Energy consumption and execution time characterization for the SensorTag IoT platform

Tiago K. C. Shibata, Roberto M. de Azevedo, Bruno Albertini, Cíntia B. Margi

Abstract— **In the context of the Internet of Things, Wireless Sensor Networks are an important component, since they enable sensing, communication and sensor data fusion. Often times these devices are energy constrained and battery powered, and applications require security mechanisms to be used. In order to develop such applications, it is important to evaluate and understand the execution time and energy consumption of common sensing, communication and cryptography mechanisms. We evaluate such tasks on the Texas Instruments CC2650 SensorTag platform, a new device with good potential for IoT applications. Evaluation metrics are execution time and energy consumption.**

Keywords— **Wireless Sensor Networks, Cryptography, Benchmarks.**

I. Introduction

Wireless Sensor Networks (WSN) have been used to support several different applications, mainly related to monitoring, detection and tracking. WSN are networks of low-cost, low energy (mostly battery-operated) embedded devices, focused on sensing and communicating, usually through a multihop ad hoc network [7]. They enable the execution of distributed sensing tasks, which are crucial in environmental sensing, medical monitoring and automation. Due to their limited energy resources, nodes usually stay in an low power state most of the time, waking up on demand to perform sensing/-communication tasks and going back to a low power state.

There is not a unique definition for the Internet of Things (IoT), but most of literature agrees that it is composed of embedded devices capable of sensing/actuation, communication and processing [27]. IoT is deeply related to WSN. By giving communication capability to physical objects and devices, things become part of the Internet. In IoT scenarios, the goal is to make smarter decisions based on sensed data, such as home tasks automation, enabling remote control of devices and smart traffic.

Security services are often not properly addressed in IoT [4], [16]. Since nodes have limited power and processing, the overhead due to processing and extra bytes for communication are often considered enough reason to avoid security mechanisms. However, given the amount of connected devices and personal data available, security services are becoming a requirement in IoT. Furthermore, malicious nodes inserted to generate false data might lead to wrong actions with unwanted consequences in critical scenarios. Thus, efficient asymmetric key agreement, symmetric ciphers and message authentication code (MAC) algorithms are needed for IoT.

Tiago K. C. Shibata, Roberto M. de Azevedo, Bruno Albertini, Cíntia B. Margi. Universidade de São Paulo, São Paulo-SP, Brazil, E-mails: {tiago.shibata, roberto.azevedo, balbertini, cintia}@usp.br.

While there has been proposals on the usage of different security mechanisms [4], [16], it is interesting to evaluate the actual impact of such mechanisms on IoT constrained devices. Previous work [18] has evaluated the impact and overhead of security mechanisms compared to sensing and communication tasks on a Crossbow TelosB [19] device for both Contiki [9] and TinyOS [11] operating systems.

Nowadays the Texas Instruments CC2650 SensorTag [26] is an example of a low-power IoT node, being an embedded device with built-in IEEE 802.15.4 radio (capable of running BlutoothLE as well), an ARM© Cortex©-M3 MCU (32-bit) and multiple sensors, such as ambient light, infrared and ambient temperature, accelerometer, gyroscope, magnetometer, pressure, humidity, microphone, and magnetic sensor. In this work, we selected the CC2650 SensorTag as our experimental platform, running Contiki [9] as the operating system.

The main contribution in this paper is to present the energy consumption and execution time characterization of the TI CC2650 SensorTag platform running Contiki OS. We consider three types of tasks: communication, sensing and processing. For communication, we look into the cost of transmitting and receiving messages. Sensing evaluates all sensors available on the board. Concerning processing, we evaluate the usage of security algorithms, namely: AES [20], Curupira 2 [25], LetterSoup [24], OCB [17], Keccak [6], Blake2s [5], Marvin [24] and HMAC [10], as well as typical asymmetric cryptography primitives such as point addition, generic point multiplication and multiplication by generator.

The remainder of this paper is organized as follows: Section II presents and overview of the selected security mechanisms, Section III describes the performance evaluation methodology, and Section IV presents the results and analysis. We conclude the paper and discuss future work in Section V.

II. Security mechanisms

In order to establish secure communication, one needs to obtain several security services. Non-repudiation and source authentication can be provided by digital signatures, achieved through asymmetric ciphers. Data authenticity, integrity and confidentiality, which could be achieved by symmetric ciphers, are also important services to be provided by IoT applications.

To achieve confidentiality the sender must encrypt the message to be sent, and then the receiver must decrypt the message using the shared key. Any other entity monitoring the communication will not be able to understand the message, unless it knows the cipher and the symmetric key. We evaluated two block ciphers: AES [20] and Curupira 2 [25].

Message Authentication Codes (MAC) provide data authenticity. In this category, we evaluated Marvin [24] and HMAC [10]. Authenticated Encryption with Associated Data (AEAD) algorithms provides both confidentiality and authenticity, and we evaluated LetterSoup [24] and OCB [17].

Integrity ensures that a message was not corrupted or altered by a malicious entity. Hash algorithms generate fixed length tags from a message, used by the receiver to support integrity verification by comparing received and generated tags. In this category, we selected the Keccak [6] and Blake2s [5].

In the context of WSN and its limited storage and resources, Elliptic Curve Cryptography (ECC) is an attractive option for asymmetric cryptography. When compared to well-known RSA asymmetric algorithms, ECC algorithms use shorter keys (256-bits) to achieve similar security levels. Building upon ECC, we can create key exchange and validation protocols. We chose to evaluate common asymmetric ECC operations, such as point addition, generic point multiplication and multiplication by generator, using the RELIC [3] toolkit.

III. Methodology

In this section, we present an overview of the TI CC2650 SensorTag device [26], then we explain our measurement setup, followed by a description of the tasks considered for our evaluation.

A. SensorTag

The following devices and features were evaluated in terms of execution time and energy consumption. Sensors MPU9250, TMP007, and HDC1000 are connected to MCU through a two-wire I2C bus.

MPU9250 [15] is a 9-DOF sensor (accelerometer, gyroscope, and magnetometer) used to measure orientation, 3D and rotational acceleration. We evaluated this sensor measuring in could start and regime, as well as standby (powered but not measuring).

TMP007 [13] is a slow IR temperature sensor (0.26s with on-chip averaging disabled [13]). Our experiment enables the sensor and reads both temperatures (ambient and focused subject) with averaging disabled.

HDC1000 [14] is a low power temperature and humidity sensor. Measurement and read (transmission of measured values from HDC1000 to MCU) operations have been evaluated.

The BMP280 [23] temperature and barometric pressure sensor was used through Contiki's drivers. However, this driver set the sensor into forced mode, performing a single sensing before entering sleep state. This behavior is too fast to be reliably detected by our measurement setup and thus its results were suppressed. OPT3001 [12] light sensor's results are not shown for similar reasons.

The board's ADC was also evaluated. It can be used to measure battery voltage and temperature (through a NTC resistor (103GT-2)).

We measured the current drained by the board in idle, busy, and by the LEDs (red and green). Current was measured when leaving the MPU9250 in powered state, which is common since its cold startup takes much more time than one single read operation. The board buzzer (HCS0503B) was measured at selected frequencies.

B. Measurement setup

Measurement testbed is powered by an Agilent E3631A power supply [2], preset to 3V. An Agilent 34401A multimeter [1], connected as ammeter, has its data transferred to a computer running a custom LabView application, sampling at 500Hz through a GPIB/SCPI usb cable. Experiments were executed multiple times in order to calculate mean, standard deviation, and confidence intervals.

We ran each task considered and measured the current drained. We obtained the charge via time integration of the current and then, since voltage is constant, we obtain the energy consumption of each task. We also measured the current drained when the system was in idle (i.e., no particular tasks being executed), obtaining the threshold that is deduced from the measurements. Therefore, the system's overall energy consumption is given by the energy consumed by each task added to the energy consumed when the system is idle.

C. Software tasks

Contiki [9] is a thread-oriented, lightweight operating system focused on IoT, providing drivers for communication stacks and most of the board's sensors. Contiki and our applications are written in C.

RELIC is a lightweight asymmetric cryptographic toolkit, supporting many basic structures to build algorithms (such as ECC and multiple precision integer arithmetic) and presents standard key agreement, pseudorandom generators and hashing algorithms. We configured RELIC to 80-bit and 128-bit security levels, a reasonable representative for embedded devices, using curves secp160 [22] and NIST P-256 [21]. The following operations were evaluated: (i) point addition and (ii) generic point multiplication, (iii) multiplication by the curve generator, and (iv) hashing (SHA1 and SHA256). Our RELIC setup was generated using the following compilation keys:

```
WITH="EP;EC;DV;CP;MD;BN;FP" ARITH=easy
BN_MUL=BASIC BN_SQR=MULTP EB_KBLTZ=OFF
EB_METHD="PROJC;LWNAF;COMBS;INTER"
BN_PRECI=FP_PRIME={180,256} FP_PMERS=ON
FP_METHD="BASIC;COMBA;COMBA;QUICK;LOWER;
BASIC" FPX_METHD="INTEG;INTEG;LAZYR"
FP_QNRES=OFF PP_METHD="LAZYR;OATEP"
MD_METHD={SHONE,SH256} BN_MXP=BASIC
FP_RDC=QUICK FP_INV=LOWER FP_EXP=BASIC
FPX_QDR=BASIC FPX_CBC=BASIC FPX_RDC=BASIC
EP_ENDOM=OFF EP_DEPTH=3 PP_EXT=BASIC
```

Block cyphers and AEAD were evaluated regarding main functions: *init*: initialize variables, keys, initialization vector and nonce, *encrypt* and *decrypt* operations. Performance of encryption and decryption depends on message length.

Each symmetric algorithm has a specific block size. Curupira 2 uses 12-bytes blocks and keys, AES uses 16-bytes blocks and keys and uses 60-bytes initialization vector (IV). LetterSoup and OCB uses 12-bytes blocks, keys and tags, and

60-bytes associated data. We used 60-bytes messages for all algorithms evaluated, since this is an average message size for IoT/WSN messages running over IEEE 802.15.4 (whose maximum frame size is 127-bytes including headers).

For Hash and MAC we selected the operations *init*: initialize variables, keys and nonce, *update*: update variables of state, and *final*: get tag or hash. Only the update phase depends on message length.

Marvin uses 12-bytes key and tag, HMAC uses 12-bytes key and 32-bytes tag, Keccak uses 64-bytes hash and Blake uses 32-bytes key and hash and 64-bytes block.

All symmetric algorithm operations were executed N times in order to measure the current drained and obtain the execution time, given our sampling rate of 500 Hz. The N value is *6000* for AES, CURUPIRA 2, Keccak, and Blake2s; *3000* for Marvin and HMAC; and *500* for LetterSoup and OCB. We used 5 random samples to calculate the average and confidence interval with confidence level of 95% for the normal distribution.

For communication tasks we evaluated execution time and energy consumption for transmission and reception of 60-bytes messages, transmitted as unicast. The application uses the Rime stack without routing protocol and ContikiMAC as the medium access layer protocol, configured with 8 Hz channel verification rate.

IV. RESULTS AND DISCUSSION

Table I depicts current drained in idle and busy (full CPU usage) state, keeping sensors enabled, and enabling LEDs.

TABLE I
SENSORTAG CURRENT DRAINED

State	Current (mA)
Idle	0.0258±0.0009
Busy	8.56±0.03
MPU9250 on	0.181±0.003
Serial on	0.105±0.002
Buzzer (625Hz)	44.9±0.1
Buzzer (2500Hz)	22.31±0.01
Buzzer (5000Hz)	22.35±0.01
Buzzer (10000Hz)	22.33±0.01
Buzzer (20000Hz)	21.94±0.01
Red LED	1.402±0.009
Green LED	2.72±0.01
Both LEDs	4.100±0.009

Table II depicts execution time and energy consumption for sensing tasks. We ran 16K measurements to evaluate battery and NTC performance (ADC).

RELIC Toolkit's point by scalar and generator by scalar multiplication have been evaluated. These are the most expensive basic operations on ECC. Point addition and data block hashing have also been evaluated to allow comparison. Results are presented in Table III.

Results for symmetric security algorithms are presented in Tables IV, V, VI and VII. We notice that Hash algorithms present the largest difference of execution time and energy usage. MAC, AEAD and Block Ciphers algorithms show a lower relative difference.

TABLE II
SENSORTAG SENSING EVALUATION

Sensor	Time (ms)	Energy (mJ)
Battery voltage (16000 executions)	13.0±0.3	0.111±0.003
NTC temperature (16000 executions)	25.1±0.3	0.207±0.002
MPU9250 cold start	92.2±0.1	0.851±0.004
TMP007	282±2	0.20±0.03
HDC1000 (100 measurements)	2468.05±0.03	9.7±0.7
HDC1000 (1000 reads)	216.8±0.5	0.156±0.003

TABLE III
SENSORTAG ASYMMETRIC CRYPTOGRAPHY EVALUATION

Operation	Time (ms)	Energy (mJ)
80-bits		
Hash (1000 executions)	111.998±0.006	1.1552±0.004
Point Addition (200 executions)	115.2±0.1	1.1843±0.005
Generic point multiplication	241±1	2.48±0.01
Multiplication by generator	114±5	1.17±0.05
128-bits		
Hash (1000 executions)	137.4±0.6	1.4168±0.005
Point Addition (200 executions)	1800±12	18.5±0.1
Generic point multiplication	2193±7	22.5±0.1
Multiplication by generator	1072±43	11.0±0.5

Comparing block ciphers, we notice that the fastest *init* operation is performed by AES. However, the speed of this operation cannot solely define the best algorithm. The best evaluation is done on operations *encrypt* and *decrypt*. CURUPIRA 2 was designed for embedded platforms, considering 8-bit or 16-bit words. Since SensorTag uses a 32-bit processor, AES presents better results than CURUPIRA 2.

TABLE IV
AVERAGE OF TIME AND ENERGY FOR ONE EXECUTION FOR EACH TASK OF BLOCK CIPHERS ALGORITHMS

Task	Time(μs)	Energy(μJ)
AES		
init	21.5±0.2	0.224±0.001
encrypt	102.1±0.1	1.071±0.001
decrypt	102.2±0.2	1.073±0.001
Curupira 2		
init	48.5±0.2	0.523±0.002
encrypt	468±1	4.872±0.008
decrypt	483.9±0.1	5.034±0.008

Similar to block ciphers, for Hash algorithms the cost of the *init* operation is independent of message size. Thus we analyzed the operation *update*, where we can see that Blake2s is more efficient. The *final* operation is the fastest between the 3 steps, and has fixed duration, so we did not consider it in the choice of the best algorithm.

TABLE V
AVERAGE OF TIME AND ENERGY FOR ONE EXECUTION OF EACH TASK OF HASH ALGORITHMS

Task	Time (μs)	Energy (μJ)
Blake2s		
init	23.2±0.2	0.241±0.001
update	54.8±0.2	0.617±0.001
final	60.4±0.2	0.6810±0.0007
Keccak		
init	109.9±0.2	1.034±0.003
update	991.4±0.2	10.38±0.02
final	1.31±0.01	0.0127±0.0001

In the case of AEAD algorithms, LetterSoup presents better performance than OCB for all operations. Thus, it is more energy efficient, even though the difference is small.

TABLE VI
AVERAGE OF TIME AND ENERGY FOR ONE EXECUTION FOR EACH TASK OF AEAD ALGORITHMS

Task	Time (μs)	Energy (μJ)
LetterSoup		
init	191±2	2.00±0.02
encrypt	867.99±0.01	9.07 ± 0.01
decrypt	867.96±0.04	9.08±0.01
OCB		
init	199±2	2.03±0.02
encrypt	985±2	10.13±0.02
decrypt	987±2	10.15±0.02

On MAC algorithms, we show that HMAC is slightly faster for any operation, due to 32-bit optimizations.

TABLE VII
AVERAGE OF TIME AND ENERGY FOR ONE EXECUTION OF EACH TASK OF MAC ALGORITHMS

Task	Time (μs)	Energy (μJ)
Marvin		
init	294.4±0.3	2.857±0.005
update	417.993±0.003	4.063±0.004
final	370.654±0.007	3.589±0.003
HMAC		
init	204.3±0.3	2.153±0.003
update	88.5±0.3	0.924±0.003
final	240.1±0.3	2.504±0.003

In terms of time and energy for tasks communication, use of ContikiMAC, Contiki's native MAC, is more optimized than X-MAC [8], when compared with measurements presented by Margi *et al.* [18]. This can be justified by ContikiMAC's synchronous wake-up mechanism, and ContikiMAC uses a cheap Clear Channel Assessment (CCA) mechanism that uses the Received Signal Strength Indicator (RSSI) of the radio transceiver to indicate radio activity on the channel.

In order to receive packets the receiver periodically verifies if the connected channel has frames being transmitted to it and, if so, it receives the next frame to obtain the whole information. If not, it sleeps and checks the channel again. The time and energy spent to receive the frame or just check for frames are very similar, as we can see in the Table VIII. For these measurements a significant difference could not be accurately obtained.

TABLE VIII
AVERAGE OF TIME AND ENERGY FOR ONE EXECUTION OF EACH TASK OF COMMUNICATION (TRANSMISSION AND RECEPTION)

Task	Time (ms)	Energy (μJ)
TX	111.986±0.008	3320±10
RX	3.9±0.2	84±5
RX - just receiving	5±1	100±30

Graphs 1, 2, 3, and 4 summarize the results presented. Energy graphs were chosen since it is considered the most valuable resource in a WSN evaluation, usually over the time to complete an operation. Notice that the four graphs are depicted in different scales, since the energy required for different tasks change in orders of magnitude. ECC operations and communication operations are shown in logarithmic scale, the ECC operations with bits of security level in parenthesis.

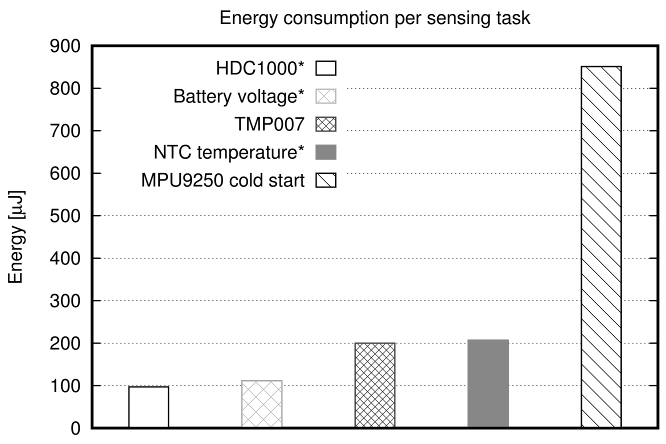

Fig. 1. Energy consumption of sensors. Columns marked with * are multiple measurements.

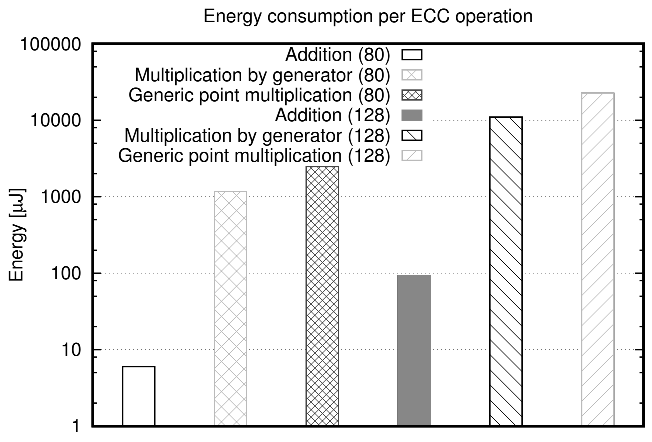

Fig. 2. Energy consumption of ECC operations (logarithmic scale)

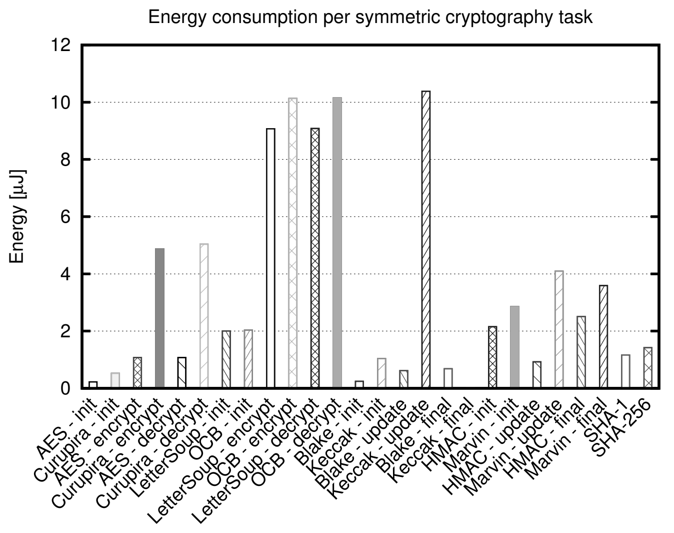

Fig. 3. Energy consumption of symmetric cryptography and hashing operations

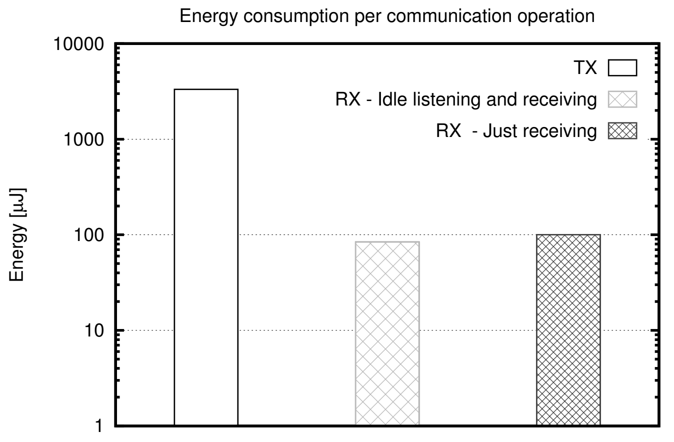

Fig. 4. Energy consumption of communication operations (logarithmic scale)

V. Conclusions

Compared with the load generated by communication, cryptography services are not expensive. ECC operations can be time costly, but are not often executed (usually once, at key agreement and validation steps, both at the bootstrap of secure communications). Symmetric cryptography algorithms follow the same trend, requiring less power, since even ones not tailored for WSN (such as SHA256) have an acceptable impact on performance/battery life.

As we expected, sensing is costly but communication determines the energy usage, even when using low power protocols. Considering communication and sensing energy cost, associated with the relative small processing time for frequent operations, we conclude that security services are feasible for Sensortag IoT device.

Our results could be compared with other works (e.g. [18]), noticing that some differences arose, such as CURUPIRA 2 worse performance due to its design being tailored for 8-bit or 16-bit MCU. This highlights the importance of knowing your target platform.

Acknowledgments

This work is partially funded by Fundação para o Desenvolvimento Tecnológico da Engenharia (FDTE) grant #1450 and FAPESP grant #2013/15417-4. Cíntia Borges Margi is supported by CNPq research fellowship #307304/2015-9.

References

[1] Agilent. Agilent 34401a multimeter, 2007.
[2] Agilent. E363xa series programmable dc power supplies, 2009.
[3] D. F. Aranha and C. P. L. Gouvêa. Relic is an efficient library for cryptography.
[4] Luigi Atzori, Antonio Iera, and Giacomo Morabito. The internet of things: A survey. *Comput. Netw.*, 54(15):2787–2805, October 2010.
[5] Jean-Philippe Aumasson, Samuel Neves, Zooko Wilcox-O'Hearn, and Christian Winnerlein. Blake2 official webpage.
[6] BERTONI, G; DAEMEN, J; PEETERS, M; VAN ASSCHE, G. The keccak reference, 2011.
[7] David Culler, Deborah Estrin, and Mani Srivastava. Overview of sensor networks. *Computer Magazine*, 37(8):41–49, 2004.
[8] Adam Dunkels. The contikimac radio duty cycling protocol. Technical report, 2011.
[9] Adam Dunkels, Bjorn Gronvall, and Thiemo Voigt. Contiki - a lightweight and flexible operating system for tiny networked sensors. In *LCN '04: Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks*, pages 455–462, Washington, DC, USA, 2004. IEEE Computer Society.
[10] Olivier Gay. HMAC implementation.
[11] Jason Hill, Robert Szewczyk, Alec Woo, Seth Hollar, David Culler, and Kristofer Pister. System architecture directions for networked sensors. *SIGPLAN Notices*, 35(11):93–104, 2000.
[12] Texas Instruments. Opt3001 ambient light sensor (als), 2014.
[13] Texas Instruments. Tmp007 infrared thermopile sensor with integrated math engine, 2015.
[14] Texas Instruments. Hdc1000 low power, high accuracy digital humidity sensor with temperature sensor, 2016.
[15] InvenSense. Mpu-9250 product specification, 2014.
[16] Sye Loong Keoh, S.S. Kumar, and H. Tschofenig. Securing the internet of things: A standardization perspective. *Internet of Things Journal, IEEE*, 1(3):265–275, June 2014.
[17] T. Krovetz and P. Rogaway. RFC 7253 - The OCB Authenticated-Encryption Algorithm.
[18] Cíntia B. Margi, Bruno T. de Oliveira, Gustavo T. de Sousa, Marcos A. Simplício, Paulo S. L. M. Barreto, Tereza C. M. B. Carvalho, Mats Näslund, and Richard Gold. Impact of operating systems on wireless sensor networks (security) applications and testbeds. In *Computer Communications and Networks (ICCCN), 2010 Proceedings of 19th International Conference on*, pages 1–6, Aug. 2010.
[19] MEMSIC Inc. telosb product details, 2012. 6020-0094-04 Rev B.
[20] NIST. Announcing the advanced encryption standard (AES), 2001.
[21] National Institute of Standards and Technology. Recommended elliptic curves for federal government use.
[22] Certicom Research. Sec 2: Recommended elliptic curve domain parameters.
[23] Bosch Sensortec. Bmp280 digital pressure sensor datasheet, 2015.
[24] Marcos Simplicio, Pedro d'Aquino, Paulo Barreto, Tereza Carvalho, and Cintia Margi. The Marvin Message Authentication Code and the LetterSoup Authenticated Encryption Scheme. In *Wiley InterScience - Security and Communication Networks*, 2009.
[25] Marcos A. Simplício, Paulo S. L. M. Barreto, Tereza C. M. B. Carvalho, Cíntia B. Margi, and Mats Näslund. The CURUPIRA-2 block cipher for constrained platforms: Specification and benchmarking. In *Proc. of the 1st International Workshop on Privacy in Location-Based Applications - 13th European Symposium on Research in Computer Security (ESORICS'2008)*, volume 397. CEUR-WS, 2008.
[26] Texas Instrument. Multi-standard cc2650 sensortag design guide, 2015. TIDU862 – March 2015.
[27] Roy Want, Bill N. Schilit, and Scott Jenson. Enabling the internet of things. *Computer*, 48(1):28–35, Jan 2015.