An ns-3 Implementation of a Battery-less Node for Energy-harvesting Internet of Things

Martina Capuzzo
Department of Information Engineering
University of Padova
Padova, Italy
capuzzom@dei.unipd.it

Carmen Delgado
i2CAT Foundation
Barcelona, Spain
carmen.delgado@i2cat.net

Jeroen Famaey
IDLab,
University of Antwerp - imec
Belgium
jeroen.famaey@uantwerpen.be

Andrea Zanella
Department of Information Engineering
University of Padova
Padova, Italy
zanella@dei.unipd.it

ABSTRACT

In the Internet of Things (IoT), thousands of devices can be deployed to acquire data from the environment and provide service to several applications in different fields. In many cases, it is desirable that devices are self-sustainable in terms of energy. Therefore, the research community is exploring the possibility of employing battery-less devices, where the energy is derived solely from external and/or environmental sources, such as solar panels. In this work, we propose an ns-3 model of a (super) capacitor, which can be used as the storage of the harvested energy in a battery-less IoT device, and add the support for the intermittent behavior of devices, turning off/on according to their energy level. To exemplify the use of the model, we apply it to a LoRaWAN node, and compare the simulation outcomes with results in the literature obtained with mathematical analysis, confirming the accuracy of the implementation. Then, we show the importance of analyzing the interaction between energy availability and communication performance, paving the way for more accurate and realistic simulations in the field. The implemented code is made available as open source.

CCS CONCEPTS

• **Networks** → **Network simulations**; • **Sensor Networks**;

KEYWORDS

Network simulations, ns-3, Internet of Things, Battery-less device, Capacitor, Energy Harvesting

ACM Reference Format:
Martina Capuzzo, Carmen Delgado, Jeroen Famaey, and Andrea Zanella. 2021. An ns-3 Implementation of a Battery-less Node for Energy-harvesting Internet of Things. In *2021 Workshop on ns-3 (WNS3 2021), June 23–24, 2021, Virtual Event, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3460797.3460805

WNS3 2021, June 23–24, 2021, Virtual Event, USA

ACM ISBN 978-1-4503-9034-7/21/06…$15.00
https://doi.org/10.1145/3460797.3460805

1 INTRODUCTION

In the last years, the Internet of Things (IoT) paradigm has been applied in many contexts, such as Smart Cities, healthcare, and industrial and agricultural scenarios. In such application scenarios, the network infrastructure must be able to support a large number of devices, generally transmitting at low bit rates. The communication technologies, in turn, need to serve wide networks with a large number of devices. Therefore, in the last years, Low Power Wide Area Network (LPWAN) technologies have been proposed, among which LoRaWAN and Sigfox have reached a large popularity.

In many cases, devices are battery-powered and, although the communication technology is designed to limit the energy consumption, adverse network conditions, channel impairments or unwise settings can negatively affect the device's lifetime. Unfortunately, battery replacing is costly from an economic and environmental perspective, pushing for a migration to green solutions. One possibility is to use energy harvesting techniques, which derive energy from renewable sources (e.g., solar power, thermal/wind energy), and can stored it in (super) capacitors integrated in battery-less devices. Nevertheless, the inconstant behavior of the harvested energy impacts on the device's capabilities, including communication.

These aspects have started gaining interest in the last few years, but have not been deeply evaluated yet. Therefore, works that deal with IoT networks with battery-less devices and energy harvesting [4–6, 9], mainly address theoretical analysis (i.e., mathematical modeling) or empirical evaluations. Instead, the use of simulations can provide a precious help in the evaluation of the interplay between the network state, the system configuration and the device's energy capabilities. For example, the communication could benefit from the robustness provided by message repetitions or the use of a higher transmission power, but this will impact the energy autonomy of the device; conversely, low energy levels may prevent the correct transmission of some packets. These aspects are further complicated when considering the variability of the energy source in nodes with harvested power and the interference of many communicating devices.

In this paper, we propose an ns-3 implementation of a battery-less node with a (super) capacitor coupled with an energy harvester, which enables the performance evaluation of IoT networks with battery-less devices. This is built as an extension of the native ns-3 energy model, and can be easily integrated with existing ns-3 modules, supporting the intermittent behavior of devices that can turn off or on according to their energy level. Also, a supplementary class makes it possible to import values for the harvested power from external files, which can be obtained from real measurements. The implemented code is available at [1]. We validate the proposed framework by leveraging the `lorawan` ns-3 module, and describe some results obtained when considering a LoRaWAN battery-less device.

The remainder of this paper is structured as follows. Section 2 describes how the battery-less device is modeled, while the specific ns-3 implementation and integration with existing modules are discussed in Section 3. In Section 4 we first briefly describe the LoRaWAN technology that we use in our simulations to validate the capacitor's implementation, and then discuss the obtained results. Finally, Section 5 reports the conclusions and possible extensions of this work.

2 BATTERY-LESS IOT DEVICES WITH ENERGY HARVESTING

To model a battery-less IoT device, we consider the approach used in [4]. The device consists of several components: a Micro Controller Unit (MCU), a radio unit, some peripherals (e.g., for sensing purposes), a capacitor to store energy, and a harvester mechanism to recharge the capacitor. The overall device can be modeled as an equivalent electrical circuit with three parts: (i) the harvester, (ii) the capacitor, and (iii) the load, as represented in Figure 1, and better described next.

(i) *The harvester:* it is the only energy source in the system. The harvester is modeled as an ideal constant voltage source (denoted by E) with a series resistance ($r_i(t)$) that determines the maximum power that can be produced by the harvester, which is given by

$$P_{harvester}(t) = \frac{E^2}{r_i(t)}. \quad (1)$$

In general, the resistance $r_i(t)$ can change in time, according to the fluctuations on the energy harvesting process. By coupling the harvester with a voltage regulator, however, the output voltage E can be stabilized. Our model makes it possible to either generate the harvested power values as independent random samples taken from a given distribution, or to read them from a pre-loaded trace file.

(ii) *The load:* it models all the components of the system that consume energy. According to the activity performed by each component, it is possible to define different *states* of the load, which are characterized by a specific power consumption. For each state, we can therefore define a load resistance $R_L(s)$, which is computed considering the total current $I_{load}(s)$ absorbed by the load in the specific state s:

$$R_L(s) = \frac{E}{I_{load}(s)}. \quad (2)$$

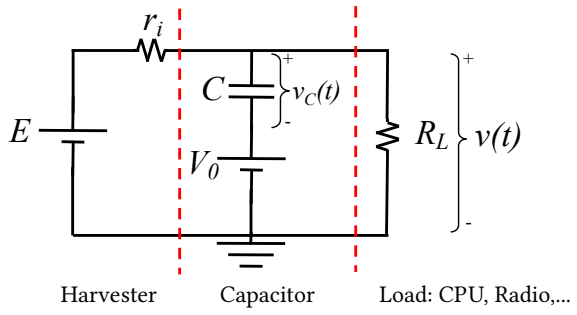


Figure 1: Electrical Circuit Model of a Battery-less IoT Device [4]

(iii) *The capacitor:* it stores the energy generated by the harvester and releases it to the load when required. The behavior of the system can be represented by a series of intervals corresponding to different events/activities (e.g., MCU active and radio transmitting), corresponding to the different states of the load. For each state s, the voltage of the capacitor can be represented by $(V_0, v_C(t))$, where V_0 is the capacitor voltage when entering the state, and $v_C(t)$ is the voltage of the capacitor after t seconds spent in state s. As depicted in Figure 1, V_0 is included in the circuit as an ideal voltage source, while $v_C(t)$ is the voltage over time of an ideal capacitor.
The voltage provided by the capacitor to the load after t seconds in state s can thus be computed as

$$v(t,s) = E\frac{R_{eq}(s,t)}{r_i(t)}\left(1 - e^{-\frac{t}{R_{eq}(s,t)C}}\right) + V_0 e^{-\frac{t}{R_{eq}(s,t)C}}, \quad (3)$$

where C is the capacitance of the capacitor [in Farads], and

$$R_{eq}(s,t) = \frac{R_L(s) r_i(t)}{R_L(s) + r_i(t)}. \quad (4)$$

To model real devices, we consider that they may switch off at anytime because of an energy level too low to continue their functioning. Therefore, we define two voltage thresholds for $v(t,s)$: V_{th_low}, below which the device switches off, and V_{th_high}, above which the device goes back to active state, thanks to the harvested energy. As an example, Figure 2 shows the voltage level of a device over time, highlighting the on/off phases with different background colors. The figure has been obtained by setting C = 5 mF, $P_{harvester}$ = 0.001 W, packet generation period of 60 s, and LoRa DR 3 (better defined in Section 4.1). The voltage thresholds determining the switch to on/off states have been set to $V_{th\_low} = 1.8\ V$ and $V_{th\_high} = 3\ V$. Markers correspond to the beginning of the state, which will be better explained in Section 4.

3 CODE IMPLEMENTATION IN NS-3

The code contribution is presented in three parts, described in this section. In Figure 3, a scheme depicts the relation between the different components. We point out that the presented framework can be extended and/or tuned to consider also different behaviors of the device (e.g., states), or values for current consumption. Indeed, the application considered in this work is based on the `LoRaRadioEnergyModel` class and inherits its specificities.

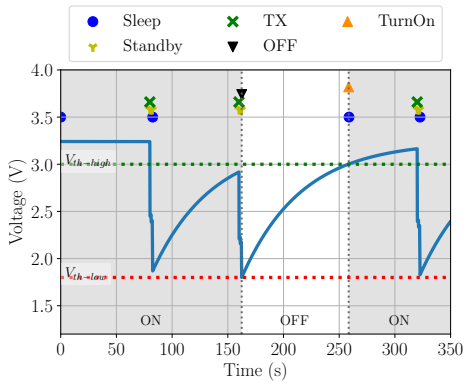


Figure 2: Example of the Device's Voltage when it enters Different Phases

3.1 Capacitor

The ns-3 class implementing the storage of the energy in a capacitor is called `CapacitorEnergySource`, as it extends the `EnergySource` class available in ns-3.[1] Thus, it is used as done for the classes implementing the Lithium Ion Battery or the non-linear battery model. As such, it can be easily connected to energy harvester components (`EnergyHarvester`) and to the class modeling the energy consumption behavior of the device (`DeviceEnergyModel`). The capacitor's features can be set using the class attributes and methods; the most relevant attributes are reported in Table 1.

The value of the energy stored in the capacitor and the corresponding voltage can be updated periodically by calling the appropriate function: `UpdateEnergySource`. This function computes $v(s,t)$ as for Eq. (3) according to the current state s, with V_0 being the voltage computed at its previous call. It is recommended to call the function also before switching the device to a new state, in order to keep the voltage up-to-date with the correct value of $R_L(s)$ and guarantee that the device's energy is not depleted, preventing the correct switching to the new state.

Besides the common "setters" and "getters" methods, and the auxiliary functions to compute the voltage level, the class also provides the following:

- `IsDepleted`, a function returning `true` if the current voltage value is below V_{th_low};
- `ComputeLoadEnergyConsumption`, a function computing the energy dissipated only by the load where a given current flows for a given time interval, and the initial voltage level of the capacitor is given as input;
- `TrackVoltage`, a function producing a file with the value of the voltage. The use of a trace source connected to the variable indicating the remaining voltage may instead have incorrect behavior because of the small difference between consecutive updates, which may not be detected by ns-3 native implementation;
- Traced values, to inspect their evolution through the simulation (e.g., the voltage level).

[1] https://www.nsnam.org/doxygen/classns3_1_1_energy_source.html

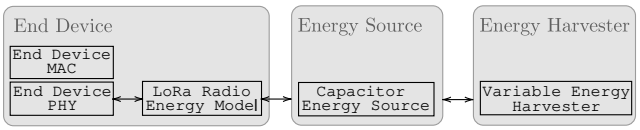


Figure 3: Diagram illustrating the Relation between Different Code Components

3.2 Variable Energy Harvester

The class `VariableEnergyHarvester` extends the `EnergyHarvester` class provided by ns-3, taking harvested power values from a .csv file given as input. The harvested power is periodically updated, and the energy source object(s) (e.g., the capacitor) connected to the harvester are updated accordingly.

The implementation of the function reading the .csv file is specific to the file we considered as input to our scripts. The code can be easily modified to consider data saved in a different format, since it only needs a pair (timestamp, $P_{harvester}$). Note also that, when running simulations, the length of the trace provided to the energy harvester should be as long as the simulated time. In case of mismatch, an error is raised during the simulation.

3.3 Integration with an Existing Module: `lorawan`

To validate the capacitor model, we employ the `lorawan` ns-3 module [2, 8]. In particular, we extended the class `LoraRadioEnergyModel`, a child class of the `DeviceEnergyModel`, to work with the capacitor's implementation, and added some states and variables to improve its compliance to real devices. Furthermore, the module's classes representing the Medium Access Control (MAC) and Physical (PHY) layers of the devices have been modified to update and verify the energy level before switching to a new state. The on/off behavior is implemented as follows: if the stored voltage is below V_{th_low}, the capacitor enters in the "depleted" state, the `LoraRadioEnergyModel` is notified, and the device enters into the Off state, interrupting ongoing transmissions, if any. In this case, the transmitted signal is considered only as interference by all the receivers. Conversely, when enough energy is harvested, and the voltage is above V_{th_high}, the capacitor switches from the "depleted" to the "recharged" state, trigging some events from the `LoraRadioEnergyModel` class (usually, the switch to the Sleep state) and enabling again packet transmission.

4 APPLICATION AND VALIDATION

In this section, we consider the LoRaWAN technology to validate the `CapacitorEnergySource` class presented in Section 3. In the following, we first introduce the LoRaWAN technology and the states that characterize the device operations. Then, we show preliminary results where the proposed framework is applied to LoRaWAN nodes, and validate the approach with a comparison with state-of-the-art approaches. Then, we expand the analysis to observe mutual relations between capacitor's properties and the configuration of the technology, and how they impact on the success of the communication.

Table 1: Relevant Attributes of the `CapacitorEnergySource` Class

Attribute	Description
Capacitance	Capacitance [F]
CapacitorEnergySourceInitialVoltage	Initial voltage of the capacitor [V]
CapacitorMaxSupplyVoltage	Maximum supply voltage for the capacitor energy source [V]
CapacitorLowVoltageThreshold	V_{th_low}, as fraction of the maximum supply voltage
CapacitorHighVoltageThreshold	V_{th_high}, as fraction of the maximum supply voltage
PeriodicVoltageUpdateInterval	Time interval between periodic voltage updates [s]

4.1 LoRaWAN

LoRaWAN [7] leverages the proprietary LoRa modulation, based on the Chirp Spread Spectrum (CSS) technique. The robustness of the modulation can be adjusted by tuning the Spreading Factor (SF) parameter, which takes integer values from 7 to 12. The SF is directly connected to the data rate: higher SFs allow for more robust transmitted signals and longer coverage ranges, but at the price of a lower data rate and, thus, longer transmission times.

LoRaWAN networks have a star-of-stars topology with three kinds of devices:

- End Devices (EDs) are peripheral nodes, usually sensors or actuators, that communicate using the LoRa modulation;
- Gateways (GWs) are relay nodes that collect messages coming from the EDs through the LoRa interface, and forward them to the Network Server using a reliable IP connection, and vice-versa;
- the Network Server (NS) acts as a central network controller that manages the communication with the EDs through the GWs.

Figure 4 illustrates an example of LoRaWAN network, where dotted lines represent LoRa links, while IP connections are shown as solid lines.

The LoRaWAN specifications define three classes of EDs, which differ in terms of energy saving capabilities and reception availability. In this work, we focus on Class A devices, which stay in sleep mode most of the time in order to minimize the energy consumption, transmit a packet whenever required by the application layer, and open at most two reception windows after each transmission. If a downlink (DL) message is received in the first receive window (RX1), the second receive window (RX2) is not opened.

Moreover, the specifications define two types of messages: confirmed and unconfirmed. For the first case, an Acknowledgment (ACK) packet is expected by the EDs, in either of the two reception windows opened after the transmission. EDs have m transmission opportunities: if the ACK is not received, the device can re-transmit the packet up to $m - 1$ times. Unconfirmed messages, instead, do not require any ACK.

LoRaWAN operates in the unlicensed Industrial, Scientific, and Medical (ISM) frequency bands. The relevant regulations define frequency bands, power and Duty Cycle (DC) restrictions to be applied. In the European region, LoRaWAN [7] defines the use of three 125 kHz wide channels, centered at 868.1, 868.3, 868.5 MHz, which are shared between uplink (UL) and DL transmissions, and must collectively respect a 1% DC constraint [3]. As per the specification [7], a fourth 125 kHz channel centered at 869.525 MHz is used for DL communication only, and is subject to a DC constraint of 10% [3]. Once fixed the channel bandwidth, there is a one-to-one correspondence between SFs and data rates: SF 7 corresponds to Data Rate (DR) 5, SF 8 to DR 4, and so on, till the most robust SF, which corresponds to DR 0.

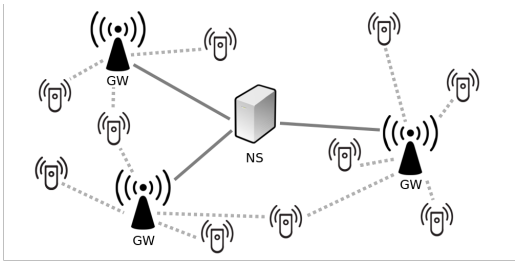


Figure 4: LoRaWAN Infrastructure

For Class A devices, the standard requires the RX1 to be opened in the same frequency channel and with the same SF used for the UL communication. The RX2, instead, is always opened on the dedicated 869.525 MHz channel and with SF 12, in order to maximize the robustness of the communication.

In the following discussion, we assume only UL traffic. Furthermore, to measure the quality of the communication, we will indicate as *UL cycle* the interval between the beginning of the ED's packet transmission till the moment when it is successfully delivered to the GW, and *UL + DL cycle* the interval from the moment when the ED starts the UL packet transmission till the successful reception of the corresponding ACK.

4.1.1 LoRaWAN Device States. As discussed in Section 3, the different states the device goes through are important to determine the energy consumption of the device. According to the LoRaWAN protocol described above, the following states are identified. The respective current consumption is reported in Table 2, considering the load composed by the MCU and radio units. The contribution of the MCU is considered only in terms of current consumption, and corresponds to 11μA in active state, 5.5 μA in standby state.

- *Off*(*): the ED's radio is switched off, and the MCU is in standby, maintaining only the clock synchronization;
- *TurnOn*(*): the device wakes up from the Off state, with a certain energy expenditure. In our implementation we consider a current consumption of 15 mA and a state duration of 300 ms, but the values can be tuned according to specific devices considered;

Table 2: Current Consumption in the Different States [4]

State	MCU	Radio current	Total current
Off	Standby	0	5.5 μA
Turn On	Active	-	15 mA
Sleep	Active	1 μA	5.6 μA
Tx	Active	28 mA	28.011 mA
Idle	Standby	1.5 μA	7 μA
Standby	Standby	10.5 mA	10.5055 mA
Rx	Active	11 mA	11.011 mA

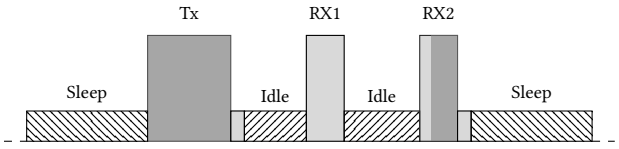


Figure 5: Example of ED's State Transitions

- *Sleep*: the radio is in sleep state, saving power, without performing any activity, and the MCU is in standby mode;
- *Tx*: the device is transmitting data;
- *Idle*: "waiting" period before the opening of the receive window;
- *Standby*: listening to idle channel when the receive windows are open. Also, the standard defines the ED to switch to Standby (for a very short time) after transmission and reception operations;
- *Rx*: the device is receiving data.

The Off and Turn On states, marked with (*), are not part of the standard, but are present in real devices. In Figure 5 a diagram depicts the operational states of the device. In this case, the device wakes up from the Sleep state to transmit data requiring an ACK, which is successfully received in RX2, after a short time spent in Standby mode. The light-grey-colored regions represent the Standby phase. Note that the device will be able to complete an UL cycle if the voltage level is enough to complete the transmission procedure (no impairments from the channel/network are considered), while an UL + DL cycle is successfully executed if the energy stored in the device is enough to complete all the operations from transmission to reception, which can happen in either RX1 or RX2, including also the intermediate Idle and Standby states.

4.2 Results

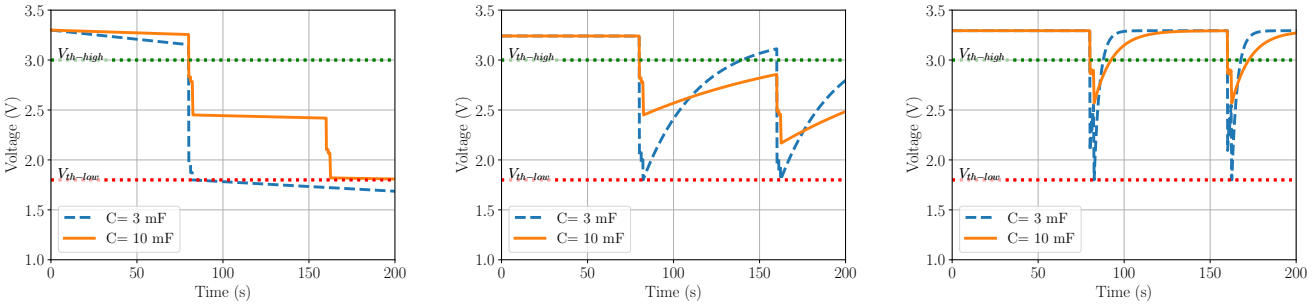
To test our ns-3 implementation, we consider a simple LoRaWAN network composed of a NS, a GW and a single ED provided with a capacitor with variable size. Furthermore, we consider V_{th_low}=1.8 V and V_{th_high}=3 V, and different values for $P_{harvester}$, from 0 W to 0.01 W, which are constant in time. The device periodically generates packets with a payload length of 10 bytes, and can use either unconfirmed or confirmed messages, with m=1. The smart option preventing a packet transmission if the energy cost is not supported by the device is used. The LoRa settings are as considered in [4].

Figure 2 shows the capacitor's voltage together with the states of the device.[2] The initial voltage of the capacitor is 3.3 V, which is almost constant for the first part of the simulation, when the device is in Sleep mode. At t =80 s, the ED performs the first transmission, entering the Tx state, which causes a first drop in the capacitor's voltage, bringing it to 2.44 V. The traffic type is unconfirmed, therefore, no DL transmission is expected. Nonetheless, as dictated by the standard, the two reception windows are opened, which cause the voltage drop around time 81 s. Note that, as it can be seen also in Figure 6a, the energy drop during RX1 is smaller than that experienced during RX2, whose duration is 2^6 = 64 times longer than that of RX1. In the simulated scenario, the harvesting rate is EH=0.001 W, which allows the capacitor to recharge during the sleeping period from 81 s to 160 s, reaching almost 3 V. Note that this is not visible during the initial sleeping period, because the voltage level is very close to the maximum voltage supported by the device. At 160 s a second transmission occurs, followed by the opening of the two reception windows. In this case, the voltage at the beginning of the cycle was lower than in the previous case, and the long duration of RX2 makes the voltage drops below V_{th_low}, so that the ED enters the Off state. Then, a recharging phase follows, and when $v(t, s)$ reached V_{th_high} =3 V, the ED starts the TurnOn phase, entering into the Sleep state. This enables it to successfully perform the next transmission, at time t=240 s.

Figure 6 shows the voltage level for different capacitor sizes and $P_{harvester}$ values, which have a determinant impact on the communication capabilities. As a benchmark, in Figure 6a we report the case with no harvesting. From this, we can appreciate the impact of different capacitors' sizes: a smaller capacitor discharges much faster than a bigger one, and can rapidly make the device switch off. However, when using smaller capacitors, also the recharging phases are faster, as it can be observed in Figures 6b and 6c: this behavior may cause the device to swap between On and Off states, preventing proper communication. Conversely, a larger capacitance will charge and discharge more slowly, allowing better communication performance (in terms of successful transmissions) also in the case of lower harvesting rates, since it will reduce the number of times the node enters in Off state and, consequently, the energy cost for taking it back to the active state. The downside is that, whenever the capacitor voltage drops below the lower threshold, it will take a longer time to accumulate enough energy to pass again the high threshold and bring the device back to an operational state. Therefore, while the V_{th_low} is typically hardware-dependant, a proper tuning of the V_{th_high} threshold should also consider the capacitor's size.

The minimum capacity size that makes it possible to complete an UL (resp. UL + DL) cycle for different UL packet sizes, harvesting rates and DRs is presented in Figure 7, and compared with mathematical results obtained from the model proposed in [4], which are represented with lines, while markers represent simulation outcomes. The results of the model had been confirmed by comparison with real devices in [9]. The DL packet size is fixed to 39 bytes (at APP layer). Also, the initial capacitor voltage is computed taking into account the current in the Off state. Since there is a single ED in the network, the GW can always use RX1, sparing the device

[2] Markers signaling when entering in Standby and Idle states are not plotted for clarity.

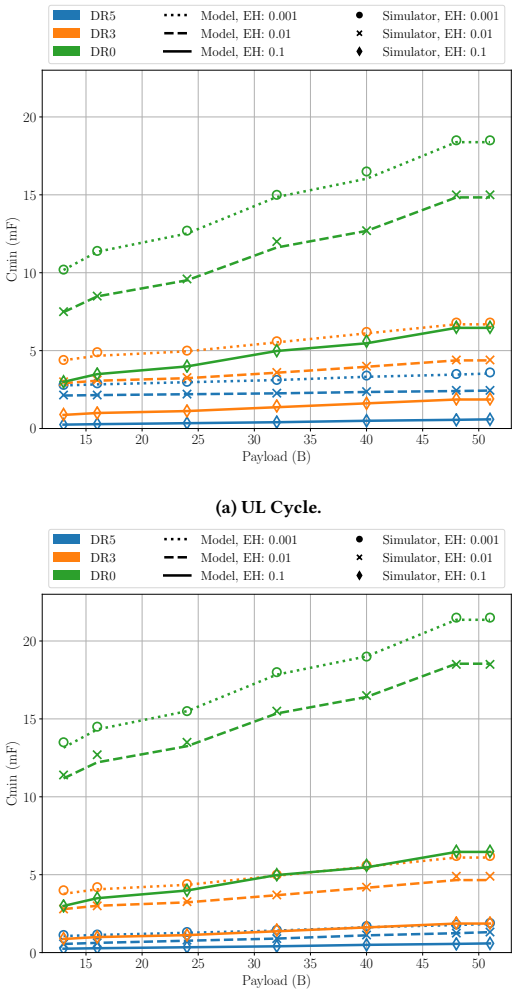


(a) $P_{harvester}$ **= 0 W.** **(b)** $P_{harvester}$ **= 0.001 W.** **(c)** $P_{harvester}$ **= 0.01 W.**

Figure 6: Voltage for different values of $P_{harvester}$ and capacitor's size.

the energy consumption due to additional states. In both plots, we can observe that the minimum required capacitance increases for bigger payloads of the UL packet, as expected. Moreover, the lower the DR, the larger the required capacitance, because of the longer transmission time. For lower harvesting rates, a larger capacitor should be employed to successfully complete a cycle, as discussed previously. Similar trends can be observed for the minimum capacitance needed to accomplish an UL + DL cycle: in this case, the values are higher than for the UL cycle only, since additional actions must be performed by the device, including the reception of a DL packet. From these plots we can finally observe that there is a strong agreement between model and simulation results, with the small discrepancy between the two only due to the quantized step used in the simulator.

A final batch of simulations was run to reproduce a realistic scenario where an IoT application periodically generates packets of fixed size. In particular, we evaluated the success probability (in terms of delivered packets) when varying the capacitor size, for an application sending confirmed/unconfirmed traffic for 6 hours, with different packet generation periods. Note that the success probability is computed as the ratio between the number of delivered packets and the total number of packets generated at the application level. Figures 8, 9 show the results for different values of the packet generation period and harvesting rate. Also in this case, we can observe a strong dependence on the DR: while lower DR values improve the transmission robustness to possible channel impairments, they are more costly in terms of energy, strongly affecting the number of packets that are successfully received by the GW. This can be mitigated by storing more energy, as happens for bigger capacitors charged with higher harvested power (Figure 8b). Instead, using higher DR values require smaller capacitors, in the order of a few mF. Furthermore, transmitting packets more sporadically leaves enough time to recharge the capacitor, obtaining a higher success probability for a given capacitor's size. For example, increasing the interval between consecutive packet transmissions from 60 s to 300 s makes it possible to halve the minimum capacitance when using DR 3 and $P_{harvester}$ = 0.001 mW (Figure 8a). Figure 9 reports similar results for the probability of also receiving the ACK: in this case, similar considerations on the relation between minimum capacity, DR and $P_{harvester}$ values can be drawn. However, it is

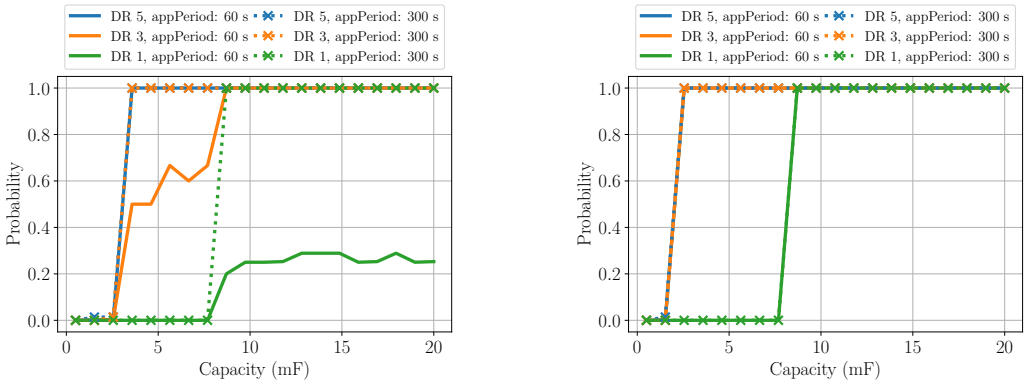


(a) UL Cycle.

(b) UL and DL Cycle.

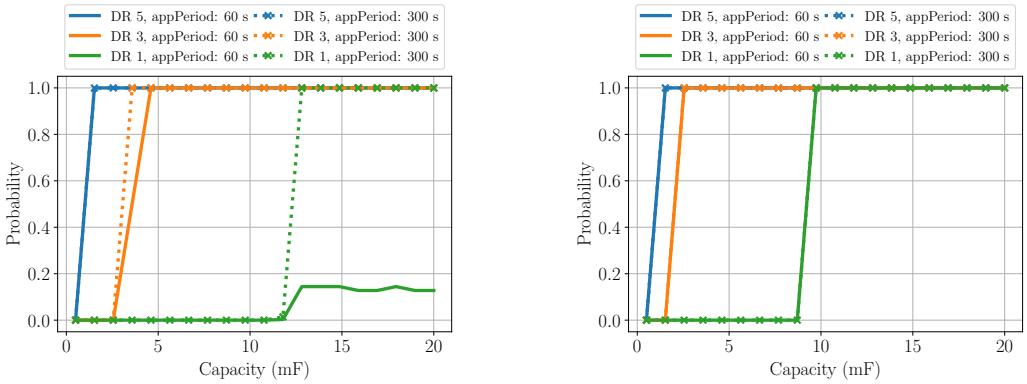
Figure 7: Minimum Capacity to Complete a Cycle compared to Mathematical Results, computed as in [4]

interesting to note that, for low values of $P_{harvester}$ (Figure 9a), the capacitor's size that maximizes the success probability is lower



(a) $P_{harvester}$ **= 0.001 mW** **(b)** $P_{harvester}$ **= 0.01 mW**

Figure 8: Success Probability for Unconfirmed Traffic



(a) $P_{harvester}$ **= 0.001 mW** **(b)** $P_{harvester}$ **= 0.01 mW**

Figure 9: Success Probability for Confirmed Traffic

in the case of confirmed traffic than when using unconfirmed traffic, despite the reception of DL packets occurs. This confirms that using an ACK (with no payload) to prevent the opening of RX2 brings some benefit on the ED's energy consumption and communication performance, specially when the harvested power is low. These aspect could be taken into account for a proper network configuration that targets energy efficiency: using shorter RX2 by employing higher DRs, or even preventing their use, could have a significant effect on the device's energy performance.

5 CONCLUSIONS

In this paper, we propose an implementation of a (super-)capacitor that considers the presence of a generic harvesting source to power battery-less devices. After presenting the considered model, we describe the code implementation and validate it by simulating a simple LoRaWAN network with a single device. The correctness of the implementation was proved by the comparison of some outcomes with those obtained in previous theoretical works. From the results obtained, it is apparent the impact of the harvested power and capacitor size on the communication performance, as well as the effect of the technology settings on the energy requirements. The possibility of including the proposed model in the simulation ecosystem makes it possible to further study how the harvesting approach influences the communication performance and, at the same time, it allows for the performance evaluation of a complete IoT network where many energy-constrained devices compete for the same resources. Future work will explore smarter battery-less devices, that take the energy level into account before performing a transmission, so as to avoid depletion before the operation is completed. Also, we plan to study the optimization of technology and network parameters in such energy-constrained scenarios. The ns-3 code of the capacitor implementation is available at [1].

ACKNOWLEDGMENTS

A special thanks goes to Davide Magrin for his useful suggestions.
Part of this research was funded MIUR (Italian Ministry for Education and Research) under the initiative "Departments of Excellence" (Law 232/2016), by the Flemish FWO SBO S001521N IoBaLeT (Sustainable Internet of Battery-Less Things) and the University of Antwerp IOF funded COMBAT (Time-Sensitive Computing on Battery-Less IoT Devices) projects, and by the CERCA program, by the Generalitat de Catalunya.

REFERENCES

[1] Martina Capuzzo. 2021. capacitor-ns3. https://github.com/signetlabdei/capacitor-ns3.

[2] Martina Capuzzo, Davide Magrin, and Andrea Zanella. 2018. Confirmed Traffic in LoRaWAN: Pitfalls and Countermeasures. In *Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net)*. Capri, Italy.

[3] CEPT. 2019. *ERC Recommendation 70-03 - Relating to the use of Short Range Devices (SRD)*. Technical Report. CEPT ECC.

[4] Carmen Delgado, José María Sanz, Chris Blondia Chris, and Jeroen Famaey. 2020. Battery-Less LoRaWAN Communications using Energy Harvesting: Modeling and Characterization. *IEEE Internet of Things Journal* (Aug 2020).

[5] Francesco Fraternali, Bharathan Balaji, Yuvraj Agarwal, Luca Benini, and Rajesh Gupta. 2018. Pible: Battery-free Mote for Perpetual Indoor BLE Applications. In *Proceedings of the 5th Conference on Systems for Built Environments*. Shenzhen, China, 168–171.

[6] Elvina Gindullina, Leonardo Badia, and Xavier Vilajosana. 2020. Energy Modeling and Adaptive Sampling Algorithms for Energy-harvesting Powered Nodes with Sampling Rate Limitations. *Transactions on Emerging Telecommunications Technologies* 31, 3 (2020), e3754.

[7] LoRa Alliance. 2017. LoRaWAN 1.1 Specification. (Oct 2017).

[8] Davide Magrin, Marco Centenaro, and Lorenzo Vangelista. 2017. Performance Evaluation of LoRa Networks in a Smart City Scenario. In *IEEE International Conference on communications (ICC)*. IEEE, Paris, France, 1–7.

[9] Adnan Sabovic, Carmen Delgado, Dragan Subotic, Bart Jooris, Eli De Poorter, and Jeroen Famaey. 2020. Energy-Aware Sensing on Battery-Less LoRaWAN Devices with Energy Harvesting. *Electronics* 9, 6 (May 2020), 904.