Inserez un titre

1st Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address

2nd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address

3rd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address

***Abstract*—Machine to Machine (M2M) communication has gained much interest in the recent past and the issues related to energy efficiency are central to all wireless networks, including wireless sensor, for an energy economy, we need to limit emissions so some nodes will make requests to their neighbors. This article will show how to make sure neighbors are woken up during queries in a hybrid network architecture or how to make sure that the nodes that will receive the information will be woken up ?**

***Index Terms*—M2M, hybrid network,hybrid network, wifi, ad-hoc**

I. Introduction

II. Operation

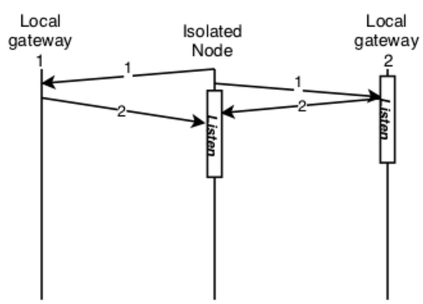

Figure 1. Discover Phase

Phase 1: Discovery Isolated nodes broadcast messages to discover(message 1) a Local Gateway. The end-nodes send messages every (X + random Integer) seconds. If a Gateway receive a discovery message, the Local Gateway will accept (message 2)and communicate to the end-nodes the communication slots.

Phase 2: Registration

The end-node will confirm the pairing to the Local gateway (message 3) . The end-node is registered to the Local gateway. There might be several Local gateway inside the system. However, there must never be multiple registrations on the same one. After the local gateway receive the message 3, gateway send to the isolates node the message 4 to give his acknowledgment. Message 5 is the first message from the isolated node, which contains datas to put an end the pairing phase. When all this stuff is done, the local gateway is able

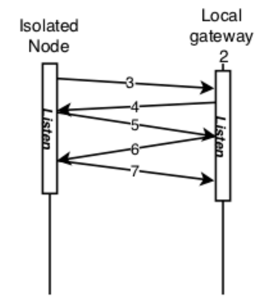

Figure 2. Registering/pairing phase

to send the message 6 : a request of data, then the node will answer his data.

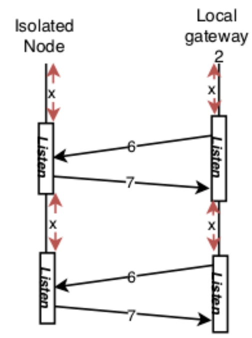

Figure 3. harvest phase

Phase 3: Collection The Local gateway will ask data to the end-nodes. If so, the isolated node sends the data to the Local gateway. This one will then forward it to another Gateway.

Identify applicable funding agency here. If none, delete this.

1) Accuracy: Each message contains information on the next slot, the time of listening on this one, as well as the id of the senders and receivers. If a message is lost or collapsed, it will be re-sent managed by timers.

2) Particular case: When two gateways are in range from the isolated node and there both answer to him with the message 2. The isolated node pick one randomly. To manage multiple isolated node,

III. Algorithm and simulation

A. Algorithm

1) Messages format: The set of system messages are of the form:

```
< message_type , source , destination ,
                 data >
```

There are several messages in the system. It is considered that each message corresponds to a function carrying the name of the message, initializing the type and the source of the message, and taking in parameter the destination and the data.

2) Message from IN to GW:

- The `discover` message
  - message set up for the discovery of a `GW` by an `IN`
  - `destination = udef` : broadcast mode
  - `data = udef` : nothing
- Messages `pair`
  - Pairing message from an IN to a GW.
  - `data = udef` : nothing
- Messages `data_response`
  - Answer from the data request of a GW.

3) les messages GW $\rightarrow IN$: For all the messages coming from the `GW` the data part is structured as follows:

- `answer_frequency` : frequency on which the IN must respond
- `next_slot` : delay by the next listening window
- `next_duration` : fixed time of the next listening window
- `next_frequency` : Frequency of the next listening window
- `data` : data space specific to the exchange

The message is like that :

```
< message_type , source , destination ,
   answer_frequency , next_slot ,
 next_duration , next_frequency , data>
```

The different messages `GW` $rightarrow$ `IN` are thus:

- the messages `candidate`
  - texttt GW reply message after receiving `discover` from `IN`
  - texttt data = udef: no info
- messages `data_request`
  - data request message
  - `data = udef` if only one data available or `data = requested_data` in the case of multiple data

4) Liste des fonctions utilises:

a) Fonction d'mission: `void send(frequency , message)`

b) Fonction de rception: la fonction `listen` coute sur la frquence `frequency` un temps dfinit par `time`. Le prototype de cette fonction est :

```
(message,time) listen(frequency , source ,
       message_type , time_listen)
```

les valeurs des paramtres de cette fonction sont :

- `frequency` : frquence d'coute
- `source` : id de l'metteur du message
  - `source = udef` : coute de tous les nœuds sur la frquence dfinie
- `message_type` : type de message attendu
  - `message_type = udef` : coute de tous les types de messages
- `time_listen` : dure de la fentre de rception
  - `time_listen = udef` : fentre infinie

Valeurs de retour :

- `message` message reu
  - passage du message dans sa totalit
  - `message == udef` : pas de rception respectant les contraintes
- `time` temps restant bas sur `time_listen`
  - `time == udef` : dans le cas de `time_listen = udef`

5) Algorithm 1 - 1:

Algorithm 1 Initialization of communication variables of IN

```
procedure init_var(msg)
    gw ← msg.source
    next_time ← msg.next_slot
    timer ← msg.next_duration
end procedure

procedure flush_var( )
    gw ← udef
    msg ← udef
    next_time ← udef
    timer ← timer_disco
end procedure
```

Algorithm 2 Algorithm IN 1-1

```
while (true) do
    flush_var()
    while (msg = udef) do
        send(freq_listen, discover(udef, udef))
        (msg, t) = listen(udef, candidate, timer + rnd())
    end while
    initVar(msg)
    send(freq_send, pair(gw, udef))

    while (gw! = udef) do
        sleep(next_time)
        (msg, t) = listen(gw, data_request, timer)
        if msg! = udef then
            initVar(msg)
            send(fdate_response(gw, local_data)
        elseflush_var()
        end if
    end while
end while
```

a) Algorithm des `IN`*:*

Algorithm 3 Initialization of communication variables of GW

```
procedure init_var( )
    freq_send → chose()
    timer ← chose()
    freq_listen ← chose()
    freq_next ← chose()
end procedure

procedure flush_var( )
    timer ← timer_disco
    freq_listen ← freq_disco
    freq_send ← freq_disco
    in ← udef
end procedure
```

Algorithm 4 Algorithm gw 1-1

```
LoRaWAN_join()
flush_var()
while (true) do
    if (in == udef) then
        (msg, t) = listen(, udef, discover, timer)
    end if
    if (msg! = udef) then
        in ← msg.source
        init_var()
        send(candidate(in, slot, duration, udef)
        (msg, t) = listen(, in, pair, timer)
        if (msg == udef) then
            flush_var()
        end if
    end if
    if (in! = udef) then
        init_var()
        send(data_request(in, slot, durationt, udef)
        (msg, t) = listen(in, data_response, timer)
        if (msg! = udef) then
            send_data(id + " : " + local_data + ";" + in + " : " + msg.data)
        else
            send_data(id + " : " + local_data + ";" + in + " : " + udef)
            flush_var()
        end if
    end if
end while
```

6) Algorithm of `GW`*:*

B. Simulation

IV. Conclusion

[6], [5], [2], [3], [4], [1]

Acknowledgment

References

[1] E. C. Haddad and J. C. Gregoire. *Implementation issues for the deployment of a WMN with a hybrid fixed/cellular backhaul network in emergency situations*, pages 525–529. 2009 1st International Conference on Wireless Communication, Vehicular Technology, Information Theory and Aerospace & Electronic Systems Technology. 2009.

[2] Ming Ann Ng and K. L. A. Yau. *An energy-efficient Hybrid Wireless Mesh Protocol (HWMP) for IEEE 802.11s mesh networks*, pages 17–21. 2013 IEEE International Conference on Control System, Computing and Engineering. 2013.

[3] N. G. Palan, B. V. Barbadekar, and S. Patil. *Low energy adaptive clustering hierarchy (LEACH) protocol: A retrospective analysis*, pages 1–12. 2017 International Conference on Inventive Systems and Control (ICISC). 2017.

[4] G. Sharma, N. B. Shroff, and R. R. Mazumdar. *Hybrid sensor and mesh networks: paradigms for fair and energy efficient communication*, pages 83–92. 2006 2nd IEEE Workshop on Wireless Mesh Networks. 2006.

[5] S. Singh, K. L. Huang, and B. S. P. Lin. *An energy-efficient scheme for WiFi-capable M2M devices in hybrid LTE network*, pages 126–130. 2012 IEEE International Conference on Advanced Networks and Telecommunciations Systems (ANTS). 2012.

[6] K. Tanaka, M. Murase, and K. Naito. *Prototype implementation of BLE based automated data collection scheme in agricultural measurement system*, pages 1–2. 2018 15th IEEE Annual Consumer Communications & Networking Conference (CCNC). 2018.