computers

MDPI

Article

Symmetric-Key-Based Security for Multicast Communication in Wireless Sensor Networks †

Matthias Carlier [1,2,*], Kris Steenhaut [1] and An Braeken [2]

[1] Departement of Electronics and Informatics, Vrije Universiteit Brussel, 1050 Brussels, Belgium; ksteenha@etrovub.be

[2] Department of Engineering Technology, Vrije Universiteit Brussel, 1050 Brussels, Belgium; an.braeken@vub.be

* Correspondence: Matthias.Carlier@vub.be

† This paper is an extended version of the conference paper: Carlier, M.; Steenhaut, K.; Braeken, A. Symmetric-key Based Security for Multicast Communication in Wireless Sensor Networks. Presented at the 4th International Conference on Cloud Computing Technologies and Applications (CloudTech 2018), Brussels, Belgium, 26–28 November 2018.

Received: 21 February 2019; Accepted: 12 March 2019; Published: 19 March 2019

Abstract: This paper presents a new key management protocol for group-based communications in non-hierarchical wireless sensor networks (WSNs), applied on a recently proposed IP-based multicast protocol. Confidentiality, integrity, and authentication are established, using solely symmetric-key-based operations. The protocol features a cloud-based network multicast manager (NMM), which can create, control, and authenticate groups in the WSN, but is not able to derive the actual constructed group key. Three main phases are distinguished in the protocol. First, in the registration phase, the motes register to the group by sending a request to the NMM. Second, the members of the group calculate the shared group key in the key construction phase. For this phase, two different methods are tested. In the unicast approach, the key material is sent to each member individually using unicast messages, and in the multicast approach, a combination of Lagrange interpolation and a multicast packet are used. Finally, in the multicast communication phase, these keys are used to send confidential and authenticated messages. To investigate the impact of the proposed mechanisms on the WSN, the protocol was implemented in ContikiOS and simulated using COOJA, considering different group sizes and multi-hop communication. These simulations show that the multicast approach compared to the unicast approach results in significant smaller delays, is a bit more energy efficient, and requires more or less the same amount of memory for the code.

Keywords: multicast; security; mutual authentication; wireless sensor networks; lagrange interpolation; symmetric key

1. Introduction

Wireless Sensor Networks (WSNs) often employ low-power devices with low data rates, which communicate over lossy networks. Therefore, there is a need for special protocols that allow the same functionality as in traditional networks, but with these restrictions in mind. A functionality of traditional networks, which can improve the performance of WSNs is point-to-multipoint communication, also called multicast. In this communication paradigm, a particular device can reach a group of other devices in the network by sending a single message, containing a multicast address. The multicast enabled routing protocol will then disseminate the message to all devices in the group, copying it when needed.

To enable secure multicast communication in WSNs, efficient group key distribution protocols need to be developed. Security in WSNs has been extensively researched and standardization efforts

have been made [1–3]. However, these standardizations often do not include mechanisms for key management, in particular not for group-based communication. Nevertheless, key management is essential in this whole procedure to send secure and authenticated messages.

Often it is considered that the key is pre-established before deployment. For instance, we also considered such assumption in [4], where we studied the impact of including secure and authenticated communication for a 6LoWPAN-based WSN, connected via an edge router to the central server. For this purpose, we implemented the AES-GCM mode in the different entities. We concluded that the impact of the security mechanisms into the global system is acceptable and its relative impact even decreases when the number of hops increases and/or the number of messages decreases. Also, the memory footprint of the global system is quite small and limited to 6% of the total.

Minimal security features to be established are confidentiality, integrity, and authentication. Efficient mechanisms are required, due to the limited bandwidth, processing power, storage capacities, and available power. Efficiency implies scalability and adjustability [5]. Scalability in the sense that the key management protocol is able to include additional nodes in a secure manner during the network's lifetime. Adjustability implies a proper mechanism to deal with network condition changes.

We will use the architecture established in [6]. In this framework, group-based communication is organized through the integration of a network multicast manager (NMM), which allows for a reduced bandwidth usage with a minimal memory footprint, by integrating multicast groups in the publish/subscribe paradigm.

The remainder of the paper is organized as follows. Section 2 provides an overview about the related work. Section 3 explains the network architecture with some key definitions and assumptions used. In Section 4, we summarize the different phases of our scheme in normal mode. Sections 5 and 6 respectively describe the implementation details and the corresponding efficiency. Finally, Section 7 concludes the paper.

2. Related Work

In general, there are three main approaches for key management in WSNs, based on symmetric-key cryptography, public key cryptography, and hybrid. Especially in a hierarchical architecture, it can make sense to use a hybrid approach, in which the most computational heavy operations are performed by a powered mote. In this case, authentication and integrity are often obtained by digital signatures. However, as there is a huge performance difference between symmetric-key and public key cryptography, it is still interesting to look at symmetric-key-based solutions, due to the limited energy, communication bandwidth, and computational capacities of the sensor nodes.

Several proposals for key management using a symmetric-key-based approach in WSNs have been published. Depending on the assumed topology, these proposals can be divided in two categories: a hierarchical and non-hierarchical (or distributed) architecture of the nodes [7]. Most proposals for symmetric-key-based key management protocols assume a hierarchical network, in which the nodes have a predefined position in the network.

Within the domain of symmetric-key-based, authenticated key management protocols for hierarchical networks, the Localized Encryption and Authentication Protocol (LEAP) [8] is the most complete one. LEAP describes procedures to derive keys for the most prevalent communication scenarios (being between two cluster nodes (CNs)), a group key for all nodes in a cluster and a network key shared by all nodes.

References [9,10] present randomized approaches for the key management in hierarchical networks, which have no guarantee on successful key establishment but limit the impact of a compromised node. Both protocols have large storage requirements for each of the nodes in the network.

Furthermore, partial solutions for authenticated symmetric-key-based protocols are described in [11–14]. In [11], a method using one-way functions to derive an authenticated pairwise key between a CN and a cluster head (CH) is explained. Yet, group keys are not derived in this paper. In [12], each node receives an evaluation of a point on a predefined bivariate polynomial of the base station

(BS), dependent on its position in the sensor topology. Lagrange interpolation (LI) is used for the computation of the group key. However, this still requires a lot of unicast communications for the distribution of the group key to the individual members. Moreover, this paper does not derive a method for using this group key in an authenticated way. Finally, the key management in [13,14] is based on a generator matrix, predefined at the BS. The computation of the group key requires the involvement of the BS and cannot be performed by the CH alone. Again, these papers do not describe a mechanism to guarantee authenticated usage of the group key. In addition, the protocols from [11–14] are restricted to star networks, and thus no mechanism is described to compute a shared key between two CNs of the same cluster.

Key distribution solutions for hierarchical network structures are often more efficient, but less flexible as it is not possible to create custom groups, because the supported groups depend on the structure of the network. Therefore, solutions for non-hierarchical architectures are required. Key management schemes of [15–17] describe this type of network architecture. However, the schemes of [16,17] are using asymmetric-key-based methods as the basis of their key establishment system. The scheme in [16] is based on the Merkle identity tree. With this scheme, the direct and secure communications between any subgroup can be implemented and the delay of communication is decreased, making the scheme more suitable for WSNs. In [17], a building method of a key tree is proposed to reduce computation and storage overhead on every sensor node. In [15] a robot-assisted network bootstrapping technique is proposed. It focuses on scheme efficiency and supports various multicast group semantics. To the best of our knowledge, no symmetric-key-based method for group key management in distributed WSNs has been described in literature.

3. System Model and Assumptions

3.1. Setting

Our network topology is shown in Figure 1. We consider a WSN with nodes N_i for $(i = 1, \ldots, m)$ and a border router (BR), which connects the WSN to the rest of the network. An NMM, as proposed in [6], is responsible for the management of the multicast groups. This NMM is considered to be more powerful and takes the role of a network gateway, often positioned in the cloud. Nodes contact the NMM and share their capabilities and interests. Based on the information given by the nodes, the NMM generates a multicast address for each match and shares the required information with the publisher, one of the members of the group, chosen by the NMM. The publisher will then be able to construct a group key with the other subscribers S_s for $(s = 1, \ldots, n)$ of the multicast group. The NMM will not be a member of the group, but is only responsible for the initialization and verification of the authentication. There can be any number of intermediate motes (I) between the different subscribers of the same multicast group. These intermediate motes are not participating in the security protocol, resulting in a protocol that is independent of the underlying network structure.

To guarantee that the NMM is not able to understand the multicast communication, an external trusted third party (TTP) will be responsible for the generation and distribution of key material among the nodes and the NMM. Please note that we consider an honest but curious NMM: the NMM will perform all the required steps, but might be interested in following the communication for its own purposes (e.g., selling of data to other parties) and thus should not be able to derive the group key, established by the publisher of the multicast group.

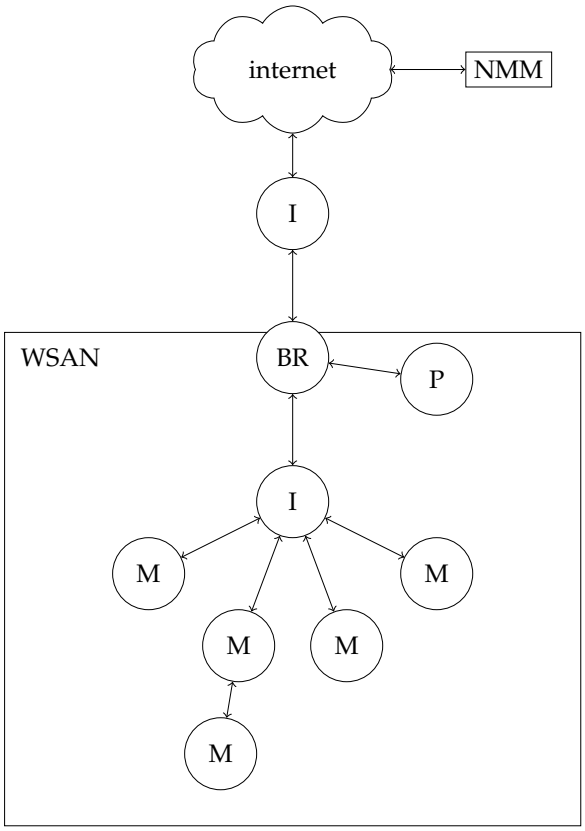

Figure 1. Example topology with the network multicast manager (NMM), a border router (BR), a publisher (P), intermediate motes (I) and members of the multicast group (M).

3.2. Multicast Engine

As the multicast engine, Bidirectional Multicast RPL Forwarding (BMRF) [18] is used. BMRF combines the best features of RPL multicast on the one hand and those of Stateless Multicast RPL Forwarding (SMRF) on the other hand, with a minimal increase in memory usage. As most important new functionalities, BMRF includes bidirectionality (the sources of multicast traffic can be located inside the network), dynamic group registration and the ability to have multiple senders (multipoint-to-multipoint).

An additional feature is that BMRF can offer a choice between Link Layer (LL) broadcast and LL unicast for which the decision for each mote is based on a threshold, mainly determined by the mote's number of interested children and the CCR. When BMRF is in unicast mode (and thus always uses LL unicasts to forward a multicast packet, independent of the threshold), BMRF is a well-designed and well-functioning implementation of RPL unicast with some additional features. When BMRF is in broadcast mode (and thus uses LL broadcast to forward multicast packets), BMRF behaves similar to SMRF, but adds some useful extensions.

The performance of the different modes of BMRF was tested in [19]. Since for our situation reliability is the key performance metric, BMRF will be used in unicast mode, which offers very good reliability and energy consumption on Radio Duty Cycling (RDC) enabled networks.

3.3. Notations

We represent a cryptographic hash function by H, which is a function that maps an input of any length to a fixed length output, with the following characteristics [20]:

- The mapping is deterministic, meaning that a given input will always result in the same output.
- It is infeasible to perform the inverse operation.
- It is infeasible to find more than one input that will result in the same output.

- A small change in the input should result in a completely different output.

The symmetric-key encryption operation of a message m under a key K to obtain the ciphertext c is denoted as $c = E_K(m)$, and the corresponding decryption operation as $m = D_K(c)$. Furthermore, the concatenation of values m_1 and m_2 is denoted by $m_1 \| m_2$ and the XOR operation by $m_1 \oplus m_2$.

3.4. Attack Model

The attackers may come from inside or outside the network. They can eavesdrop on the traffic, inject new messages, replay and change old messages, or spoof other identities.

As already mentioned above, we assume the TTP as completely trustworthy and the NMM as honest but curious. We further assume that the NMM does not collaborate with malicious nodes. Moreover, we assume that the security related information at the nodes and at the NMM is stored in tamper resistant hardware, which is currently very common and available at a reasonable price.

We do not discuss the mechanisms to detect malbehavior of a node (e.g., by storing trust tables in each node) and we refer to the literature on intrusion detection mechanisms [21] and attestation mechanisms [22] to detect abnormal behavior of a compromised node.

4. Security Scheme

We distinguish five different phases in our security scheme. First, there is the key distribution phase, in which the TTP distributes the secret key material to the different participants. Second, in the registration phase, the nodes share their capabilities and interests with the NMM. Third, when the multicast groups are defined by the NMM, during the group key construction phase, the NMM shares the required information with the publisher P, allowing the construction of the group key. Fourth, in the multicast communication phase, the publisher securely communicates data to the subscribed members. Finally, there is the group key update phase, in which the process is discussed when a node leaves or joins a multicast group.

4.1. Key Distribution Phase

In the first phase, the key distribution phase, the TTP selects three master secret values $x, y, z \in F_2^q$ and generates two different types of security related information. The first type is for the nodes N_i with $i = 1, \ldots, m$ in the network and the second type is for the NMM. Denote the identity of N_i by ID_i and NMM by ID_{NMM}.

For each N_i, the TTP executes the following computations:

$$\begin{aligned} K_i &= H(ID_i \| x \| y \| z) \\ A_i &= H(ID_i \| x) \\ B_i &= H(y) \oplus A_i \\ H(A_i) &= H(H(ID_i \| x)) \end{aligned}$$

The values $K_i, B_i, H(A_i), H(x), ID_{NMM}$ are sent over a secure channel (for instance by pre-storage) to each node N_i. Here, B_i replaces the identity of the node for the outside world. The parameter $H(A_i)$ is used to authenticate its identity with the NMM, and $H(x)$ for the authentication of the NMM with the node. The value K_i represents a common shared secret key with the TTP and is used to remotely update the security material by the TTP at later stages.

The security related information for the NMM is limited to the two parameters $H(ID_{NMM} \| H(x))$ and $H(y)$, which are also securely sent by the TTP to the NMM. Please note that the scheme is constructed in such a way that multiple NMMs can be considered. Only the identities of the different NMMs should then be communicated to the nodes. For ease in notation, we restrict the explanation to a situation involving only one NMM.

4.2. Registration Phase

In the registration phase, the legitimate nodes, being the nodes with security material generated by the TTP, contact the NMM to construct a common shared secret key. To do so, the N_i first selects a random value R_1 at timestamp T_1 and computes the following two values M_1 and M_2.

$$\begin{aligned} M_1 &= R_1 \oplus H(A_i) \\ M_2 &= H(H(ID_{NMM} \| H(x)) \| H(A_i) \| R_1 \| T_1) \end{aligned}$$

The message (B_i, M_1, M_2, T_1) is sent to the NMM.

Upon arrival, the NMM verifies whether the current timestamp T_2 is sufficiently close to T_1. If so, the NMM derives the random value R_1 using its stored secret $H(y)$.

$$\begin{aligned} A_i &= H(y) \oplus B_i \\ R_1 &= M_1 \oplus H(A_i) \end{aligned}$$

To check the integrity of R_1, the NMM will substitute the parameters $R_1, T_1, H(A_i)$ together with the secret key material $H(ID_{NMM} \| H(x))$ into the hash value $H(H(ID_{NMM} \| H(x)) \| H(A_i) \| R_1 \| T_1)$. If the result is equal to the received M_2, the NMM continues the process. Otherwise, the communication is interrupted. Next, the NMM chooses a second random value R_2 and computes

$$\begin{aligned} k_i &= H(R_1 \| R_2 \| T_1 \| T_2 \| B_i \| ID_{NMM}) \\ M_3 &= R_2 \oplus B_i \oplus H(A_i) \\ M_4 &= H(k_i \| H(ID_{NMM} \| H(x)) \| T_1 \| T_2) \end{aligned}$$

Please note that k_i represents a common shared secret key between N_i and the NMM. The message (T_2, M_3, M_4) is sent from the NMM to the node N_i.

After receiving this message, the node first verifies the freshness of the communication, i.e., if the current timestamp T_3 is not much larger than T_2. If so, it continues and derives R_2 from M_3. Using this value, the key k_i can be computed and its integrity verified with the parameter M_4.

As a result, both N_i and the NMM possess a mutual authenticated shared key, which is used to communicate the interests and capabilities of N_i to the NMM.

A summary of the registration phase can be found in Figure 2.

4.3. Group Key Construction Phase

Based on the information received after the registration phase, the multicast groups are created. For each group, a multicast address ID_G is generated. The NMM communicates the address ID_G to each subscriber, denoted by $S_s, s = 1, \ldots, n$ (with corresponding key material A_s, B_s). To the publisher of the group, denoted by P (with corresponding key material A_p, B_p), key related information of the subscribers in its group is sent using their common shared key k_p. This key related information for each subscriber equals to $k_{ps} = H(ID_G \| k_s)$ and is combined with identity B_s, for all S_s with $s = 1, \ldots, n$ in the multicast group. Consequently, the message

$$E_{k_p}(B_1, k_{p1}, \ldots, B_n, k_{pn})$$

is sent to the publisher. After decryption of this message, P possesses a common shared key with each of its subscribers. This information allows P to share a randomly chosen group key k_g for the multicast communication.

Here, P first chooses a random value k_g, which will serve as the group key to be shared with the other members in a multicast communication message. It also selects a random value h_0 for the construction of a one-way key chain [23] to be used for authentication purposes. Consequently, w

consecutive hashes are taken, i.e., $h_1 = H(h_0), h_2 = H(H(h_0)) = H^2(h_0), \ldots, h_w = H^w(h_0)$. The value h_w will also be shared with the members.

Since P has a common key with each member of the group, two methods can be used to distribute the group key: the multicast and unicast approach. In the multicast approach, a combination of LI, a classical technique from the secret sharing approach, with the communication of an IP multicast packet is applied. In the unicast approach, multiple IP unicast packets are sent to share the encrypted group key to each of the individual nodes. Both approaches are discussed in more detail below.

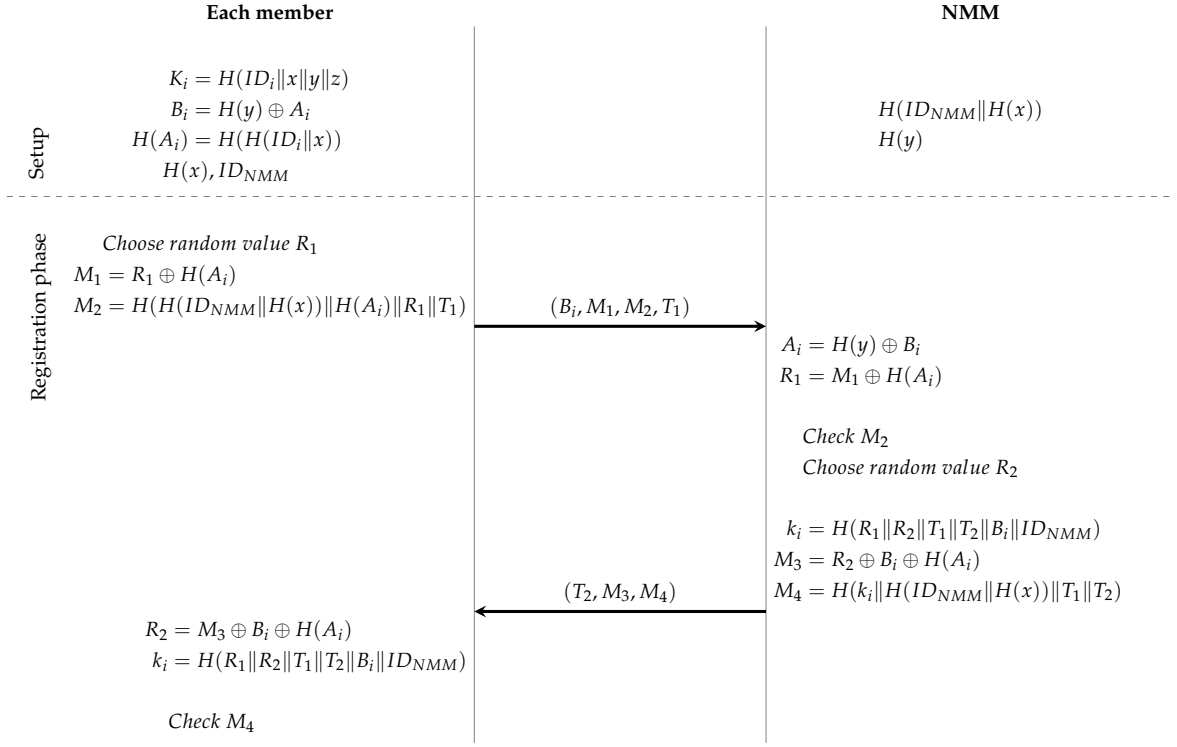

Figure 2. Summary of the registration phase.

4.3.1. Multicast Approach

Using LI through the n points $(x_s, y_s) = (B_s, k_{ps})$ with $s = 1, \ldots, n$, together with the point containing the group key $(0, k_g)$, a polynomial of degree n can be constructed. Next, n other points (v, V_v) with $v = 1, \ldots, n$ on this polynomial are derived. Finally, P sends the following message using the multicast address ID_g:

$$T_g, h_w, V_1 \oplus H(x), \ldots, V_n \oplus H(x),$$
$$H(ID_G \| T_g \| V_1 \| \ldots \| V_n \| k_g \| h_w)$$

with T_g the current timestamp.

When this message arrives to all subscribers of the multicast group ID_G, each subscriber $S_s, s = 1, \ldots, n$ constructs the polynomial through the $n + 1$ points with coordinates (v, V_v), (for $v = 1, \ldots, n$) and its own point $(B_s, H(ID_G \| k_s))$. The group key k_g is derived as the intersection point of this polynomial with the X-axis. Its integrity is confirmed using the hash value of the message.

4.3.2. Unicast Approach

In the unicast approach, P shares the random value k_g through the following message with all the subscribers $(S_s, s = 1, \ldots, n)$:

$$T_g, h_w, E_{k_s}(k_g \oplus H(x)), H(ID_G \| T_g \| k_g \| h_w)$$

Each subscriber can decrypt this message using its secret key. After XORing the obtained result with the stored value $H(x)$, the subscriber retrieves the group key.

To complete the process (for both the unicast and multicast approaches), each subscriber $S_s, (s = 1, \ldots, n)$ must send an ACK of reception to the publisher P.

$$B_s, E_{k_{ps}}(H(k_g, T_g, h_w), H(T_1 \| T_2 \| h_w \| k_s))$$

If after decryption of the second part of the message, the hash value of the original timestamp T_g and hash value h_w are recovered, the publisher stores the rest of the ciphertext $H(T_g \| h_w \| k_s)$. If all acknowledgements have been collected, P computes

$$\bigoplus_{s=1}^{n} H(T_1 \| T_2 \| h_w \| k_s)$$

and sends this result, together with the value h_w, to the NMM. The NMM can compute the individual values of this sum and thus it can verify whether the resulting sum corresponds with the received value. If so, the NMM approves the multicast group and sends a confirmation using unicast communication to each of the members. If not, the individual messages $(B_s, H(T_g \| h_w \| k_s))$ should be transmitted to the NMM to identify the potential source of the problem. The members of the multicast group now store k_g, h_w.

A summary of the group key construction phase can be found in Figures 3 and 4, for the multicast and unicast approaches, respectively.

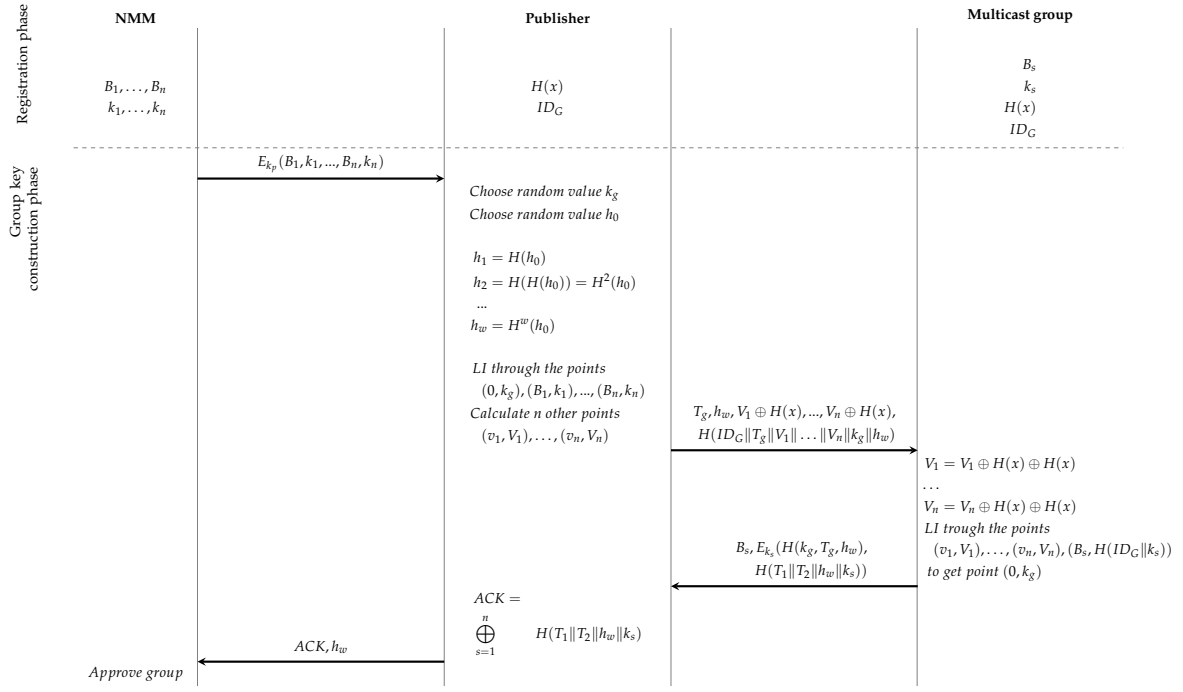

Figure 3. Summary of the multicast approach in the group key construction phase.

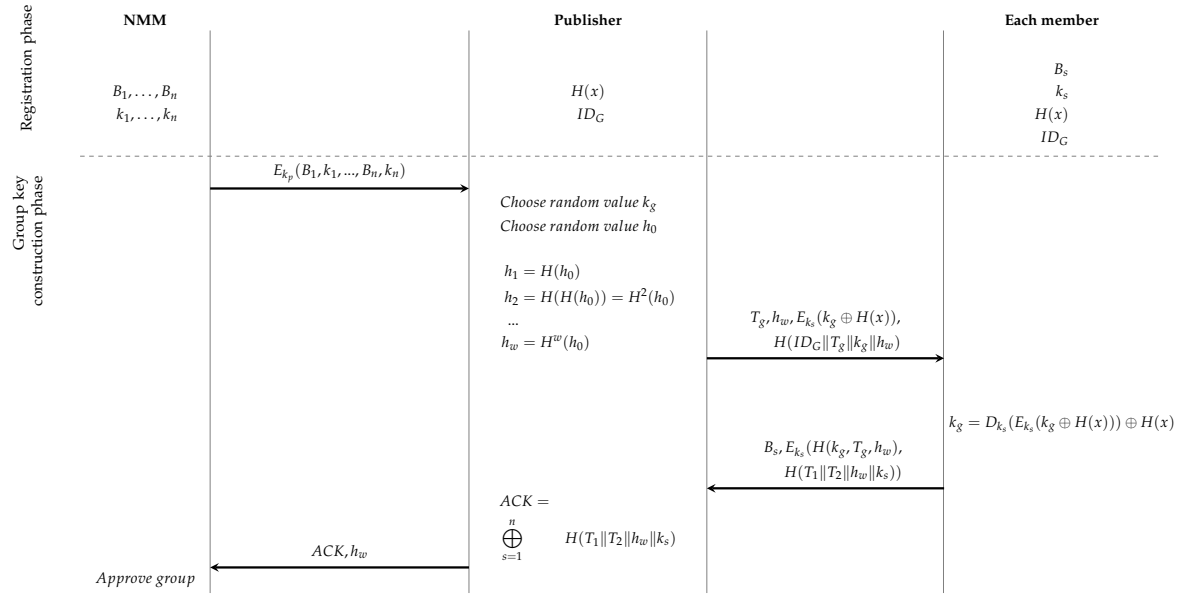

Figure 4. Summary of the unicast approach in the group key construction phase.

4.4. Multicast Communication Phase

Since all members of the multicast group now share the group key k_g, confidentiality is easily realized. Authentication of the messages from the publisher is obtained by using the one-way key chain on the side of the publisher and the stored end value h_w on the side of the subscriber. Consequently, the publisher can now securely submit the message M in an authenticated way to its subscribers as follows:

$$T_c, E_{k_g}(M, h_{w-1}), H(T_c \| M \| h_{w-1}).$$

Here, T_c denotes the current timestamp and h_{w-1} the hash value prior to $h(w)$. Upon receiving this message, the subscribers can decrypt the ciphertext and verify whether $h_w = H(h_{w-1})$. If so, the message is authenticated and h_{w-1} is stored instead of h_w.

5. Implementation

To verify the validity of the proposed security scheme, it was implemented in ContikiOS and simulated in COOJA (using COOJA motes). BMRF is used as multicast engine, with all the default values of Contiki 2.7, except for the changes listed below. Since a missing packet in the key distribution phase can result in the entire group being invalidated by the NMM, reliability is important. Therefore, BMRF is configured to use several LL unicast packets to forward IP multicast packets to its interested children. As COOJA motes do not support any RDC protocols, nullRDC is used. The following points are taken into account during the implementation.

- To allow larger packets to be sent, the 6LoWPAN queue buffer and the uIP/IP buffer have been increased.
- Please note that the largest variable type that can be used in ContikiOS is a double, which is 8 bytes long. To get a 16-byte key, two Lagrange polynomials need to be calculated and transmitted.
- When doing floating point calculations with potentially large numbers, the precision of the result can be limited. Since we are using LI with points that are dependent on the calculated key values, results can get very large. To assure each member works with the same result, they must be rounded.
- The calculations for the key material are performed using two temporary buffers to store intermediate values. The size of these buffers can be set in the configuration files. The minimum

required size for these buffers depends on the used approach (unicast or multicast) and on the maximum number of members of the multicast group.

- For encryption, the AES128 library, written by Texas Instruments [24] is used. For hashing, a sha256 library [25] is used.

6. Efficiency

Let us compare the two approaches (unicast and multicast) to distribute the group key to all the members in terms of delay, energy consumption, storage requirements, and packet fragmentation. The storage requirements and packet fragmentation are analyzed using both simulations and theory. The used topology is shown in Figure 1. In this topology, the NMM is in the cloud. The publisher is a child of the BR. Each member of the multicast group (M) is located at a certain number of hops from the BR (varying in different tests), with a particular number of intermediate motes (I) in between. Other configuration parameters can be found in Table 1.

Table 1. Simulation configuration parameters.

OS and Simulator	Contiki (version 2.7) and COOJA
Motes	COOJA motes
Radio Medium	Unit disk graph medium (UDGM)
Ranges	Transmit: 50 m Interference: 50 m
Topology	See Figure 1
Number of members	2 to 5
Number of intermediate motes	0 to 3 (hops (h): 1 to 4)
PHY and MAC	IEEE 802.15.4 and CSMA
Iterations	10 for each configuration
RNG seeds	New seed each iteration
Traffic	According to protocol
RDC protocols	NullRDC

6.1. Delay

The registration phase for both the unicast and multicast methods is identical, as can be observed in Figure 5. The time required to finish the registration phase only increases with the number of motes in the multicast group and with the number of hops between the motes and the NMM.

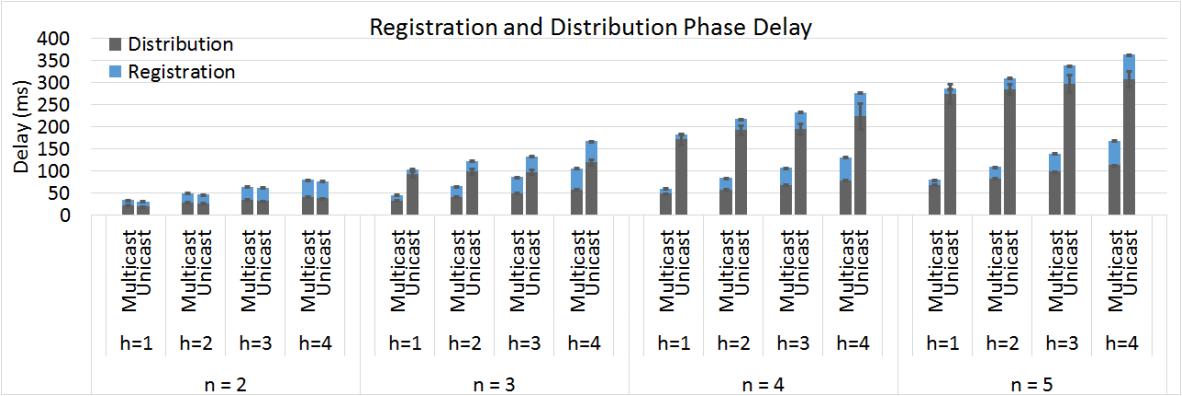

Figure 5. Average registration and distribution phase delay. The number of members in the group is denoted as n and the number of hops between the publisher and the members is denoted as h.

In contrast, Figure 5 also shows that the distribution phase of the two approaches is not the same. Whereas both approaches are equally dependent on the number of hops between the publisher and the other members, we observe that the method using unicast transmissions is less dependent on the number of motes in the multicast group. The latter can be explained by the fact that the multicast method does not duplicate its transmission into different packets until the last hop, while the transmission of the unicast method is duplicated by the publisher. This means that the unicast packet for the last mote in the group cannot be sent until all previous unicast packets were sent, resulting in additional delays.

6.2. Packet Fragmentation

Taking into account that we use the 6LoWPAN protocol, packets with a length of 70 bytes or less are never fragmented. When a packet is larger and fragmentation occurs, the number of fragments required is given by $1 + (length - 64)/72$, since the first fragment is maximally 64 bytes long and all subsequent fragments are maximally 72 bytes.

Only the packets which are sent in the key construction phase (depending on the number of members in the group) possibly require fragmentation. These are the packet sent by the NMM to the publisher and the packet sent from the publisher to all the other nodes. Please note that using the method without LI will not result in fragmentation for the packet from the publisher, since it is always 32 bytes long, resulting in less LL frames being sent.

Table 2 indicates the size of these packets, depending on the number of group members, and the corresponding number of fragments. These numbers are confirmed by simulations.

Table 2. Packet length and number of fragments for the longest types of packets.

	NMM to Publisher		**Publisher to Members**	
Members	**Packet Size (bytes)**	**Fragments**	**Packet Size (bytes)**	**Fragments**
2	64	1	68	2
3	96	2	84	2
4	128	2	100	2
5	160	3	116	2
6	192	3	132	2
7	224	4	148	3
8	256	4	164	3
9	288	5	180	3
10	320	5	196	3

6.3. Energy Consumption

The number of packets sent by the key distribution protocol mainly determines its energy consumption. In this section, we are going to calculate the number of packets sent on the application layer, when there are n members in the group.

In the unicast approach, each mote needs to send one packet to the NMM to register to the group, resulting in n packets. The NMM sends the information of each mote to the publisher in a single packet. Next, the publisher sends a packet to each other mote in the group, resulting in $n - 1$ packets. Finally, each mote transmits an acknowledgement to the publisher, who sends the final acknowledgement to the NMM. The average number of packets sent by the motes in the group equals $(3n - 1)/n$. Please note that we are not counting the packet sent by the NMM, since the NMM is in the cloud and thus not part of the WSN.

In the multicast approach, the registration phase and acknowledgement are the same as for the unicast approach, resulting in $2 \cdot n$ packets. However, the distribution phase only results in one multicast packet being sent. Hence, the average number of packets sent by the motes in the group

equals $(2n+1)/n$. Again, the packet from the NMM is not included. Our simulation results, shown in Figure 6, confirm these formulas for both the unicast and the multicast approach.

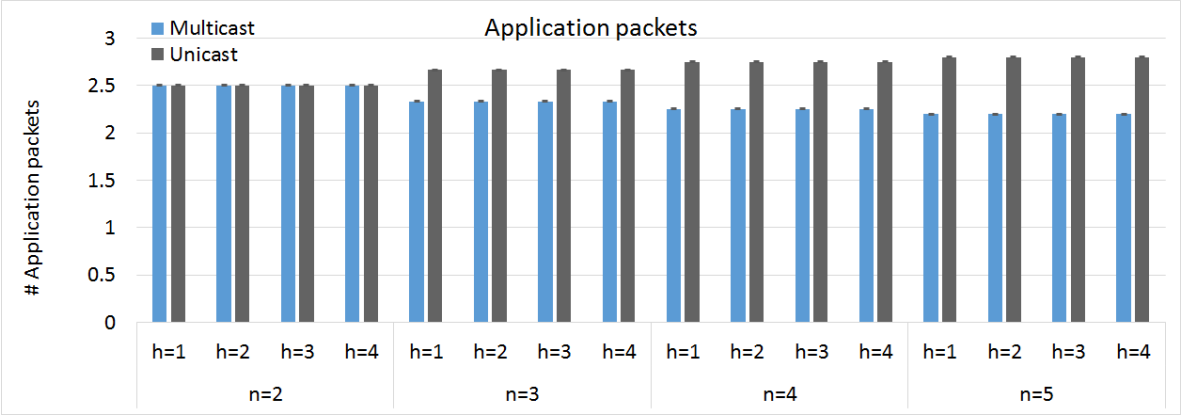

Figure 6. Average number of application packets sent since the start of the registration phase of the protocol. The number of members in the group is denoted as n and the number of hops between the publisher and the members is denoted as h.

Of course, the number of frames sent by the LL is more important to estimate the energy consumption. It is possible for a single packet sent on the application layer to result in multiple frames being sent on the LL in the entire network, because of fragmentation and the need to forward packets multiple hops and to multiple children. In Figure 7 we observe that the number of hops between the members and the publisher has a bigger influence on the number of sent frames then the number of members in the group. There are less LL frames being sent for the multicast approach since multicast packets do not need to be duplicated until the last hop. The difference is mostly noticeable with increasing number of hops between P and the other members and with an increasing number of members in the multicast group. This difference is smaller than expected when compared with the number of sent packets on the application layer, because the larger multicast packet (containing the LI information) results in more fragmentation, as explained previously in Section 6.2. The simulations show that in our topology, after 4 hops with 5 members, this still results in an 8% advantage in terms of sent LL frames when using the multicast approach.

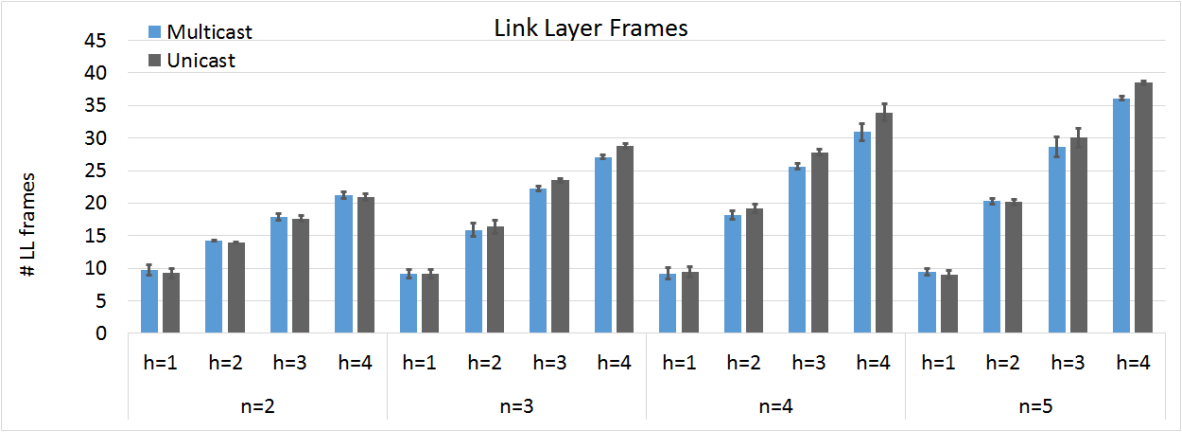

Figure 7. Average number of LL frames sent since the start of the registration phase of the protocol. The number of members in the group is denoted as n and the number of hops between the publisher and the members is denoted as h.

6.4. Storage Requirements

In our protocol, each mote must store four auxiliary parameters and two critical parameters of 16 bytes each. Critical parameters are those needed to be stored preferentially in tamper proof hardware and whose initial values are pre-installed. This key material allows the node to efficiently construct any type of keys, on the fly and ad-hoc. This is the same for both the unicast and multicast approaches.

Different buffers need to be reserved to calculate all required values and to send and receive all packets. In ContikiOS, a single buffer to send and receive packets is available, shared between all layers. The size of this buffer determines the largest packet size that can be sent/received by a mote. In our protocol, the largest packet that needs to be received is the packet from the NMM to the P, containing the identities and keys of all the motes in the group. This packet is $32 \cdot n$ bytes long, with n the number of members in the group.

To calculate all values, only two additional buffers are needed, since they can be reused. The largest value that needs to be calculated is the packet containing the group key information, which needs a $36 + 16 \cdot n$ byte buffer for the multicast approach and a 56 byte buffer for the unicast approach.

After the group has been approved by the NMM, each mote needs to store two 16-byte values (the group key k_g and the one-way key chain h_w) and the group ID.

This results in the memory consumption found in Table 3. Both implementations fit in the Zolertia RE-Mote platform [26], a popular development platform for WSANs, which has 512 KB flash and 32 KB ROM. In addition, for each packet being sent in the group, the members need to store the 16 byte value from the one-way key chain. This results in an additional 1.6 kB (about 12%) RAM usage. However, the value can be calculated on the fly, to trade off memory usage with processing time.

Table 3. Memory consumption (compared with available memory in the Zolertia RE-Mote platform) of an application using BMRF and the proposed protocol using the different approaches.

	Unicast	**Multicast**
ROM/flash	18,7057 bytes (36.5%)	18,9473 bytes (37.0%)
RAM	13,428 bytes (42.0%)	13,512 bytes (42.2%)

7. Conclusions

In this paper, we proposed a highly efficient symmetric-key-based key management protocol for group-based communications in a distributed WSN with a cloud-based NMM. The protocol establishes confidentiality, integrity, and authentication for communication between an arbitrary group of motes in the network. In addition, the established group key is kept secret for the NMM.

Two different approaches are described to share the group keys with all members. In the unicast approach, each member receives the key using a unicast message, whereas in the multicast approach, a combination of LI and a single multicast message is used to disseminate the group key to all members.

Using simulations considering a different number of nodes in the group and a different number of hops from the publisher, we show that the multicast approach results in all situations in a significant smaller delay, compared to the unicast approach. When the number of nodes in the network is larger than three, less LL frames need to be sent over the network, resulting in better energy consumption behavior. The additional code size required for the multicast approach is negligible.

Author Contributions: Conceptualization, A.B. and M.C.; methodology, A.B., M.C. and K.S.; software, M.C.; validation, M.C., A.B. and K.S.; formal analysis, M.C.; writing—original draft preparation, M.C. and A.B.; writing—review and editing, M.C.; visualization, M.C.; funding acquisition, M.C.

Funding: This research has been supported by the Agency for Innovation by Science and Technology in Flanders (IWT).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Grover, J.; Sharma, S. Security issues in Wireless Sensor Network—A review. In Proceedings of the 2016 IEEE 5th International Conference on Reliability, Infocom Technologies and Optimization, ICRITO 2016: Trends and Future Directions, Noida, India, 7–9 September 2016; pp. 397–404. [CrossRef]
2. Hari, P.B.; Singh, S.N. Security issues in Wireless Sensor Networks: Current research and challenges. In Proceedings of the 2016 IEEE International Conference on Advances in Computing, Communication and Automation, Dehradun, India, 8–9 April 2016; pp. 1–6. [CrossRef]
3. Al Ameen, M.; Liu, J.; Kwak, K. Security and privacy issues in wireless sensor networks for healthcare. In *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering*; Springer: Berlin/Heidelberg, Germany, 2015; Volume 150, pp. 223–228. [CrossRef]
4. Carlier, M.; Steenhaut, K.; Braeken, A.; Smeets, R.; Mentens, N.; Aerts, K. Study on impact of adding security in a 6LoWPAN based network. In Proceedings of the 2015 IEEE Conference on Communications and NetworkSecurity, Florence, Italy, 28–30 September 2015; pp. 577–584. [CrossRef]
5. Xiao, Y.; Rayi, V.K.; Sun, B.; Du, X.; Hu, F.; Galloway, M. A survey of key management schemes in wireless sensor networks. *Comput. Commun.* **2007**, *30*, 2314–2341. [CrossRef]
6. Akkermans, S.; Bachiller, R.; Matthys, N.; Joosen, W.; Hughes, D.; Vučinić, M. Towards efficient publish-subscribe middleware in the IoT with IPv6 multicast. In Proceedings of the 2016 IEEE International Conference on Communications, Kuala Lumpur, Malaysia, 23–27 May 2016; pp. 1–6. [CrossRef]
7. Bala, S.; Sharma, G.; Verma, A. Classification of symmetric key management schemes for wireless sensor networks. *Int. J. Secur. Its Appl.* **2013**, *7*, 117–138.
8. Zhu, S.; Setia, S.; Jajodia, S. LEAP: Efficient security mechanisms for large-scale distributed sensor networks. *ACM Trans. Sensor Netw.* **2006**, *2*, 500–528. [CrossRef]
9. Durresi, A.; Bulusu, V.; Paruchuri, V.; Durresi, M.; Jain, R. WSN09-4: Key Distribution in Mobile Heterogeneous Sensor Networks. In Proceedings of the IEEE Globecom 2006, San Francisco, CA, USA, 27 November–1 December 2006; pp. 1–5. [CrossRef]
10. Hussain, S.; Kausar, F.; Masood, A. An efficient key distribution scheme for heterogeneous sensor networks. In Proceedings of the 2007 international conference on Wireless communications and mobile computing, Honolulu, HI, USA, 12–16 August 2007; ACM Press: New York, NY, USA, 2007; p. 388. [CrossRef]
11. Ou, G.; Huang, J.; Li, J. A key-chain based key management scheme for heterogeneous sensor network. In Proceedings of the 2010 IEEE International Conference on Information Theory and Information Security, Beijing, China, 17–19 December 2010; pp. 358–361. [CrossRef]
12. Zhang, Y.; Shen, Y.; Lee, S.; Setup, A.K.P.d. A Cluster-Based Group Key Management Scheme for Wireless Sensor Networks. In Proceedings of the 2010 12th International Asia-Pacific Web Conference, Busan, Korea, 6–8 April 2010; pp. 3–5. [CrossRef]
13. Li, L.; Wang, X. A High Security Dynamic Secret Key Management Scheme for Wireless Sensor Networks. In Proceedings of the 2010 Third International Symposium on Intelligent Information Technology and Security Informatics, Jinggangshan, China, 2–4 April 2010; pp. 507–510. [CrossRef]
14. Shnaikat, K.N.; Alqudah, A.A. Key Management Techniques In Wireless Sensor Networks. *Int. J. Netw. Secur. Appl.* **2014**. [CrossRef]
15. Ren, K.; Lou, W.; Zhu, B.; Jajodia, S. Secure and efficient multicast in wireless sensor networks allowing ad hoc group formation. *IEEE Trans. Veh. Technol.* **2009**, *58*, 2018–2029. [CrossRef]
16. Chen, L.Q.; Sun, C.F.; Zhu, Q.Y. A Novel Group Key Agreement Scheme for Wireless Sensor Networks Based on Merkle Identity Tree. *Adv. Mater. Res.* **2013**, *846–847*, 869–875. [CrossRef]
17. Yao, W.; Han, S.; Li, X. LKH++ Based Group Key Management Scheme for Wireless Sensor Network. *Wirel. Pers. Commun.* **2015**, *83*, 3057–3073. [CrossRef]
18. Gastón Lorente, G.; Lemmens, B.; Carlier, M.; Braeken, A.; Steenhaut, K. BMRF: Bidirectional Multicast RPL Forwarding. *Ad Hoc Netw.* **2017**, *54*, 69–84. [CrossRef]
19. Carlier, M.; Garcia Algora, C.M.; Braeken, A.; Steenhaut, K. Analysis of internet protocol based multicast on duty-cycled wireless sensor networks. *IEEE Sens. J.* **2018**, *18*, 4317–4327. [CrossRef]
20. Selvakumar, A.A.L.; Ganandhas, C.S. The evaluation report of sha-256 crypt analysis hash function. In Proceedings of the 2009 International Conference on Communication Software and Networks, Macau, China, 27–28 February 2009; pp. 588–592. [CrossRef]

21. Butun, I.; Morgera, S.D.; Sankar, R. A survey of intrusion detection systems in wireless sensor networks. *IEEE Commun. Surv. Tutor.* **2014**, *16*, 266–282. [CrossRef]
22. Steiner, R.V.; Lupu, E. Attestation in Wireless Sensor Networks. *ACM Comput. Surv.* **2016**, *49*, 1–31. [CrossRef]
23. Lamport, L. Password authentication with insecure communication. *Commun. ACM* **1981**, *24*, 770–772. [CrossRef]
24. AES-128 Advanced Encryption Standard | TI.com. Available online: http://www.ti.com/tool/AES-128 (accessed on 16 May 2018).
25. Sha256-Github. Available online: https://github.com/itszero/librvm-course-impl/blob/master/sha256.c (accessed on 16 May 2018).
26. Zolertia RE-Mote-Github. Available online: https://github.com/Zolertia/Resources/wiki/RE-Mote. (accessed on 12 May 2018).