Energy-Efficient Medium Access Control for Energy Harvesting Communications

Xenofon Fafoutis, *Member*, IEEE, Alessio Di Mauro, Charalampos Orfanidis, and Nicola Dragoni

**Abstract — ***While energy consumption is widely considered the primary challenge of wireless networked devices, energy harvesting emerges as a promising way of powering the Internet of Things (IoT). In the Medium Access Control (MAC) layer of the communication stack, energy harvesting introduces spatial and temporal uncertainty in the availability of energy. In this context, this paper focuses on the design and implementation of the MAC layer of wireless embedded systems that are powered by energy harvesting; providing novel protocol features and practical experiences to designers of consumer electronics who opt for tailoring their own protocol solutions instead of using the standards*[1].

Index Terms — Medium access control; energy harvesting communications; energy harvesting - wireless sensor networks; internet of things

I. Introduction

Continuous advances in low power electronics enable the realization of the Internet of Things (IoT). IoT-enabling technologies, such as wearable devices [1] and smart home infrastructures [2], emerge in the consumer electronics market. Fall detection for monitoring the elderly [3], wearable activity monitors [4] and building lighting automation [5] are only few of numerous applications.

With a trend towards miniaturized hardware and ubiquitous technologies, battery lifetime is a common engineering challenge. Energy-efficient wireless communications is widely considered a key factor for long battery lifetimes. A battery, however, is bound to be depleted at some point in time. As a result, energy harvesting becomes an attractive alternative. Indeed, energy harvesting technologies are able to offer a perpetual operation that is not limited by the capacity of the battery. Several sources of ambient energy have been considered in the literature, including ambient indoors light [6], RF energy harvesting [7], inductive contactless charging [8] and even human powered solutions [9].

Energy harvesting introduces new challenges in energy-efficient communications. Ambient energy is neither always available nor evenly distributed in space. Due to such spatiotemporal variations, it can be neither assumed that sensor nodes within a network are always available, nor that all sensor nodes have equal resources. Similarly to battery-powered communications, energy-efficiency is vital. However, in scenarios where plenty of ambient energy is available, energy harvesting communication protocols can, instead, leverage any excess energy to improve the performance of the system.

The Medium Access Control (MAC) communications layer plays a key role in energy-efficient wireless networks [10]. In addition to being responsible for the establishment of communication links between nodes and for regulating the access to the wireless channel, the MAC protocol manages the duty cycle of the radio. In links where both the sender and the receiver are energy-constrained, radio duty cycling introduces the challenge of finding a moment in time where both the sender and the receiver are active, and a communication link can be established. MAC protocols follow either a synchronous or an asynchronous approach against this challenge [11]. Within the asynchronous direction, there are two fundamental protocol classes, namely the sender-initiated [12] and the receiver-initiated [13] paradigms of asynchronous communication.

This paper focuses on the design, implementation, and practical evaluation of ODMAC (On-demand MAC). ODMAC is MAC protocol that follows the receiver-initiated paradigm of asynchronous communications and aims to address the challenges of energy-harvesting communications: (i) energy-efficient utilization of available resources; (ii) sustainable operation in environments of unpredictable energy input; and (iii) utilization of any excess of available energy for the application performance. The paper follows a system engineering approach, providing protocol features and practical insight to researchers and engineers who opt for tailoring their own protocol solutions instead of using communication standards.

The remainder of the paper is structured as follows. Section II briefly summarizes the related work. Section III focuses on protocol design. Section IV presents the implementation of the protocol on a commercial off-the-shelf microcontroller. Section V evaluates the protocol using a commercial off-the-shelf photovoltaic energy harvester. Lastly, Section VI concludes the paper.

[1] This work was supported in part by the IDEA4CPS project granted by the Danish National Research Foundation.

X. Fafoutis was with the Technical University of Denmark, DK. He is now with the University of Bristol, UK (email: xenofon.fafoutis@bristol.ac.uk).

A. Di Mauro was with the Technical University of Denmark, DK. He is now with Yubico, SE (email: alessio@yubico.com).

C. Orfanidis was with the Technical University of Denmark, DK. He is now with the University of Uppsala, SE (charalampos.orfanidis@it.uu.se).

N. Dragoni is with the Technical University of Denmark, DK. He is also with the Örebro University, SE (ndra@dtu.dk).

Contributed Paper
Manuscript received 09/25/15
Current version published 12/29/15
Electronic version published 12/29/15.

II. Related Work

In links where both the sender and the receiver duty cycle their radios, MAC protocols follow either a synchronous or asynchronous approach on finding a moment in time where both nodes are active and a link can be established.

In protocols that follow the synchronous approach, like the beacon-based IEEE 802.15.4, Sensor-MAC (S-MAC) [14], T-MAC [15], and Pairwise [16], nodes organize their active and sleeping states to align. Synchronous schemes can be based either on contention or on reserved timeslots. In both cases, a portion of the active state is used to synchronize all the nodes to a global active/sleep schedule. Synchronous schemes can be tolerant to schedule misalignment; however, they still require a globally synchronized schedule, which creates an additional energy overhead. Moreover, synchronous protocols have a cost associated with the creation and maintenance of the schedule. Furthermore, the coupling of nodes via a global clock also hinders a node's ability to have a fully independent duty cycle, so that each node can adapt, in a fully distributed way, to the surrounding conditions.

Asynchronous schemes do not require synchronization, as the nodes sleep and wake up independently of the others. This leads to the need of techniques on deciding a rendezvous point for nodes to communicate. There are two fundamental asynchronous techniques. Preamble sampling is the basic technique used in sender-initiated asynchronous MAC schemes. The sender transmits a preamble to indicate that there is a pending need for communication. The receiver wakes up occasionally into the active state to listen to such a preamble transmission. Once the preamble is detected, the receiver replies with a positive acknowledgment to the sender after the preamble transmission is finished. This establishes a communication link between the sender and receiver. Most notable examples of sender-initiated MAC protocols are Berkley MAC (B-MAC) [17] and Short Preamble MAC (X-MAC) [18]. Sender-initiated MAC protocols are primarily used in popular operating systems, such as Contiki and TinyOS. Comparisons suggest that asynchronous schemes are more efficient than synchronous approaches [17][29].

In contrast to the preamble sampling technique of sender-initiated schemes, receiver-initiated schemes use a different approach to achieve asynchronous communication: instead of long preambles, the sender listens to the channel, waiting for small beacons transmitted by the receiver. The receiver transmits the beacons in a period that is defined by its duty cycle. Beacons are used by the sender to synchronize with the receiver. The receiver-initiated paradigm was introduced by Receiver Initiated Cycled Receiver (RICER) [19] and made popular by Receiver Initiated MAC (RI-MAC) [20] and A-MAC [21]. Receiver-initiated MAC schemes are shown to be more energy-efficient than sender-initiated schemes [20].

There are dozens of receiver-initiated MAC protocols in the literature [13], offering various features that focus on different constraints, scenarios and applications. For example, QAEE-MAC [23] and EEPB-MAC [24] focus on traffic with high

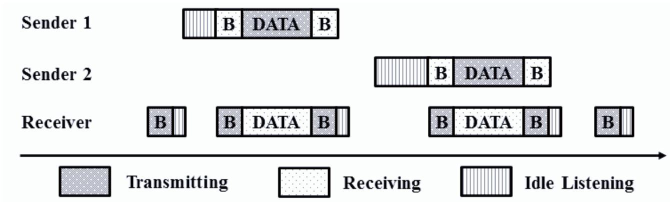


Fig. 1. ODMAC follows the receiver-initiated paradigm of asynchronous communication.

priority; ERI-MAC [25] concatenates small packets to larger packets; RP-MAC [26] reorders packets in the transmission queue; RF-MAC [27] focuses on RF energy harvesting; and ADM [28] extends the paradigm with broadcasting support. The wide variety of protocol variations and features in the literature indicates that there is no global solution that performs well in every possible environment and application. On the contrary, a specific technique could be very good in one scenario and disastrous in another. Therefore, a network designer is supposed to mix and match the available protocol features in order to craft a solution that suits the constraints of the desired application. ODMAC, being no exception, offers a collection of unique features that are summarized in the following section. Although designed for synergies, these features do not necessarily have to be used as a package, as presented in this paper. Instead, the authors urge the reader to combine them with other protocol features in the literature and tailor a solution for their particular target application.

ODMAC has been thoroughly compared with the sender-initiated X-MAC [18] and other receiver-initiated protocols in previous works [13][22].

III. Protocol Design

ODMAC is a protocol that is designed on the key system goals of energy harvesting communications: sustainability, energy-efficiency and application performance [30].

Energy harvesting introduces spatial and temporal uncertainty in the availability of energy. A key requirement of MAC schemes for energy harvesting communications is allowing nodes to independently adjust their duty cycle to adapt to the energy that can be harvested. Hence, synchronous protocols are considered unsuitable as, in a synchronous network, the duty cycles of the sensor nodes are coupled to each other via a global clock. ODMAC is following the receiver-initiated asynchronous paradigm, which is shown to be more efficient than the other communication paradigms [22][29].

The unpredictable, ever-changing and small-scale nature of the energy input makes adaptable radio duty-cycling the only means to achieving sustainable operation. Specifically, in energy constrained environments, the MAC protocol needs to support very low duty cycles in order to guarantee the long-term sustainability of the system. On the other hand, when the energy is abundant, it has to efficiently use the energy surplus to increase the application performance. Beyond the adaptable duty cycles, ODMAC incorporates additional features that address most of the challenges of receiver-initiated MAC protocols, including the mitigation of idle listening; the energy-efficient avoidance of collisions; the prioritization of urgent traffic; and the provision of secure communication.

A. Basic Operation and Adaptive Duty Cycles

The receiver-initiated paradigm constitutes the foundation of all the receiver-initiated asynchronous protocols, including ODMAC. According to the paradigm, a node willing to receive data, wakes up periodically and checks for incoming transmissions via the transmission of a special control frame, named beacon (B). To do so, a Clear Channel Assessment (CCA) is performed immediately after waking up, and upon the broadcast of a beacon, the receiver continues to listen to the channel for a short predefined period of time. Meanwhile, whenever a node with data ready to be sent enters the active state, it listens silently for a beacon from the intended receiver. Once the beacon is received, the sender transmits its data packet, and waits for another beacon which acknowledges (ACK) the reception of the data. Conversely, if there is no incoming data after transmitting the beacon, the receiver enters the sleeping state. The process is shown in Fig. 1.

To adapt to the ever-changing unpredictable nature of the energy input, nodes dynamically adjust their duty cycle in a completely independent and distributed manner. Nodes in the network have a double role of receivers for forwarding tasks and senders for sensing tasks. ODMAC decouples the duty cycles of these two tasks within a single node. Hence, a node has a beaconing and a sensing duty cycle. The beaconing duty cycle controls the trade-off between energy consumption and end-to-end delay, while the sensing duty cycle controls the trade-off between energy consumption and throughput. Thus, ODMAC grants the user the ability to tune the network to different application requirements.

ODMAC adapts the duty cycles based on the ENO (Energy Neutral Operation) principle [31]. According to the ENO principle, a node is sustainable if, over a time period that its energy buffers can support, the consumed energy is less than or equal to the harvested energy. All nodes in the network dynamically adjust the beacon and sensing duty cycle, in order to achieve and maintain an ENO-Max state [32], which is defined as an ENO state with maximum performance. This means that when the consumed energy is more than the harvested energy, the duty cycles are decreased to reduce the energy consumption. In the same manner, when the consumed energy is lower than the harvested energy, the duty cycles are increased. Thus, the adaptation of the duty cycles follows a greedy approach [33].

B. Opportunistic Forwarding

Opportunistic Forwarding (OF) is a forwarding scheme that leverages the random nature of a beacon reception for the energy-efficiency and sustainability of the network.

If a routing protocol is aware that the energy consumption (*i.e.* primarily idle listening while waiting for a beacon) depends on the duty cycles, it can include this information in its routing metric and, essentially, route traffic, more energy-efficiently, through the nodes that have higher beaconing frequencies. This solution has two limitations. The first limitation is that the selected node is overloaded with all data transmissions. The second limitation is that routing traffic through the nodes that transmit beacons more frequently is not always the most efficient choice. Instead, it is *on average* the most efficient choice. If one evaluates each beacon transmission separately, there will be some cases where a beacon from other nodes would arrive earlier.

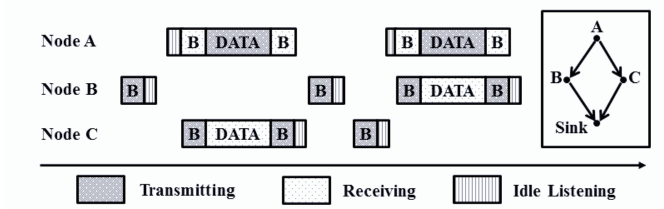


Fig. 2. Opportunistic forwarding reduces idle listening by forwarding traffic via nodes that harvest more energy on a per-packet basis.

The proposed scheme builds upon these two limitations. Instead of waiting for a specific receiver to wake up, a sender opportunistically forwards data to any approved receiver (*anycast* forwarding), based on the beacon obtained first, as illustrated in Fig. 2. Since the probability of receiving beacons from a receiver with surplus energy is high, this forwarding scheme creates a more robust network, which is adaptive to changes in energy, by keeping the load in the network balanced between the routing options. In the long-term, the traffic is divided, in a fully autonomous manner, to multiple receivers according to the harvested energy and the duty cycles of each individual node. Inherently, the traffic distribution autonomously adapts to changes in the energy input, as it follows the adaptation of the duty cycle. Furthermore, this mechanism significantly improves the energy efficiency of the link, as the time spent by the senders waiting for a beacon (*i.e.* idle listening), and therefore their energy consumption, is reduced.

OF requires a routing protocol that assigns each sender a list of approved receivers. Layer-based Anycast Routing (LAR) is a simple, minimal overhead, hop-count routing protocol that selects multiple forwarders. While, technically, not part of the MAC layer, it is implemented inside the MAC layer in whole.

LAR operates as follows. The layer of node u is defined as its distance from the sink, expressed in number of hops. The sink is initialized at layer 0. All nodes advertise their layer through their beacons and nodes update their layer upon beacon reception. As node u operates, it receives a set of beacons from the neighboring nodes that are in range (with a link quality indicator that exceeds a set threshold). It assigns its own layer by incrementing the minimum layer value of the beacons received by 1. If no beacon is received after a predefined amount of time, node u resets its layer to a maximum value. Nodes advertising a layer lower than the one of the sender, thus leading towards the sink, are considered forwarding candidates. By using the link-layer beacons to distribute information required for routing decisions, the transmission of extra control packets is avoided and energy is saved. Moreover, this routing scheme is resilient to nodes entering and exiting the network.

C. Collision Avoidance and Traffic Differentiation

In receiver-initiated protocols, beacons form time slots of communication. Randomization techniques can distribute data transmissions among multiple beacons. Nevertheless, when multiple nodes wake up and wait for the same beacon, a collision is inevitable.

The standard solution for collision avoidance in wireless networks is named Random Backoff (RB). The idea is that the MAC protocol delays the data transmission by a random number of timeslots, while listening to the channel for other transmissions. If the channel remains idle, data transmission follows. If the channel gets occupied by another transmission, the node backs off and attempts to transmit at a later time. Variations of the RB algorithm are the most commonly used collision avoidance mechanisms in receiver-initiated MAC protocols ([19]-[21], [23]). The mechanics of RB imply that senders that contend for the same beacon will spend a vast amount of energy waiting for the beacon and the collision will be detected and resolved only after the beacon transmission.

ODMAC incorporates Altruistic Backoff (AB); a collision avoidance mechanism that detects potential collisions and avoids them before the actual beacon transmission [34]. In AB, a node with data to transmit wakes up and, before it starts waiting for a beacon, it transmits a control packet, named Altruistic Backoff Request (ABR), which contains the identifiers of the intended receivers. A node that is already waiting for the same beacon and receives this packet altruistically backs off, offering the beacon to the node that wakes up last. At the low overhead of one extra control packet transmission per data transmission, collisions are mitigated and idle listening is significantly reduced. Fig. 3 shows an example of collision avoidance with both AB and RB, providing intuition on the benefits of the former (ABR is denoted as R).

AB does not suffer from fairness issues. Random channel access provides similar probabilities for all nodes to use the beacon. Essentially, the beacon, and thus the channel, is taken by the sender that wakes up last. As long as different senders have equal opportunities to wake up last, they have equal opportunities to take the beacon. Therefore, random channel access guarantees long-term fairness.

AB also offers traffic differentiation. Traffic differentiation is valuable in case of applications that generate traffic of different urgency (e.g. alerts vs. monitoring traffic). Two types of data packets, which correspond to two traffic classes, are defined: the high-priority class and best-effort class. The priority number that defines the priority class is included in the ABR. Upon the reception of an ABR, a node compares the priority number indicated in the ABR to the priority number of the local packet it has to transmit. If and only if the local packet belongs to the high-priority traffic class and the remote packet belongs to the best-effort traffic class, the node immediately transmits a new ABR to reclaim the beacon, as shown in Fig. 4. As a result, the priority number guarantees that ABR retransmissions occur only when a node has a higher priority than the node that currently has the beacon.

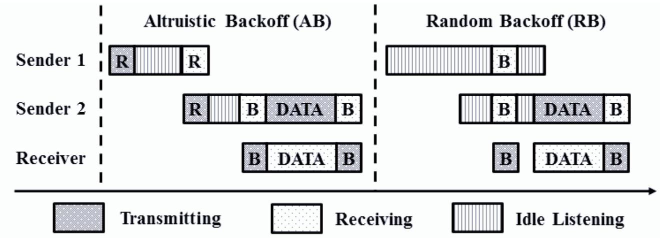


Fig. 3. With Altruistic Backoff, contending nodes avoid collisions before the transmission of a beacon, avoiding unnecessary idle listening.

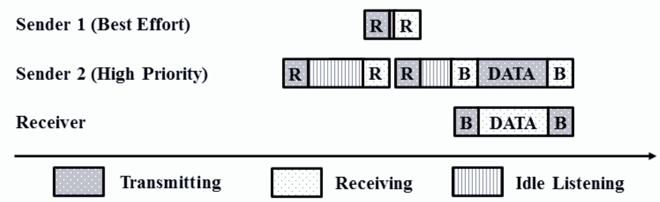


Fig. 4. ODMAC prioritizes urgent traffic by allowing senders of high priority data to reclaim the channel.

Upon a back-off event, the time of a next transmission attempt can follow different policies with respect to the importance of the data. Two extreme policies can be identified. On one hand, the sender might attempt to transmit immediately, as recommended for traffic of high priority. On the other hand, the sender might choose to buffer the packet and transmit it together with the following packet. This policy is recommended for best-effort traffic, as it is the policy that minimizes the energy consumption. Additionally, the sender may choose a solution in between, compromising the two extremes.

D. Link-Layer Authentication and Encryption

State-of-the-art security extensions have been included within ODMAC to provide confidentiality and integrity. The security subsystem is loosely based on TinySec [35] and provides four modes of operations: no security, authentication, encryption, and both. All the properties are provided by using Skipjack; an inexpensive cryptographic primitive. According to the required functionality, authentication and encryption can be activated with a single message granularity. Besides payload messages, encryption is also applied to beacon messages.

IV. Protocol Implementation

The implementation of ODMAC constitutes part of a complete firmware that also implements power management, routing and application-related functionalities [36]. The firmware implementation is targeted for off-the-shelf microcontrollers and 2.4 GHz radios. For energy harvesting, an off-the-shelf photovoltaic energy harvester is employed.

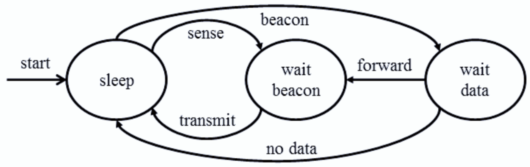


Fig. 5. ODMAC's sensing and beaconing / forwarding functionalities are implemented as a state machine.

A. *ODMAC as a Finite State Machine (FSM)*

The core of ODMAC is implemented as an FSM, as shown in Fig. 5. Its functionality is mainly based upon two routines, namely *Send* and *Receive*. Unless one of these two handlers is invoked, ODMAC is in sleeping state and the radio is off.

The *Send* routine generates and formats a packet around the payload (*e.g.* the result of a sensing operation). When the packet is ready, the radio is set to listening mode and the state machine waits for an interrupt signaling the reception of a beacon. Different packet types might be received while waiting for a suitable beacon. While non-beacon packets are simply discarded, all beacons are evaluated. Upon the signaling of the first suitable beacon, ODMAC continues its execution, and the packet is transmitted. At the end of a transmission, the radio is switched off.

The *Receive* handler is invoked during the forwarding duty cycle. In particular, it generates and broadcasts a beacon packet. At this point, the radio is set to listening mode and the protocol waits for a data packet for a fixed amount of time. If no incoming data is received during this period, the radio is set back to sleep mode and the routine ends. On the other hand, upon receiving a data packet, the information contained is extracted and the radio is set back to sleep mode. In order to forward the newly received packet toward the sink, a new invocation of *Send* is performed.

B. *Implementation of Duty Cycles*

Duty cycles are implemented through wake-up interrupts using the low-frequency timer of the microcontroller. A time quantum is defined. It controls the sleeping time between two subsequent wake-up events. On top of that, the two independent duty cycles for the sensing and the forwarding tasks are implemented as multiples of the basic time quantum. Additionally, the time quantum is periodically adjusted, by adding a uniformly random number of cycles to the defined value. This randomization prevents unfortunate synchronizations and decreases the collisions by enforcing random channel access between different nodes. While in the sleep state, the MCU is configured in a low power mode in which only the auxiliary low-frequency oscillator, used to schedule the interrupts, is active.

C. *Implementation of Anycast Forwarding*

ODMAC implements and incorporates inside its routines the OF scheme and the LAR algorithm. Specifically, the sink node initializes its layer to 0, while all the sensor nodes initialize their layer to 99 which represents that the nodes are disconnected from the network. Unless disconnected from the network, nodes advertise their layer through their beacons.

All nodes update their layer during the beacon evaluation of the *Send* routine. In particular, there are four distinct cases. (i) A sender may receive a beacon that advertises a layer that is greater than or equal to its own layer. In this case, the beacon is discarded and the node remains in listening mode. (ii) A sender may receive a beacon that advertises a layer that is lower than its own layer by exactly 1. In this case, the beacon is marked as suitable and an interrupt is generated that signals the data packet transmission. (iii) A sender may receive a beacon that advertises a layer that is lower than its own layer by more than 1. In this case, the sender updates its layer to 1 more than the layer advertised by the beacon. Then, the beacon is marked as suitable and an interrupt is generated that signals the data packet transmission. (iv) A sender may not receive any suitable beacon within a predefined time interval. In this case, it updates its layer to 99 and considers itself disconnected from the network.

D. *Implementation of Collision Avoidance*

AB extends the *Send* routine as follows. The ABR control packet is implemented similarly to a beacon. Specifically, the ABR includes the layer that indicates the group of beacons that the sender is waiting for, to any potential contenders that happen to be awake. After a successful CCA the transmission of the ABR follows. Then, the radio is switched to listening mode and the sender begins to listen for a beacon. Listening is interrupted either by the reception of a suitable beacon or by the reception of an ABR that advertises the same layer as the layer of the sender. In the former case, data transmission follows normally. In the latter case, the routine returns and indicates a backoff. The *Send* routine performs one attempt to transmit the packet. In case of backoff, it is up to the higher layer to decide at which point in the future will retry to transmit the packet.

For the traffic differentiation services of AB, a priority bit is added in the header of ABR control packets. The priority bit indicates if the data packet is classified as *High Priority* or *Best Effort*. When a sender that waits for a beacon receives an ABR packet, it compares its local priority bit with the received priority bit. If and only if the local data packet is classified as *High Priority* and the received ABR indicates a *Best Effort* data packet, the sender reclaims the channel by invoking the *Send* routine again. Otherwise, the sender backs off.

E. *Security Extensions*

Link-layer authentication and encryption services are provided through the Skipjack encryption primitive. Its implementation is based on the open source version available within OpenBSD, and it is changed accordingly to meet the memory constraints of the microcontroller. Authentication appends to the packet a 4-byte footer that contains the message authentication code. Any authenticated packet whose code is not verified correctly is dropped. In case both encryption and authentication are enabled, encryption is

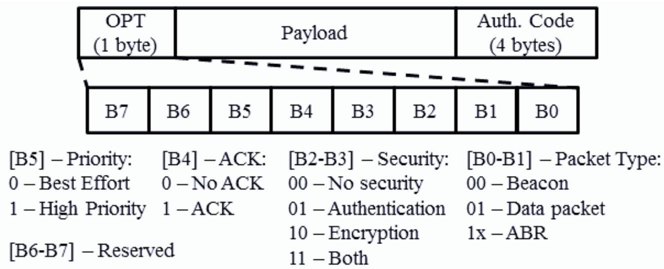


Fig. 6. The packet format of ODMAC.

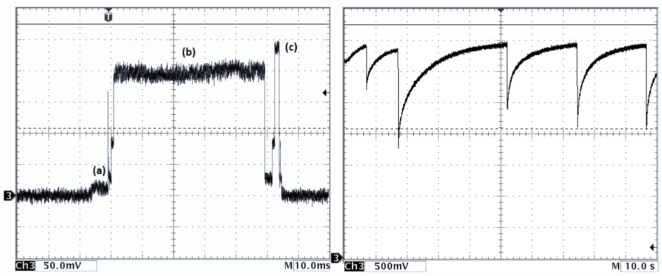


Fig. 7. The current profile of a typical duty cycle (left) demonstrates the current required for processing a packet (a), waiting for a beacon (b) and transmitting the packet (c). The voltage across the energy harvester (right) demonstrates a typical series of duty cycles.

performed first and the authentication code is computed on the cipher-text.

F. *Energy Awareness*

To incorporate energy awareness the voltage of the energy buffer is connected to the Analog-to-Digital Converter (ADC) of the microcontroller. Before turning the radio on, in both communication routines (*Send* and *Receive*), the firmware measures the voltage of the buffer and proceeds only if its value is above a configurable threshold. This mechanism dynamically alters the duty cycles in such a way that the radio is never switched on unless there is sufficient available energy.

G. *Packet Format*

Both beacons and data packets have an 8-bit options field in their header (OPT). The options field is a bitmap that specifies how each packet should be handled by its receiver. The two least significant bits specify the type of the packet. The next two least significant bits specify the security mode for the specific packet. The fourth least significant bit indicates an acknowledgement. The fifth least significant bit is defining the priority class of the packet. The two most significant bits in the options are reserved for future extensions. The payload of a beacon or an ABR frame consists of a 1-byte field that specifies the layer, which is used to assess the suitability of the beacon or the need for a backoff respectively. The payload of a data packet is 20 bytes in total, and contains information such as the identification number of the node and the sequence number of the data packet. In case authentication is enabled, packets also have a 4-byte footer that includes the message authentication code. Fig. 6 summarizes the packet format.

V. Protocol Evaluation

This section experimentally evaluates ODMAC in a practical testbed. The experiments focus on the link layer.

A. *Current Profile*

The current profile of a typical sender cycle, in which authentication and encryption are both enabled, is shown in Fig. 7 (left). Specifically, the figure shows the voltage drop across a 10 Ω shunt resistor, connected between the load and the power source. In the figure, one can clearly notice the time the node is listening for a beacon, which follows some initial MCU activity. After the beacon reception, it is possible to see the consumption spike related to the packet transmission. The example indicates that the main source of energy consumption comes from the time the radio spends in listening mode, waiting for a beacon. Hence, it highlights the significance of idle listening mitigation mechanisms, such as OF and AB.

B. *Integration of a Harvester*

The energy harvesting sensor nodes are powered by a photovoltaic (PV) panel connected to an off-the-shelf energy harvester. To implement energy-awareness, the positive side of the output capacitor of the energy harvester is connected to the ADC of the MCU.

The employed harvester is designed around factory specifications that support relatively short activity periods (*i.e.* radio activity cycles). Empirically, it is found that it can only support activity periods (listening and transmission) with duration of up to 150 ms, and at a frequency of up to 0.1 Hz. The experiments presented in this section are designed within these limits of the energy harvester.

The duty cycle of a sender is configured as follows. Every 10 seconds, a wake up event is initiated by comparing the voltage of the output capacitor of the energy harvester against a configurable threshold. This solution enables the dynamic adaptation of the duty cycle (and therefore the amount of packets sent) according to the amount of energy harvested, making the firmware energy aware. The voltage across the capacitor in a succession of packet transmissions is shown in Fig. 7 (right). Observe how the energy required for different transmissions varies with respect to the duration of the listening period, while the time for the capacitor to recover changes accordingly.

C. *Sustainability and Performance*

The first experiments focus on an ODMAC sender and demonstrate how the protocol adapts to the available energy to provide sustainable operation and prioritize throughput (*i.e.* sustainable throughput).

Specifically, a single transmitting node, u, which is part of a single link to a receiver node, is considered. From the perspective of u, the activity of the receiver is unknown. Two identical nodes, one with high and one with low beaconing duty cycle (periods of 33 ms and 66 ms respectively) act as receivers. Given this specific configuration and network topology, the average duration of an active period of u was

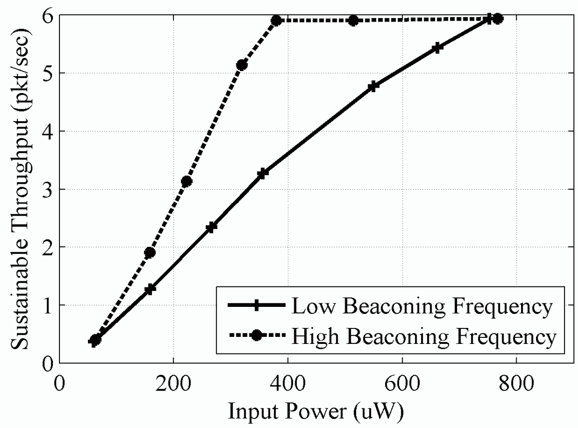


Fig. 8. Sustainable throughput for various harvesting power levels. The protocol adapts to receivers with different capabilities (high beaconing and low beaconing frequencies), while the adaptation of the sensing duty cycle effectively uses the available power for throughput.

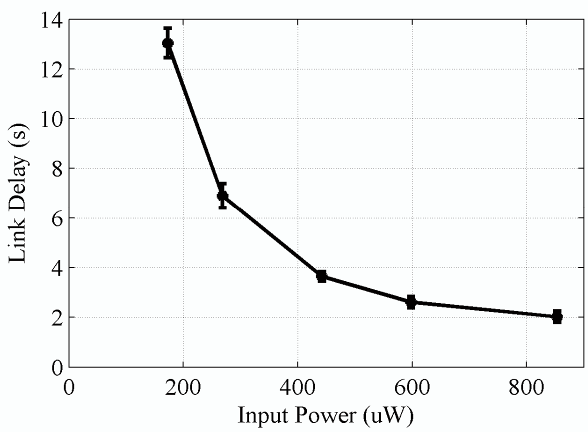


Fig. 9. Link delay for various harvesting power levels. The adaptation of the beaconing duty cycles effectively uses the available power to shorten the link delay.

measured at 43 ms with a standard deviation of 11 ms, in the case of the former receiver, and at 61 ms with a standard deviation of 23 ms, in the case of the latter.

In this setting, the energy harvester is exposed to different levels of constant input power, by adjusting the distance between a light source and the PV panels in a controlled environment, while the amount of packets that the node manages to successfully transmit in 30 minutes is measured. Fig. 8 shows the sustainable throughput of several experiments, each of which is initiated after the depletion of all the stored energy. In addition to demonstrating sustainable operation, the results show how the excess of harvested energy is used to improve the throughput of the application. Indeed, the throughput increases linearly with the amount of available energy, while it is constrained only by the maximum throughput supported by the energy harvester, *i.e.* 1 transmission every 10 seconds. The difference in throughput, in the cases of the two receivers, is attributed to listening time for a beacon reception, and thus to the beaconing duty cycle of the receiver. It demonstrates that the ODMAC transmitter not only adapts to the harvested energy but also to external conditions, such as the duty cycle of the receiver.

The next experiments focus on an ODMAC receiver and demonstrate how the protocol adapts to the available energy to provide sustainable operation in scenarios where link delay is the performance priority. Focusing again on a single link, node *u* is now the receiver that forwards traffic from a sender. The sender is programmed to transmit data traffic at random times (1 packet per minute on average). The receiver attempts to transmit a beacon every second. Similarly to the previous experiments, the beacon transmission occurs only if the voltage across the capacitor of the energy harvester is above a configurable threshold. In this setting, the link delay is measured as the duration of an activity period at a sender node. This approach disregards the propagation delay, which is negligible compared to the other delay sources. Fig. 9 shows the average link delay of several hundreds of transmissions at several constant input power levels. The error bars indicate the 90% confidence intervals. In each input power level, the link operated for several continuous hours, demonstrating the balanced energy budget of the receiver node. Furthermore, it can be observed that the link delay decreases exponentially with the amount of available energy, converging to the period of a beaconing cycle.

D. Evaluation of Opportunistic Forwarding

This section evaluates OF by comparing it to typical unicast forwarding. The next experiment assumes a simple topology with two forwarding options, *H* and *L*, similar to the one shown in Fig. 2; one with high (*H*) and one with low (*L*) beaconing duty cycle (33 ms and 66 ms respectively), emulating different energy harvesting conditions for each available receiver. The sender node, *u*, has three options: unicast to node *H*; unicast to node *L*; or use OF and opportunistically forward the traffic to the node that wakes up first. In this setting, the average idle listening per packet and the amount of packets served by each receiver are measured.

TABLE I
OPPORTUNISTIC VS UNICAST FORWARDING

	UNICAST TO H	UNICAST TO L	OF
Load Balancing (H - L)	100% - 0%	0% - 100%	62% - 38%
Avg. Idle Listening (ms)	22.5	39.1	17.8
Energy Efficiency Gain	1	0.58	1.26

Table I summarizes the results. The first row shows how the load is distributed between the receivers. As expected, when unicast forwarding is used, all traffic is relayed through the respective node. When OF is used, on the other hand, the load is distributed to both nodes, with 62% of the traffic being relayed by the node that has access to more energy, *H*, and 38% to the other. In addition to load balancing, OF is also reducing the average idle listening per packet; thus, improving the energy efficiency of the communication. As shown in the second row of the table, forwarding the traffic through node *H*, which is the natural choice of the routing protocol, yields an average idle listening per packet of 22.5 ms. When OF is used the average idle listening per packet of 17.8 ms, that corresponds to a 26% energy efficiency gain (third row) compared to unicasting to node *H*.

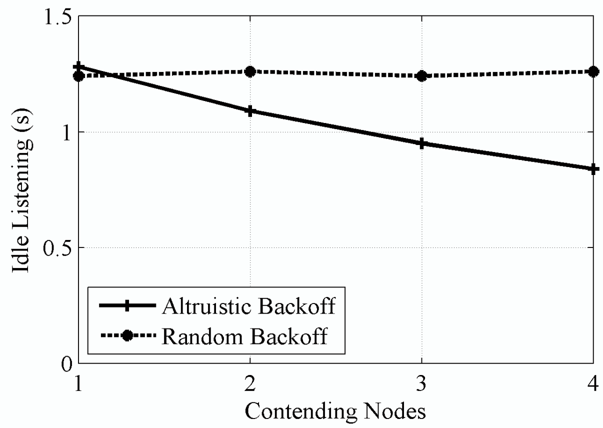


Fig. 10. Altruistic Backoff resolves collisions before the beacon reception, allowing the contending nodes to sleep early and save energy.

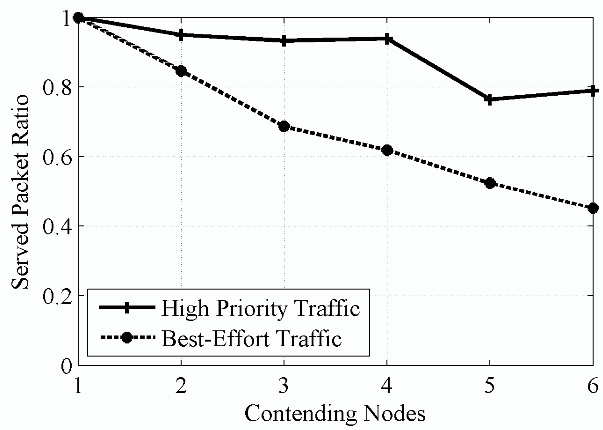


Fig. 11. As contention increases, best effort traffic backs off and urgent traffic gets the priority.

E. Evaluation of Altruistic Backoff

The evaluation continues with ODMAC's collision avoidance algorithm, AB. For the experiments presented in this section, a single-hop star network topology is considered. The network consists of multiple senders contending to transmit to a single receiver. The contending senders are placed physically close to each other and to the receiver, in order to mitigate channel errors. The beaconing period of the receiver is set to 1 second and the period of transmission attempts of the senders is randomized with an average of approximately 3 seconds.

The first experiment shows the effect of the number of contending nodes, including the scenario of no contention. Fig. 10 shows the average time each node spends on idle listening per transmission attempt when AB is used for collision avoidance, and in case of a typical RB scheme that resolves the collision after the beacon transmission. When there is no contention the two schemes perform similarly. As the number of contending nodes increase, for the case of RB, the average time spent in idle listening remains constant, being dominated by the time the node waits for a beacon. In the case of AB, on the other hand, idle listening decreases as the contention increases. It is shown in Fig. 7 that on a typical duty cycle, significantly more energy is spent in idle listening, rather than during the actual transmission. AB resolves collisions earlier than typical RB solutions, mitigating idle listening and increasing the overall energy-efficiency of the MAC layer.

The next experiment evaluates the traffic differentiation mechanism of the protocol. Each contending node generates data packets of *High Priority* with a probability of $p = 0.05$. In cycles where multiple contending nodes expect the same beacon to convey their traffic, the traffic differentiation scheme of ODMAC dictates which packets are prioritized. Fig. 11 shows the average ratio of the amount of data packets that are served over the total amount of generated packets, for each priority class. The results show that as the contention increases, a larger amount of *Best Effort* traffic backs off, giving priority to the *High Priority* traffic. As the contention increases, more *High Priority* packets back off as well. This happens because of the non-negligible probability of multiple *High Priority* data packets listening for the same beacon; a scenario where one packet can be served.

VI. Conclusions

Energy harvesting constitutes a promising solution for powering sensing and IoT-enabling technologies; yet, it introduces a spatial and temporal uncertainty in the MAC layer of communications that controls the duty cycles.

ODMAC is an energy-efficient MAC protocol that builds upon the receiver-initiated paradigm of asynchronous communication, and aims to tackle the challenges of energy harvesting communications. The individual control of the duty cycles, along with other novel protocol features, contributes to a sustainable performance and to the overall energy efficiency of the link. In particular, ODMAC is implemented for an off-the-shelf MCU and experimentally evaluated using off-the-shelf hardware. The evaluation demonstrates long-term sustainable operation, offering configuration parameters that allow the system designer to control the trade-off between sustainable throughput and link-delay. Moreover, the experiments show that opportunistic forwarding and altruistic backoff improve the energy efficiency of the link by up to 35% and 26% respectively compared to standard solutions.

References

[1] M. Chan, D. Estève, J.-Y. Fourniols, C. Escriba, and E. Campo, "Smart wearable systems: current status and future challenges," *Artif. Intell. Med.*, vol. 56, no. 3, pp. 137–56, 2012.

[2] M. Chan, D. Estève, C. Escriba, and E. Campo, "A review of smart homes - present state and future challenges," *Comput. Methods Programs Biomed.*, vol. 91, no. 1, pp. 55–81, 2008.

[3] J. Wang, Z. Zhang, B. Li, S. Lee, R. S. Sherratt, "An enhanced fall detection system for elderly person monitoring using consumer home networks," *IEEE Trans. Consum. Electron.*, vol. 60, no. 1, pp. 23–29, 2014.

[4] J. Winkley, P. Jiang, and W. Jiang, "Verity: an ambient assisted living platform," *IEEE Trans. Consum. Electron.*, vol. 58, no. 2, pp. 364–373, 2012.

[5] F. Bellido-Outeirino, J. Flores-Arias, F. Domingo-Perez, A. Gil-de-Castro, and A. Moreno-Munoz, "Building lighting automation through the integration of DALI with wireless sensor networks," *IEEE Trans. Consum. Electron.*, vol. 58, no. 1, pp. 47–52, 2012.

[6] Y.-W. Bai and M.-Y. Yang, "An improved design of a wireless keyboard powered by solar cells and a large capacitor," *IEEE Trans. Consum. Electron.*, vol. 54, no. 3, pp. 1355–1359, 2008.

[7] H. Jabbar, Y. Song, and T. Jeong, "RF energy harvesting system and circuits for charging of mobile devices," *IEEE Trans. Consum. Electron.*, vol. 56, no. 1, pp. 247–253, 2010.

[8] H.-Y. Shen, J.-Y. Lee, and T.-W. Chang, "Study of contactless inductive charging platform with core array structure for portable products," in *Int. Conf. on Consum. Electron., Commun. and Networks*, Xianning, China, pp. 756–759, 2011.

[9] Dan Dai and Jing Liu, "Human powered wireless charger for low-power mobile electronic devices," *IEEE Trans. Consum. Electron.*, vol. 58, no. 3, pp. 767–774, 2012.

[10] A. Bachir, M. Dohler, T. Watteyne, and K. K. Leung, "MAC Essentials for wireless sensor networks," *IEEE Commun. Surv. Tutorials*, vol. 12, no. 2, pp. 222–248, 2010.

[11] P. Huang, L. Xiao, S. Soltani, M. W. Mutka, and N. Xi, "The evolution of MAC protocols in wireless sensor networks: A Survey," *IEEE Commun. Surv. Tutorials*, vol. 15, no. 1, pp. 101–120, 2013.

[12] C. Cano, B. Bellalta, A. Sfairopoulou, and M. Oliver, "Low energy operation in WSNs: A survey of preamble sampling MAC protocols," *Comput. Networks*, vol. 55, no. 15, pp. 3351–3363, 2011.

[13] X. Fafoutis, A. Di Mauro, M. D. Vithanage, and N. Dragoni, "Receiver-initiated medium access control protocols for wireless sensor networks," *Comput. Networks*, vol. 76, pp. 55–74, 2015.

[14] W. Ye, J. Heidemann, and D. Estrin. "An energy-efficient MAC protocol for wireless sensor networks," in *Proc. 21st Ann. Joint Conf. IEEE Comput. and Commun. Soc.,* New York City, USA, vol. 3, pp. 1567–1576, 2002.

[15] T. Van Dam and K. Langendoen. "An adaptive energy-efficient MAC protocol for wireless sensor networks," in *Proc. 1st ACM Int. Conf. Embedded Networked Sensor Syst.*, Los Angeles, USA, pp. 171–180, 2003.

[16] K.-W. Chin, "Pairwise: a time hopping medium access control protocol for wireless sensor networks," *IEEE Trans. Consum. Electron.,* vol. 55, no. 4, pp. 1898–1906, 2009.

[17] J. Polastre, J. Hill, and D. Culler. "Versatile low power media access for wireless sensor networks," in *Proc. 2nd ACM Int. Conf. Embedded Networked Sensor Syst.*, Baltimore, USA, pp. 95–107, 2004.

[18] M. Buettner, G. V. Yee, E. Anderson, and R. Han. "X-MAC: A short preamble MAC protocol for duty-cycled wireless sensor networks," in *Proc. 4th ACM Int. Conf. Embedded Networked Sensor Syst.*, Boulder, USA, pp. 307–320, 2006.

[19] E.-Y. A. Lin, J. M. Rabaey, and A. Wolisz. "Power-efficient rendez-vous schemes for dense wireless sensor networks," in *Proc. IEEE Int. Conf. on Commun.*, Paris, France, vol. 7, pp. 3769–3776, 2004.

[20] Y. Sun, O. Gurewitz, and D. B. Johnson. "RI-MAC: A receiver-initiated asynchronous duty cycle MAC protocol for dynamic traffic loads in wireless sensor networks," in *Proc. 6th ACM Int. Conf. Embedded Networked Sensor Syst.*, Raleigh, USA, pp. 1–14, 2008.

[21] P. Dutta, S. Dawson-Haggerty, Y. Chen, C.-J. M. Liang, and A. Terzis. "A-MAC: A versatile and efficient receiver-initiated link layer for low-power wireless". *ACM Trans. Sensor Networks*, 8(4):30:1–30:29, 2012.

[22] X. Fafoutis and N. Dragoni, "Analytical comparison of MAC schemes for energy harvesting — wireless sensor networks," in *Proc. 9th Int. Conf. on Networked Sensing Syst.*, Antwerp, Belgium, pp. 1–6, 2012.

[23] S.-C. Kim, J.-H. Jeon, and H. J. Park. "QoS aware energy-efficient (QAEE) MAC protocol for energy harvesting wireless sensor networks," *Convergence and Hybrid Inform. Technology*, vol. 7425, pp. 41–48. Springer Berlin Heidelberg, 2012.

[24] S.-C. Kim and J.-H. Jeon, "EEPB-MAC: Energy efficient & priority-based MAC protocol for energy harvesting wireless sensor networks," *J. Korean Inst. Inf. Commun. Eng.*, vol. 17, no. 3, pp. 755–761, Mar. 2013.

[25] K. Nguyen, V.-H. Nguyen, D.-D. Le, Y. Ji, D. A. Duong, and S. Yamada, "ERI-MAC: An energy-harvested receiver-initiated MAC protocol for wireless sensor networks," *Int. J. of Distributed Sensor Networks*, vol. 2014, Article ID 514169, 2014.

[26] Q. Hu, Q. Tian, and Z. Tang, "RP-MAC: A passive MAC protocol with frame reordering for wireless sensor networks," *Int. J. Wirel. Inf. Networks*, vol. 20, no. 1, pp. 74–80, 2012.

[27] M. Y. Naderi, P. Nintanavongsa, and K. R. Chowdhury, "RF-MAC: A medium access control protocol for re-chargeable sensor networks powered by wireless energy harvesting," *IEEE Trans. Wirel. Commun.*, vol. 13, no. 7, pp. 3926–3937, 2014.

[28] Y. Sun, O. Gurewitz, S. Du, L. Tang, and D. B. Johnson. "ADB: An efficient multihop broadcast protocol based on asynchronous duty-cycling in wireless sensor networks," in *Proc. 7th ACM Int. Conf. on Embedded Networked Sensor Syst.,* Berkley, USA, pp. 43–56, 2009.

[29] G. P. Halkes, T. Van Dam, and K. G. Langendoen. "Comparing energy-saving MAC protocols for wireless sensor networks," *Mobile Networks and Applications*, 10(5):783–791, 2005.

[30] X. Fafoutis and N. Dragoni, "ODMAC□: An on-demand MAC protocol for energy harvesting - wireless sensor networks," in *Proc. 8th Int. ACM Symp. on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks*, Miami, USA, 2011.

[31] A. Kansal, J. Hsu, S. Zahedi, and M. B. Srivastava. "Power management in energy harvesting sensor networks," *ACM Trans. on Embedded Computing Syst.*, 6(4):32, 2007.

[32] C. M. Vigorito, D. Ganesan, and A. G. Barto. "Adaptive control of Duty cycling in energy-harvesting wireless sensor networks," in *Proc. 4th IEEE Conf. on Sensor, Mesh and Ad Hoc Commun. and Networks*, San Diego, USA, pp. 21–30, 2007.

[33] X. Fafoutis and N. Dragoni, "Adaptive media access control for energy harvesting — wireless sensor networks," in *Proc. 9th Int. Conf. on Networked Sensing Syst.*, Antwerp, Belgium, pp. 1–4, 2012.

[34] X. Fafoutis, C. Orfanidis and N. Dragoni. "Altruistic Backoff: Collision avoidance for receiver-initiated MAC protocols for wireless sensor networks," *Int. J. of Distributed Sensor Networks*, vol. 2014, Article 576401, Hindawi, 2014.

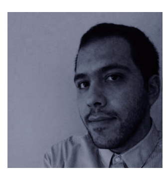
[35] C. Karlof, N. Sastry, and D. Wagner. "TinySec: A link layer security architecture for wireless sensor networks," in *Proc. 2nd ACM Int. Conf. on Embedded Networked Sensor Syst.,* Baltimore, USA, pp. 162–175, 2004.

[36] X. Fafoutis, A. Di Mauro, and N. Dragoni, "Sustainable medium access control: Implementation and evaluation of ODMAC," in Proc. *IEEE Int. Conf. on Commun. Workshops*, Budapest, Hungary, pp. 407–412, 2013.

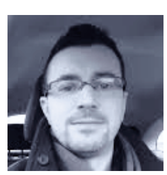
BIOGRAPHIES

Xenofon Fafoutis (S'09-M'14) received a PhD degree from the Embedded Systems Engineering section of the Technical University of Denmark in 2014; an MSc degree in Computer Science from the University of Crete in 2010; and a BSc in Informatics and Telecommunications from the University of Athens in 2007. He was also associated with the Foundation for Research and Technology - Hellas (FORTH). Since 2014, he is researcher with the University of Bristol. His interests lie in the fields of wireless embedded systems, wireless sensor networks and the internet of things.

Alessio Di Mauro received his PhD degree from the Department of Applied Mathematics and Computer Science of the Technical University of Denmark (DTU) and both his BSc and MSc in Computer Science from the Sapienza University of Rome, Italy. His main research topics are computer security, mobile security and distributed systems. Currently he is with Yubico, Sweden, where he works on multi-factor authentication and PKI.



Charalampos Orfanidis received his MSc in Computer Science and Engineering from the Department of DTU Compute of the Technical University of Denmark in 2015 and a BSc in Industrial Informatics Engineering from the Technological Educational Institute of Kavala in 2012. Right now, he is pursuing a PhD in Computer Science with focus on Computer Communication in the department of Information Technology of Uppsala University. His current research interests include wireless sensor networks, software defined networking and internet of things.



Nicola Dragoni obtained a MSc Degree and a PhD in computer science, respectively in 2002 and 2006, both at University of Bologna, Italy. He visited the Knowledge Media Institute at the Open University (UK) in 2004 and the MIT Center for Collective Intelligence (USA) in 2006. In 2007 and 2008 he was a postdoctoral research fellow at University of Trento, working on security for mobile systems. In 2009 he joined Technical University of Denmark (DTU) as assistant professor in security and distributed systems and was promoted to associate professor in 2011. Since 2014 he is also professor in computer engineering at Örebro University, Sweden.