------------+----------------------+-----------------------+
|    Version    |        Flags        |     Source rank      |       Checksum        |
+---------------+---------------------+----------------------+-----------------------+
|                                                                                    |
|                                  Source linkaddr                                   |
+------------------------------------------------------------------------------------+
|                                                                                    |
|                                Destination linkaddr                                |
+------------------------------------------------------------------------------------+
|                                                                                    |
|                                  Payload 96 bits                                   |
|                                                                                    |
+------------------------------------------------------------------------------------+
```

Figure 1: Structure of the packets sent on the network

2.1 Flags Description

The `flags` field is a single byte where each bit represents a specific flag. The possible flags are defined as follows:

Flag	Value	Description
TCP	128	mobile terminal maintenance messages
MLT	64	multicast from server
ACK	32	Acknowledged / Packet set in was received.
NACK	16	Not Acknowledged /Packet set in was not received.
RLY	8	Relay broadcasted packet (reserved for future use)
DIO	4	Alive
DIS	2	Discover: Packet sent to discover neighbour.
PRT	1	Parent: Packet sent to ask to become the child to a parent
UDP	0	If using UDP, all flags are set to 0 (packet from mote/user)

3 Routing Protocol

The wireless sensor network protocol described here is implemented in C using the Contiki OS. It is designed to facilitate device discovery, communication, and maintenance within a hierarchical network structure. The protocol supports various device types, including light sensors, irrigation systems, light bulbs, and mobile devices. Nodes within the network are assigned specific roles, such as gateway, subgateway, or sensor, each with unique functions and responsibilities.

3.1 Node Initialization

Initialization of the nodes involves setting the device type and configuring them according to their roles. Gateways, for instance, can accept child nodes and do not need a parent. Similarly, subgateways and regular sensor nodes are set up with the necessary configurations to participate effectively in the network.
A gateway has a rank of 2, subgateway has 1 and sensors have 0. This parameters is shared in packets to make the tree correctly.

3.2 Neighbor Protocol

Neighbor discovery and management are crucial for maintaining an updated network map. Nodes broadcast discovery messages to identify neighbors and handle incoming discovery messages to update their neighbor lists. They maintain this list by updating the last heard time of each neighbor and periodically checking if neighbors are still active. This process helps in managing network topology dynamically and ensures that communication paths are always optimized.

3.2.1 Discovery Protocol

In the following subsection the packets with a DIS flag set (or DIS+ACK) are called DIS packet.
When the node start, it broadcast a DIS packet on radio wave using
NULLSTACK_NETWORK.output(NULL). All node in the same sub-network and the gateway will receive it.

When a DIS packet is received by the node X, it will respond in unicast with DIS+ACK to acknowledge he exists too. If the node X doesn't know the packet sender address, it will add it as a new neighbor in its list.

3.2.2 Heartbeats

We decided to send heartbeats, or keep-alive messages, for each node in the network. These messages are packets, with their flag set to DIO and their payload being the type of the device. These messages are broadcasted to every neighbors every 20 seconds.

3.2.3 Neighbor Management

The first part of neighbor management concerns the reception of DIO messages that are received. Such messages states that a node in our neighborhood is active. If for any reason a node receives a DIO message and finds that the source of it is not in its neighbors list, the node will start a neighbors discovery activity where he will broadcast DIS packets. On the other hand, if the source of the message is in the list, the node will update the neighbor last time seen value. Furthermore, the device type of the device will be set thanks to the payload of the DIO message and the signal strength is also updated at this time.

Once every 20 seconds, each node will perform a neighbors lookup. This consists of looping through the list of neighbors and check a node was seen alive in the last 30 seconds (i.e. our node received a DIO message from the other node). If the last time seen is higher than this, the neighbor will be considered dead and removed from the list.

```c
typedef struct mote{
    int rank;                           // GATEWAY, SUBGATEWAY or SENSOR
    linkaddr_t adress;                  // The real address of the mote
    int signal_strenght;                // The last msg received signal
    unsigned long long last_time_heard; // The last time DIO msg received
    linkaddr_t src;                     // The adress where to send message
    int device_type;                    // Sensor type (i.e light bulb)
} mote_t;
```

A neighbor is represented by the mote structure above.

3.3 Network Tree building

Nodes form hierarchical relationships through parent and child attachments.
This process involves sending and receiving parent request (PRT) packets. When a node receives a PRT packet, it attaches a child node, establishing a structured network topology that ensures efficient communication and data flow.

A parent does not acknowledge a PRT request, we assume that the parent received it and add it into our parent. If our neighbor manager system discard the selectionned parent it will simply remains NULL until a new concurrent is discovered through DIO or DIS message.
A parent can be a node with the superior rank of you, but no more and no less. We didn't implement the fact that sensor can be used as relay.
The Tree formation is fully automatised to have a good packet routing.

3.4 Communication Protocols

3.4.1 Unicast

As all joinable neighbors are know by all node in the network, we simply transfer the packet on the neighbor source (*mote.src*) with an address (*mote.adress*) equal to the node you would join.
For example the Sensor S is known through the SubGateway SG by the Gateway G, then if G want to send an unicast packet to S, it will format the packet normally and then send it to SG, which will transfer it to S.

3.4.2 Broadcast

Packets received in a broadcast way will be handle in unicast by the parent of the sender. If a node receives a broadcasted packet, it will add a RLY flag to the packet and then transfer it to its parent and all its childs (excepting the sender one) in an unicast maner. The gateway doesn't transmit broadcasted message it received.

4 Multicasting Implementation

The multicasting implementation is designed to send messages to specific types of devices within a network. It operates using two main functions: one for the gateway, to receive messages from the server and another to send messages to the corresponding devices.

4.1 Server messages

When the server sends messages to the gateway, the gateway will receive it and analyse the payload of the message to determine what kind of action the server wants.

There are 2 cases that the gateway needs to handle :

- Server message starts with "L" : this kind of message states that the server commands to turn on the lights of a specific greenhouse. The gateway will look at the remaining part of the message to fetch the address of the sugbateway that it needs to contact and the duration the lights must stay on. Based on this the gateway will send a packet with a flag MLT to the subgateway, and with a payload corresponding to the light system and the duration asked by the server.

- Server message starts with "I" : For this message the gateway will look at the list of his children. For each child, it will send them a packet with a flag MLT and a payload corresponding to the irrigation system and the duration.

4.2 Reception of packet with flag MLT

When receiving such MLT packet, a node will again check the payload of the packet received from the gateway, using the function (***multicast_ message***), and react accordingly. Based on this payload, the node will either:

- Send a unicast message to all the light bulbs of its subnetwork by simply changing the flag to UDP and the destination address of the packet for each of them.

- Send a unicast message to all the irrigation nodes of theh subnetwork with a packet with flag TCP and the corresponding destination address. The choice of flag TCP here comes from the need to have the irrigation node to respond and send an acknowledgment packet (TCP+ACK) to the server.

5 Github Repository

Here is the link to our Github Repository: Link.

6 Conclusion

This project showcases how a smart greenhouse network can enhance modern farming.

By setting up a structured wireless sensor network and using efficient routing protocols, we ensured unreliable communication between sensors, subgateways, and gateways.
Our multicasting feature sends targeted messages to specific devices, optimizing resource use. Regular maintenance and neighbor discovery keep the network robust and adaptable.

This project gave us difficulties to develop a functional network protocol because of the usage of radio communications between the motes instead of wired communications and the little memory and power z1 motes have.