IKM -- An Identity based Key Management Scheme for Heterogeneous Sensor Networks

Manel Boujelben
National school of engineers of Sfax, Tunisia
Email: manel.boujelben@ceslab.org

Habib Youssef
Institut Supérieur d'Informatique et des Technologies de Communication, Hammam Sousse, Tunisia
Email: habib.youssef@fsm.rnu.tn

Rania Mzid and Mohamed Abid
National school of engineers of Sfax, Tunisia
Email: rania.mzid@yahoo.fr, Mohamed.abid@enis.rnu.tn

***Abstract*— Wireless sensor networks applications are growing and so are their security needs. However, due to severe memory, computing and communication limitations, wireless sensor networks security presents tremendous challenges. Central to any security service, key management is a fundamental cryptographic primitive to provide other security operations. In this paper, we propose IKM, an identity based key management scheme designed for heterogeneous sensor networks. This scheme provides a high level of security as it is based on a variant of public key cryptography named pairing identity based cryptography. The IKM scheme supports the establishment of two types of keys, pairwise key to enable point to point communication between pairs of neighbouring nodes, and cluster key to make in-network processing feasible in each cluster of nodes. IKM also supports the addition of new nodes and re-keying mechanism. A security analysis is presented to prove the scheme resilience against several types of attacks especially the node compromise attack. We also perform an overhead analysis of the proposed scheme in terms of storage, communication, and computation requirements. To demonstrate the feasibility of this scheme, we present implementation and performance evaluation of the proposed scheme on Crossbow TelosB motes running TinyOS. The results indicate that it can be deployed efficiently in resource-constrained sensor networks that need a high level of security.**

***Index Terms*— Key management, Identity based cryptography, Sensor networks.**

I. Introduction

Wireless Sensor Networks (WSNs) are widely used in several applications such as military, healthcare, automotive, research, and so on. Many sensors based applications are often deployed in potentially adverse or even hostile environment. Therefore, they are dependent upon the secure operation of a sensor network, and suffer serious consequences if the network is compromised or disrupted. WSN security presents several challenges. First, communication is difficult to protect since it is realized over a wireless channel. Hence, adversaries can easily eavesdrop, intercept, inject, and alter transmitted data. Second, since sensor networks may be deployed in a variety of physically insecure environments, adversaries can steal nodes, recover their cryptographic material, and pose as authorized nodes in the network. Third, sensor networks are vulnerable to resource consumption attacks. Adversaries can repeatedly send packets to drain a node battery and waste network bandwidth. However, WSNs present inherent limitations. Sensor nodes are battery powered and equipped with integrated sensors, low data processing capabilities, and short-range radio communications. This makes security a challenging problem.

All security mechanisms are usually orchestrated around robust encryption and authentication algorithms. Keys for encryption and authentication purposes must be agreed upon by communicating nodes. In resource constrained WSNs, achieving such key agreement is nontrivial. So, besides lightweight ciphers, efficient key management mechanisms are needed.

Previous research on key management in sensor networks mainly considers homogeneous sensor networks organised in a flat architecture [1][2]. Following this design, homogeneous sensors are dispersed in the region of study and each sensor is programmed to perform all possible application tasks and to communicate with its neighbours via Adhoc networking. However, despite the efficiency and the simplicity of flat Adhoc sensor network applications, research has demonstrated the limitations of the flat topology in terms of performance and scalability [3][4]. Recently, hierarchical networks have been proposed as an alternative topology to flat Adhoc topologies. This architecture considers two types of nodes: a small number of powerful High-end sensors

Manuscript received August 14, 2010; revised January 4, 2011; accepted January 31, 2011.

doi:10.4304/jcm.6.2.185-197

(H-sensors), e.g., Imote2 [5], and a large number of Low-end sensors (L-sensors), e.g., the TelosB motes [6] (see Fig. 2). L-sensors are in charge of performing simpler tasks, while resource-rich, high-power H-sensors take on more complex tasks. A heterogeneous architecture provides scalability, notable energy efficiency and security benefits [4] [7].

In this paper, we propose IKM, a key management scheme for such a heterogeneous architecture [33]. The proposed scheme uses one variant of public key cryptography, named Identity Based Cryptography (IBC), to establish pairwise keys between sensor devices. Identity based cryptography is based on Elliptic Curve Cryptography (ECC) and pairing function. It avoids the use of digital certificates which bind a wireless sensor node identity to its public key. Also IBC enables the generation of keys between pairs of nodes without any pairwise node interaction. Indeed, a node will establish a secure key with the other node knowing only its public identity. Another advantage of IBC is the possibility of scaling the number of nodes in the network since each node keeps only its secret key.

Public key cryptography has been considered too expensive for resources constrained sensor nodes. Traditional public-key algorithms (such as RSA [8]) require extensive computations, and are therefore not suitable for tiny sensor devices. However, recent progress in ECC research [9] provides new opportunities to utilize ECC-based public-key cryptography in sensor networks. The recent implementation of 160-bit ECC on Atmel ATmega128, a CPU of 8Hz and 8 bits, shows that an ECC point multiplication takes less than one second.

Since in-network processing techniques lend themselves to the logical structure of Heterogeneous Sensor Networks (HSN), our proposed key management protocol must support these techniques. In fact, in-network processing removes the redundancy in collected data and so it increases the computation tasks while decreasing the communication operations. Since computation is much less energy consuming than communication, in–network processing saves the energy budget in WSN. In addition, in-network processing enables WSN to provide more complex services to application layer, and not only data gathering functionality. However, in-network data processing introduces also many challenging security issues. Particular keying mechanisms may preclude or reduce its effectiveness. So, the proposed key management protocol should not only support the traditional pair-wise key establishment mechanism but also cluster keys must be established. Hence, two types of keys must be computed: (1) a pairwise key shared between every two communicating nodes, and is unique to those nodes and (2) a cluster key which is a common key shared among all nodes in the same cluster, and is mainly used for securing locally broadcasted messages. Hence, the proposed scheme consists of a pairwise key establishment protocol and a cluster key transport protocol. IKM key management solution can scale well and supports the addition of new sensor nodes. Also, the proposed scheme supports re-keying mechanisms.

To analyse the security robustness of the proposed scheme, we discuss several attacks that threat its secure functioning and demonstrate how it can efficiently defend against them. Then, we evaluate analytically the storage, computational and communication overhead introduced by our scheme. Also, we report results obtained from implementation of the proposed scheme over TelosB motes.

The rest of this paper is organised as follows: Section 2 provides related work. Section 3 gives some preliminaries on elliptic curves and bilinear pairing and describes the proposed key management protocol for pairwise keys and cluster keys establishment. In Section 4, a security analysis of the proposed scheme is made to prove its resilience against various types of attacks. An overhead analysis and an evaluation of the IKM scheme using TelosB motes are also presented in Section 5. Finally, Section 6 concludes the paper.

II. RELATED WORKS

The distribution of symmetric keys is one of the main challenges in WSN. Many schemes were proposed in the literature. The simplest way is to let the network nodes share a single secret key. Unfortunately, the compromise of even a single node in a network would reveal the secret key and thus allow decryption of all network traffic. Yet another approach is the full pairwise scheme. This approach uses a shared unique symmetric key between each pair of nodes. This scheme is memory-intensive and does not scale up. To fully take advantage of the information available to the sensor networks, schemes using information from the environment were proposed. Deployment knowledge about the environment is frequently used for a more optimized design. For example, Du et al. proposed a key management scheme using deployment knowledge [11]. Liu et al. proposed a location-based pairwise key establishment for relatively static sensor networks [12] using the prior knowledge obtained before distributing the sensor nodes. The memory usage per sensor is greatly improved while the connectivity of the sensor network is maintained.

Blom [13] and Blundo et al. [14] proposed respectively a matrix and a polynomial key generation schemes. These schemes guarantee that any two nodes in a network of size n will be able to perform pairwise key, but each of these schemes also involves an Ω(n) high memory cost if we require that the system be secure against an adversary capable of compromising a fraction λ of the total number of nodes. The solution is λ secure, meaning that coalition of less than $\lambda+1$ sensor nodes knows nothing about pairwise keys of others. Another important key establishment scheme relying on probabilistic key pre-distribution technique was firstly published by Eschenauer and Gligor [1]. This approach consists of assigning to each sensor node a random subset of keys from the key pool before deployment. Any two nodes able to find one common key within their respective subsets can use that key as their shared secret to initiate communication. Further schemes were proposed based on [1], such as the q-composite random key pre-distribution scheme and the multi-path key reinforcement scheme [15].

Moreover, Du et al. developed a pairwise key management scheme [16]. This scheme combines the random key pre-distribution scheme [1] and the Blom scheme [13] to substantially improve network resilience against node capture over existing schemes, without increasing the memory overhead.

One of the important mechanisms in sensor networks, i.e. in-network processing, is not considered in the previous schemes. So, hierarchical key management solutions are proposed. Zhu et al. proposed a protocol named LEAP [2] to help establish individual keys between sensors and a base station, pairwise keys between sensors, cluster keys within a local area, and a group key shared by all nodes.

All the previous described solutions consider a network with a homogenous set of nodes. Recently deployed sensor network systems are increasingly following heterogeneous designs. In [4], Du et al. considered key management in a Heterogeneous Sensor Network (HSN) that consists of a small number of powerful High-end sensors (H-sensors, e.g., PDAs) and a large number of Low-end sensors (L-sensors, e.g., the MICA2 nodes). Their basic idea is the use of the Asymmetric pre-distribution which consists of pre-loading a large number of keys in each H-sensor while only pre-loading a small number of keys in each L-sensor. Similarly, Traynor et al. [17] proposed a probabilistic unbalanced distribution of keys throughout the network that leverages the existence of a small percentage of more capable nodes. They demonstrated that this solution can not only provide a significant level of security but also reduce the consequences of node compromise.

Another major approach for distributing keys in WSN is the asymmetric approaches which rely on public key cryptography. In these schemes, a private/public key pair is assigned to each node. Rivest Shamir Adleman (RSA) [8] algorithm is one of the most popular public-key encryption algorithms currently available. It is frequently used in wired networks. Its security is based on the difficulty of solving the Integer factoring problem. Elliptic curve cryptography (ECC) [9] was then developed by Koblitz and Miller. ECDLP “Elliptic Curve Discrete logarithm Problem” is one variant of ECC used to compute pair-wise keys. It is based on the difficulty of solving the discrete logarithm problem on elliptic curves. ECC can obtain the same security level as RSA while using a smaller key. A 160-bit ECC key has the same security as a 1024-bit RSA key [10].This smaller key size translates directly into an energy saving for the device, as fewer bits are required to be transmitted by the radio.

Many cryptographic applications based on elliptic curves use bilinear pairings, a powerful mathematical tool which enables efficient implementation of Identity-Based Cryptography (IBC). Bilinear pairings allow Diffe-Hellman problem, a well-known class of hard computational problems, to have an easy decisional version. This approach would seem to be the best approach for distributing symmetric keys in a WSN. Since Boneh et al. proposed an ID-based signature scheme [18] and encryption scheme [19] from the pairing; many schemes using pairing have been proposed. Previously, evaluating pairing was a more complex and costly operation compared to scalar multiplications of ECC. However, since several efficient algorithms for computing the pairing have been proposed [20], the pairing cost is no longer a heavy burden on sensor nodes. The first known implementation of pairings for sensor nodes based on the 8-bit/7.3828-MHz ATmega128L microcontroller (e.g., MICA2 and MICAz motes) has been investigated in [21], and it concludes that pairings based cryptography is indeed viable in resource-constrained nodes.

Given the advantages of IBC and its possible efficient implementation, we adopt it in our proposed key management scheme for HSN. Using pairing-Identity based cryptography properties, each node in the network needs only to be preloaded with its secret key from the base station and no more keys of other nodes. Later, a node can establish a shared secret key with any node in the network knowing only its public identity.

III. PROPOSED KEY MANAGEMENT PROTOCOL

A. *Preliminaries*

As briefly discussed in the introduction, key management is crucial for security sensitive applications. Many key management solutions in wireless and wired networks are based on public key cryptography. However, the traditional RSA algorithm can not be implemented for resource constrained sensor devices. Recent advances in elliptic curve cryptography make public key cryptography feasible for WSNs. In this paper, we focus on a variant of ECC named identity based cryptography which is based on bilinear pairing. Before the new key management scheme based on bilinear pairing is proposed, we first introduce some basic concepts regarding elliptic curves and pairing functions [22] [23].

(i) Elliptic curves

Elliptic curves are usually defined over binary fields $F_{2^m}(m \geq 1)$, or over prime fields $F_q(q > 3)$. Let q be a large prime number. An elliptic curve E_q defined over a finite field F_q is given by:

$$y^2 = x^3 + ax + b \text{ such that } a, b \in F_q \quad (1)$$

E_q is then the set of points P(x,y) where $x, y \in F_q$ verifying (1). E_q is a finite group if the discriminant of (1) is nonzero, i.e. $4a^3 + 27b^2 \neq 0$. The set of solutions (x,y) of (1) together with a point O, called the point at infinity, and a special addition operation define an Abelian group, called the Elliptic Curve group. The point O acts as the identity element (for more details, see [23]).

The security of elliptic curve cryptosystems is based on the difficulty of solving the Discrete Logarithm

Problem (DLP) on the elliptic curve group. The Elliptic Curve Discrete Logarithm Problem (ECDLP) defined in E_q is the following: given $P \text{ and } kP \in E_q$, it is very difficult to determine the value of k.

(ii) Bilinear pairing

Identity-based cryptography is based on the existence of bilinear maps called pairings, which are mathematically defined in terms of the elliptic curves algebra with coefficients in a finite field. Such a function is used to compute a common secret between two interlocutors to establish a confidential channel between them. A centralized entity that contains a confidential value s, associates to each user a secret built as the output of a known one-way hash function multiplied by its secret s. The hash function uses the user's public identity as input. So any pair of nodes can establish a common secret key knowing only their respective public keys. And as long as an intruder doesn't explore the unique secret key of each node, any pairwise key that this node computes can't be explored or regenerated by this intruder.

To be more accurate, the pairing can be seen as a function: $e: G_1 \times G_1 \rightarrow G_2$

where G_1 is a finite cyclic group, defined usually by a set of points of an elliptic curve, with an additive law of composition, and G_2 is a finite cyclic group implemented using a multiplicative subgroup of an extension of the underlying finite field. The two groups are of the same order. Let P be an arbitrary generator of G_1. Note that aP denotes P added to itself a times. Assume that the ECDLP problem is hard in G_1 and the DLP is hard in G_2. This pairing function must satisfy the following three properties:

(1) Bilinearity: if $P, Q, R \in G_1$ and $a \in Z_q^*$,
$e(P+Q, R) = e(P,R).e(Q,R)$,
$e(P, Q+R) = e(P,Q).e(P,R)$ and
$e(aP, Q) = e(P, aQ) = e(P,Q)^a$.

(2) Non-degenerate: There exists $P, Q \in G_1$ such that $e(P,Q) \neq 1$

(3) Computability: There exist efficient algorithms to compute e(P,Q) for all $P, Q \in G_1$

Two types of pairing are proposed in the literature: the Weil pairing and the Tate pairing. According to [22], the Tate pairing seems to be more efficient than the Weil pairing. In this paper, we adopt the Tate pairing.

To efficiently compute a Tate pairing, we must choose a pairing friendly curve such us the supersingular curves. The reader can refer to [20] to know how we can select such a curve.

Much research effort has been aimed at the optimization of the algorithm for pairing computation on supersingular curves [20][24]. The first algorithm of pairing was proposed by Miller [35] in 1986. In 2002, Baretto et al. [24] proposed some optimization to this algorithm. Then Dursma lee [25] proposed in 2003 some improvements on supersingular elliptic curves. This research has led to a new and efficient algorithm, the ηT pairing, to compute the Tate pairing on a supersingular elliptic curve defined in a binary field F_2^m. This algorithm is very efficient and well optimized for use in the context of resource constrained sensor Networks (see Fig. 1).

Algorithm 1 : ηT on $E(F_2^m)$): $y^2 + y = x^3 + x + b$

Input : $P(x_P,y_P)$, $Q(x_Q,y_Q)$
Output : $\eta_T(P,Q)$

1. $u \leftarrow x_P + 1$
2. $f \leftarrow u.(x_P + x_Q + 1) + y_P + y_Q + b + 1 + (u + x_Q)s$
3. $for\ i \leftarrow 1\ to\ (m+1)/2\ do$
4. $u \leftarrow x_P,\ x_P \leftarrow \sqrt{x_P},\ y_P \leftarrow \sqrt{y_P}$
5. $g \leftarrow u.(x_P + x_Q) + y_P + y_Q + x_P + (u + x_Q)s + t$
6. $f \leftarrow f.g$
7. $x_Q \leftarrow x_Q^2,\ y_Q \leftarrow y_Q^2$
8. end for
9. Return $f^{(2^{2m}-1)(2^m - 2^{(m+1)/2}+1)}$

Figure 1. The ηT algorithm [20]

B. Network Model & Assumptions

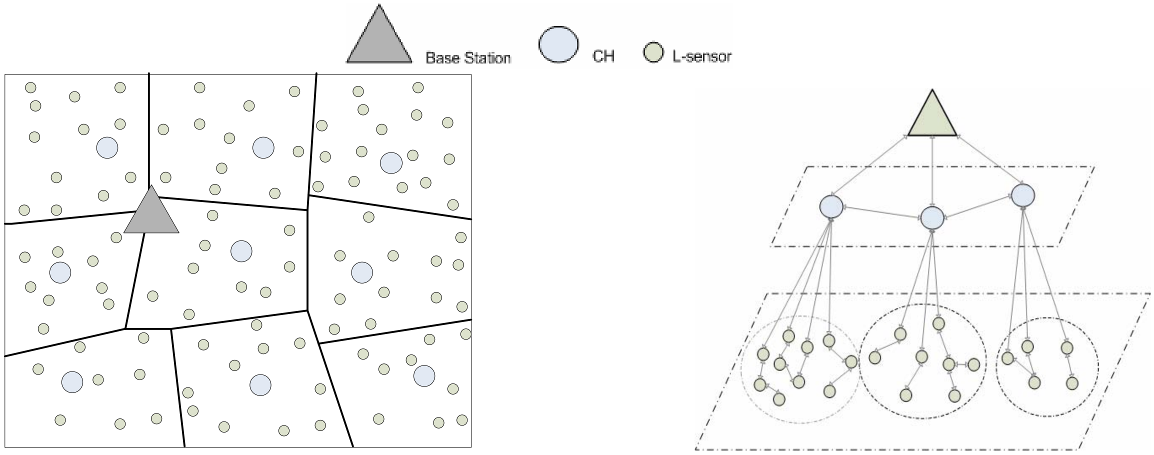
We consider a hierarchical sensor network consisting of a Base Station (BS) and a set of heterogeneous sensor nodes grouped in clusters as depicted in Fig. 2. Each cluster contains several L-sensors deployed with one H-sensors. The general function of an L-sensor is to collect raw data and forward it to the corresponding H-sensor, designated as cluster head. This traffic represents the intra cluster communication. The inter cluster traffic is made by the H-sensors. H-sensors perform data aggregation of information flows coming from L-sensor nodes, and forward the aggregated data toward the Base Station either directly or via other Cluster Heads (CHs). One can build a heterogeneous sensor network by distributing H-sensors and L-sensors at the same time, or by adding a small number of H-sensors into an existing homogeneous sensor network. In addition, the BS, which is the ultimate destination of data streams from all the sensor nodes, may be connected to an outside network.

We assume a static sensor network, where all sensor nodes have fixed locations and arranged according to a cluster based topology. Each node is preloaded with a unique ID. Both H-sensors and L-sensors are assumed to know their location information. Also, nodes must be loosely synchronized as they are pre-loaded with a bootstrapping time. The BS, acting as a sink, is assumed trusted and will never be compromised. It is equipped with tamper resistant material.

It is expected that after cluster formation, each cluster head knows the IDs of each node in its cluster and the BS knows the IDs of the cluster heads. Also, each node is assumed to have a list of its immediate neighbors.

C. Identity based key management protocol

In this section, we describe IKM, the proposed key management protocol based on the difficulty of solving the discrete logarithm problem on elliptic curves and the



a) Physical view b) Logical view

Figure 2. Clustered HSN model.

Tate pairing. The proposed protocol reduces key space requirements at the nodes. In fact a node does not need to store any key of the other nodes, rather it computes secret sharing key by using pairing. In addition, the proposed protocol allows the addition of new nodes and updates keys after a certain time to avoid cryptanalysis or following a node compromise. Since a single key is inappropriate for securing all communication patterns in a sensor network, our protocol supports the establishment of two different types of keys: the pairwise key and the cluster key. This helps minimize the impact of any key compromise to only a certain number of nodes and also enables the use of in-network processing techniques which are very important in WSN.

The protocol is composed of two phases: A pre-deployment phase where the BS preloads each node with corresponding information to be used to generate keys. The second phase sets up the network structure and establishes pairwise keys between neighbouring nodes and cluster keys for each group of nodes belonging to the same cluster.

Pre-deployment phase: During the bootstrap of the network, the BS must perform the following extra tasks.

- The BS chooses two groups G_1 and G_2, of the same prime order q. G_1 is an additive group and G_2 is a multiplicative group. Let P be an arbitrary generator of G_1. Then, a bilinear map $e: G_1 \times G_1 \to G_2$ and two collision resistant cryptographic hash functions H_1 and H_2 are determined, where $H_1 : \{0,1\}^* \to G_1$, mapping from arbitrary length strings to points in G_1 and $H_2 : G_1 \to Z_q^*$, a mapping from G_1 to a value in Z_q^*.
- The BS picks a random number $s \in Z_q^*$ and then sets $P_{pub} = sP$. The base station keeps *s* secret. The BS uses the value of s to generate secret keys of sensor nodes.
- The BS preloads each node in the network with a unique identity and a bootstrapping time. The bootstrapping time must be sufficiently long to allow each node to establish pairwise keys with all its neighbours and also to obtain the cluster key. After this time, every node will not establish any other key except when the BS requests this and makes the necessary configuration.
- Then, the BS computes a private key $S_i = sQ_i$ where $Q_i = H_1(ID_i)$ for each node i, which is assigned to it prior to deployment. Therefore, the system parameters are $< G_1, G_2, e, q, P, P_{pub}, H_1, H_2 >$.

Post deployment phase: Once the nodes have been deployed, each node initialises a timer that expires when the bootstrapping time is finished. During this phase, pairwise keys as well as cluster keys are established.

(i) Establishing pairwise keys between a pair of nodes N_i and N_j

A pairwise key must be computed for each pair of adjacent nodes. We denote by N either an H-sensor node or an L-sensor node or the BS. As illustrated in Fig. 3, a pairwise key is established as follows:

1. Node N_i generates a key $V_{i,j} = e(S_i, Q_j) = e(Q_i, Q_j)^s$ which is used as a message authentication code (MAC) key between N_i and N_j. In the same way, N_j generates $V_{i,j} = e(Q_i, S_j) = e(Q_i, Q_j)^s$. Then, the pairwise key is established as follows:
2. N_i chooses a random $r_i \in Z_q^*$, computes $R_i = r_i P$ and $M_i = MAC_{V_{i,j}}(N_i, N_j, R_i)$ and sends the message $[N_i, N_j, R_i, M_i]$ to N_j.
3. Similarly, N_j chooses a random $r_j \in Z_q^*$, computes $R_j = r_j P$ and $M_j = MAC_{V_{i,j}}(N_i, N_j, R_j)$ and sends the message $[N_j, N_i, R_j, M_j]$ to N_i.
4. Upon receiving the message, N_i verifies the MAC and computes the pairwise key $K_{i,j} = H_2(r_i R_j) = H_2(r_i r_j P)$.
5. Similarly, N_j verifies the MAC and computes the pairwise key $K_{i,j} = H_2(r_j R_i) = H_2(r_i r_j P)$.

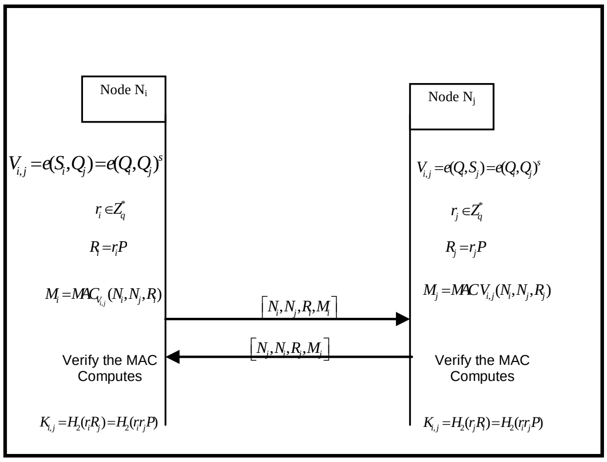


Figure 3. Key establishment between two nodes N_i and N_j

(ii) Establishing cluster key

A cluster key is a key shared by all the nodes of the same cluster. The cluster key establishment phase (see Fig. 4) follows the pairwise key establishment phase. At the beginning of this phase:

1. Each cluster head H_i first generates a random key $KC_i \in Z_q^*$.
2. Then, H_i transmits the message $[H_i, L_j, E_{i,j}, T_i, M_{i,j}]$ to each low end node L_j at its transmission range where:
   - $M_{i,j} = MAC_{K_{i,j}}(H_i, L_j, T_i, KC_i)$: the message authentication code
   - $E_{i,j} = Encrypt(KC_i, K_{i,j})$: the encrypted cluster key
   - $K_{i,j}$: the pairwise key shared between H_i and L_j
   - And T_i a fresh timestamp
3. Upon receiving this message at the time T_j, L_j verifies whether $(T_j - T_i) \le \Delta T$. If it holds, L_j accepts H_i's message, where ΔT is preset constant to protect against replay attacks. Then, L_j verifies the MAC and decrypts the message to obtain the key KC_i and stores it in its table.
4. Next, L_j retransmits the cluster key, in the same way, using the pairwise key shared with each L-sensor in its neighbourhood. This process is repeated until all the nodes in the cluster receive key KC_i.

The CH must update the cluster key when one of the neighbours is revoked. This also protects against cryptanalysis attacks. So, it regenerates a new cluster key and transmits it to all the remaining neighbours in the same way.

(iii)Node addition

A desirable property in a scalable key management scheme is the ability of adding new sensors to the network. New nodes may be added to the network upon sensor node failure or to increase the network coverage. The newly deployed sensor nodes need to establish secret keys with existing nodes. Our key establishment solution allows for the dynamic addition of new sensors without having to contact the previously deployed sensors. In

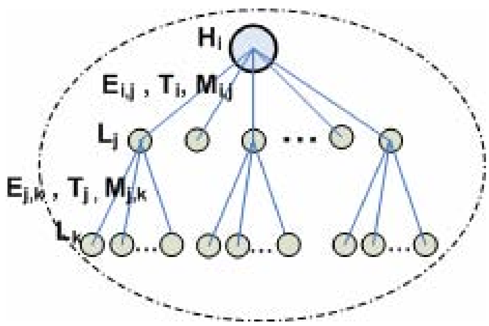


Figure 4. Establishment of cluster key

fact, each node must be pre-loaded with the pair (public key, secret key) assigned to it by the key generation center (the BS). Hence, this newly added node can then establish a secure communication with any of its neighbours. Our scheme allows the network to grow without an upper limit on the network size.

(iv)Re-keying

The pairwise keys and the cluster keys must be updated after a certain time. Using the same encryption key for an extended period of time may expose the network to a cryptanalytic risk. Also, a cluster key must be renewed after detecting a compromised node.

To renew a pairwise session key, nodes must choose a new random number and re-execute steps 2 to 5 of the proposed pairwise key scheme. Also, to update a session cluster key, a cluster head must choose a new random number and repeat the process of the cluster key distribution.

IV. Security Analysis

We define typical attacks on sensor networks and demonstrate how our proposed protocol can prevent them. An attack consists of one or more of the following operations.

- Eavesdropping, modifying, and false messages injecting.

The first type of attacks is to eavesdrop on the information carried in the messages. This information may be the data related to pairwise keys or the cluster key it self. This operation threatens message confidentiality. Modifying messages threatens message integrity. Finally, fabricating false messages threatens message authenticity.

An adversary cannot know the value of a pairwise key because it is generated at each end point, which is programmed with a random secret number. As encryption is used when transmitting cluster keys, their confidentiality is guaranteed. Modifying keying materials or keys and injecting false messages is prevented as MACs are used. Regarding the fact that every node in the network goes through key discovery and authentication phase, these attacks are also prevented for ordinary messages. In fact, during network bootstrapping, nodes establish two types of keys that should secure their communication from eavesdropping, modification, and injection of false messages. Hence, confidentiality, integrity, and data authentication are assured.

- Replaying of messages.

This operation threatens message freshness. In the bootstrapping phase, an attacker can replay the old pairwise key establishment messages. This attack is prevented as each sensor has a secret random number which allows it to generate the key. So, even if an adversary replays an old message trying to establish a pairwise key with a valid node, it will not be able to do this as he must solve the discrete logarithm problem to get the random numbers r_i or r_j. For the cluster key, this attack is prevented as in each transmitted message there is a timestamp which guarantees its freshness. Moreover, when the bootstrapping time expires, any replay of messages designed for establishing any key is eliminated.

- Node capture attack

In a node capture attack, an adversary gains full control over sensor nodes through direct physical access. Our proposed protocol cannot eliminate the node compromise problem, which is a very hard problem in WSN. However, the proposed protocol can prevent adversaries from spreading the impact of node compromise across the entire network. Given the assumptions mentioned before, an adversary cannot compromise a node in the bootstrapping phase. This is a powerful property as all the keys are established during this phase and hence the adversary cannot explore these keys. After this phase, an adversary can compromise a node and compromise all the links directly connected to this node. The rest of the network remains secure as this adversary cannot establish any other links with other nodes placed in a distant region. So the impact of the attack will be local to the compromised node and its immediate neighbours.

- Attacks on routing protocols

Many attacks on routing protocols are based on the node capture attack. Our proposed protocol can efficiently defend against several of them such that the sinkhole attack, the wormhole attack, the Sybil attacks etc. More details about these attacks can be found in [26].

After deployment, it is assumed that each node would keep a list of its immediate neighbours. This list is trusted as it is acquired in the bootstrapping phase. Therefore, a sinkhole attack or a wormhole attack can be detected. Moreover, the Sybil attack is prevented as valid identities are assigned by the trusted BS, and since each node knows the set of valid nodes with which it may communicate. Hence, an adversary cannot convince another node, which is not really in its neighbourhood, that it is a near one. A variant of the Sybil attack, the node clone attack, consists of cloning a device and placing it in different parts of the network or in the same place in order to disrupt the routing protocol. If the adversary places the cloned nodes in different regions, a legitimate node will detect this attack as it does not include this identity in the list of neighbours. If the nodes are placed in the same region, they can establish denial of service attack. It is then the role of the media access control layer to prevent this attack.

V. Performance evaluation

In this section, we present the performance evaluation of the proposed key management scheme. We compare in the first subsection the storage overhead of our scheme with the key pre-distribution scheme proposed by Eschenauer and Gligor [1] which is referred as the E-G scheme, Du et al. scheme [4] which is referred as the Asymmetric Pre-distribution scheme (AP scheme), Du et al. scheme [27] which is referred as the centralized ECC key management scheme, and the TinyIBE scheme [34] proposed by Szczechowiak et al. Next, the communication overhead and the computational overhead introduced by the proposed pairwise key establishment scheme are compared with E-G scheme and TinyIBE. In the second subsection, we give implementation details and evaluation of the proposed scheme.

A. Overhead analysis

In this section, we analyse the storage cost (Scost), the computation cost (Cpcost) and the communication cost (Cccost) of the proposed key management scheme.

(i) Storage Overhead

We assume that our Heterogeneous Sensor Network (HSN) is composed of M H-sensors and N L-sensors. Typically, we have M<<N. Because we use pairing in our proposed key management scheme, neither H-sensor nor L-sensor nodes need to pre-load private or public keys of the other nodes. Each node needs only to store its secret key assigned to it by the key generation centre (which is the base station in our HSN). Hence, after deployment, each node can compute its shared secret with each of its neighbouring nodes using pairing computation. Thus, the total storage requirement in key length unit for a network with M+ N sensor nodes is:

$$S\cos t = M + N \tag{2}$$

For the AP scheme an L-sensor node and an H-sensor node are assumed to have a probability of connecting given by the following equation:

$$P[Match] = 1 - \frac{(P-k)!(P-m)!}{P!(P-m-k)!} \tag{3}$$

Where P is the key pool size, k is the size of the key ring in each L-sensor, and m the size of the key ring in H-sensor nodes with m>>k. The total number of pre-loaded keys in a network using the AP scheme is:

$$S\cos t = N * k + M * m \tag{4}$$

For the E-G scheme and a homogeneous sensor network with M+N sensors, the probability of connectivity is given by the following equation:

$$P[Match] = 1 - \frac{((P-k)!)^2}{(P-2k)!P!} \quad (5)$$

Where P is the key pool size and k is the key ring size. The total number of pre-loaded keys in the network is:

$$S\cos t = (N+M)*k \quad (6)$$

In the centralized ECC key management scheme designed for HSN, each L-sensor is pre-loaded with its private key and the public key of H-sensor. Each H-sensor is pre-loaded with public keys of all L-sensors, plus a pair of private/public key for itself, and a key K_H for newly deployed sensors. Thus, an H-sensor is pre-loaded with N + 3 keys. The total number of pre-loaded keys is:

$$S\cos t = M*(N+3) + 2*N = (M+2)N + 3M \quad (7)$$

For the TinyIBE scheme, each H-sensor is pre-loaded with 3 keys and the total number of pre-loaded keys in the network is:

$$S\cos t = 3*M \quad (8)$$

Next, we use an example to compare the storage requirement of our proposed IKM scheme, the E-G scheme, the AP scheme, the centralized key management scheme, and TinyIBE. If we consider a HSN network with N=1000 L-sensors and M=10 H-sensors, the total memory requirement for our proposed scheme is 1010 keys. However, in the AP scheme, if we consider a key pool with 10,000 keys and each H-sensor is loaded with m=500 keys from this pool where each L-sensor is loaded with k=20 keys, the total storage requirement is 1000*20+10*500=25000 keys, which is almost 25 times larger than our proposed scheme for a connection probability not exceeding 65%. Now, if we consider the E-G scheme applied in a homogeneous sensor network with M+N=1010 sensors, where each sensor is pre loaded with 100 keys, and for the same connection probability as the AP scheme (65%), the memory requirement will be 1010*100=101000 keys, which is 100 times larger than our proposed scheme. To increase the connection probability, we must also increase the size of the key rings in the E-G scheme or in the AP scheme, resulting in more intensive memory requirement. However, for our proposed scheme, the storage requirement is independent of node connectivity.

For the centralized key management scheme, which is an asymmetric key pre-distribution scheme, the total storage requirement is 12030 keys, which is 12 times larger than the storage requirement of our proposed IKM scheme. The TinyIBE scheme has a storage cost of 30 keys which is lower then the memory requirement of our proposed scheme.

In Fig. 5 and Fig. 6, we plot the total storage requirements of the IKM scheme, the centralized key management scheme, and the E-G scheme, for different sizes of the sensor network and different numbers of pre-loaded keys in E-G scheme. Recall that IKM and ECC are designed for heterogeneous sensor networks. For these schemes, we assume that the network is always

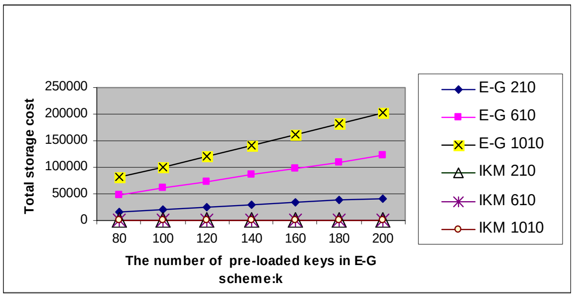


Figure 5: Comparison of the required storage space expressed in key length unit as a function of the number of preloaded keys in E-G scheme between the IKM scheme and the E-G scheme.

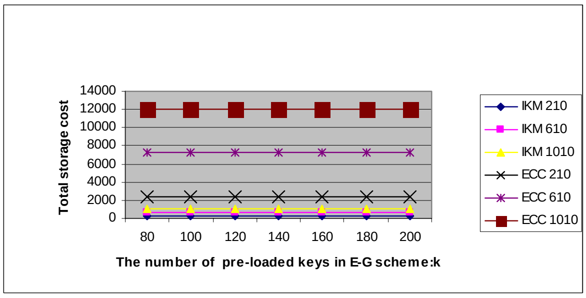


Figure 6: Comparison of the required storage space expressed in the unit of key length as a function of the number of preloaded keys in E-G scheme between the IKM scheme and the ECC key management scheme.

composed of 10 H-sensors and the rest are L-sensor nodes. The plots show clearly that the proposed IKM scheme has significantly less storage overhead than the other schemes.

(ii) Communication Overhead

The most energy consuming operation in sensor networks is communication. Therefore, it is important that transmissions be used sparingly. In our proposed key management protocol, pairwise keys and cluster keys must be established. To establish a pairwise key with a desired neighbour, a node must activate its transmitter twice, once for transmitting and once for receiving. . If a node has more than one neighbour, these two operations will be repeated with each of its communicating neighbours. So, there is only one message to be exchanged for each node to establish a key with a neighbouring node. For the other comparative schemes, each node must also exchange one message to establish a pairwise key. However, the size of the message varies from one scheme to another. For example, the message broadcasted by a node in the E-G scheme contains the list of key identifiers on its key ring. If the key pool contains 10000 keys and each node is preloaded with k=100 keys, we have a connection probability of 65%. Each key identifier requires 14 bits and the broadcasted message have a size of 174 bytes. Now, we evaluate the message length of TinyIBE. Using this protocol, the exchanged message contains two values C1 and C2. C1 is a point on the elliptic curve $E(F_2^{271})$, which can be compressed to 34 bytes and C2 has the size of the session key (128-bits). The resulting message has 52 bytes. For our proposed pairwise key establishment scheme, the exchanged message is composed of two values R and M. R is a point on the elliptic curve $E(F_2^{271})$ coded on 34 bytes and M is a MAC of size 8 bytes. So, the resulting message has 42 bytes. Therefore, our proposed scheme uses a smaller message compared to E-G scheme and TinyIBE.

Concerning the cluster key establishment process, the CH generates the cluster key and sends it using a secure channel to all the nodes in its neighbourhood. Each neighbouring node in the same cluster will activate its transceiver once to receive the cluster key. If one-hop cluster topology is used, this reception is the only communication cost at each of the L-sensor nodes. Otherwise, if a multi-hop cluster topology is used, the communication cost per L-sensor node will be more important as the intermediate L-sensors must activate their transceivers again to distribute the cluster key to other L-sensors in their neighbourhood.

Let D_H be the average number of neighbouring nodes with which an H-sensor node can communicate and D_L the average number of neighbouring nodes of an L-sensor in the same cluster. We assume that $D_H >> D_L$. Then, the total communication cost in unit of transmission operations (send or receive) is:

For an L-sensor:

$$Cc\cos t = 2 * D_L + 1 \text{ (For a leaf node)} \quad (9)$$

$$Cc\cos t = 3 * D_L \text{ (For a non leaf node)} \quad (10)$$

For an H-sensor:

$$Cc\cos t = 3 * D_H \quad (11)$$

(iii) Computational overhead

The proposed IKM scheme requires several computational operations such the Tate pairing, hash operation, MAC calculation, encryption and decryption. The most computationally intensive module is the Tate pairing. It requires a large number of arithmetic operations (addition, multiplication,..), performed on large integers. Table 1 summarizes the total number of arithmetic operations needed by our pairing function.

Each sensor node must then compute the Tate pairing, two point multiplications, two MAC functions, and one hash function to establish a pairwise key.

Let us compare the computational overhead of our proposed scheme with the E-G scheme and TinyIBE. The E-G scheme introduces a high storage and communication overhead but it has no computational overhead. The TinyIBE makes two steps (Encrypt and decrypt) on-line to establish a pairwise session key. The encryption step involves two hashing operations, two point multiplications, one exponentiation, one addition, and one XOR operation. The decryption step requires one ηT pairing calculations and one hashing to retrieve the session key. We can notice that our proposed scheme for

TABLE I. Number of arithmetic operations executed by the pairing function of the IKM scheme

Arithmetic operation	Modular addition	Modular multiplication	Square root	Modular square
Tate pairing	1542	1169	268	816

pairwise key establishment introduces too much computational overhead compared to E-G scheme but almost the same overhead when compared to TinyIBE. This overhead is reasonable as TinyIBE and the proposed IKM scheme are based on asymmetric cryptography which provides a high level of security but is more computationally intensive.

To compute the cluster key, the H-sensor must make D_H encryptions and MAC calculations. A leaf L-sensor makes only one decryption and one MAC calculation, whereas a non leaf L-sensor must make one decryption and one MAC calculation, plus (D_L-1) encryptions and MAC calculations to deliver the cluster key to the nodes which have not yet received it. Therefore, the total computation cost is:

For L-sensor:

(For a leaf node)

$$\begin{aligned} &Cp\cos t = D_L Pairing + 2 * D_L Mul + D_L Hash \\ &+2 * D_L MAC + 1Dec \end{aligned} \quad (12)$$

(for non-leaf node)

$$\begin{aligned} &Cp\cos t = D_L Pairing + 2 * D_L Mul + D_L Hash + 2 * D_L MAC \\ &+1Dec + (D_L - 1)(Enc + MAC) \end{aligned} \quad (13)$$

For H-sensor:

$$\begin{aligned} &Cp\cos t = D_H Pairing + 2 * D_H Mul + D_H Hash \\ &+2 * D_H MAC + D_H (Enc + MAC) \end{aligned} \quad (14)$$

We notice that our proposed IKM scheme requires a relatively high computation overhead but, at the same time, provides a high level of security. Such protocol can be used for security critical applications, where the level of security is more important than the lifetime of the sensor node.

Next, we present experimental results obtained with an implementation of the IKM scheme using TelosB sensor nodes.

B. Evaluation of the IKM scheme

As a proof of concept, and to show that the proposed scheme can be efficiently implemented on resource-constrained sensor nodes, we implemented the proposed IKM scheme on a real testbed of TelosB motes [6]. We provide measurements of the storage cost, the computation time of pairwise key, and the energy to compute this key. Also, we consider the impact of the cluster size on the latency of cluster key delivery and on the amount of energy consumed by the whole cluster to distribute this cluster key.

(i) TelosB testbed and parameter setting

A TelosB mote is of the size of two AA batteries. It has an IEEE 802.15.4/ZigBee compliant RF transceiver, allowing radio communication in the frequency range 2.4 to 2.4835 GHz (a globally compatible ISM band), and a bit rate of 250 kbps data rate. It is based on the TI MSP430 microcontroller. The MSP430 incorporates an 8MHz, 16-bit RISC CPU, 48K bytes flash memory (ROM), and 10K bytes RAM. The TelosB motes run TinyOS operating system version 1.1.15 and support NesC as programming language [28]. We adopted TelosB motes as L-sensor and H-sensor nodes as they are the only sensor motes at our disposal.

Our objective is to implement the proposed pairing based key management protocol and integrate it on these motes. Our implementation is based on the MIRACL Library [29] (Multiprecision Integer and Rational Arithmetic C/C++ Library), which provides all the necessary elliptic curve primitives and functions to compute pairings and other cryptographic algorithms. MIRACL handles large numbers arithmetic and offers full support for ECC over the prime field F_p, and the binary field F_2^m. It is a suitable choice when the implementation is designed for embedded and constrained environment.

To compute the pairing function, in a first step, we must find a suitable elliptic curve. Curve parameters must be chosen with care to allow efficient computations and provide a reasonable level of security, where our protocol is neither vulnerable to the Pohlig-Hellman attack [23], nor to the index calculus attack [23]. To satisfy these requirements, we decided to use the supersingular elliptic curve defined by (15) over the binary field F_2^{271} (m=271) which has an embedding degree of k=4 (i.e. the output of pairing will be in the field $F_2^{4\times271}$, the quadratic extension field).

$$y^2 + y = x^3 + x \tag{15}$$

Therfore, in our case, m=271 and k=4. We have then $k * m = 1084 > 1024$ (recommended to achieve security requirements). Hence, the key generated by the pairing function is a 1084 bit key, thus giving us a wide safety margin.

The pairing algorithm requires a large number of arithmetic operations. These operations are made modulo an irreducible polynomial f(x). For the particular binary field F_2^{271}, we have selected the following trinomial:

$$f(x) = x^{271} + x^{201} + 1 \tag{16}$$

In the second step, we must specify which pairing algorithm to adopt. There has been a lot of work on efficient algorithms for computing pairings on elliptic curves. Research results show that the improved Duursma-Lee algorithm to compute the Tate pairing over F_2^m based on ηT pairing, is one of the fastest known [20]. For this reason, we adopt this algorithm in implementing the pairing function. This algorithm is implemented in the C language and using the MIRACL library. However, running the code in resource-constrained nodes, such as TelosB, is not straightforward and thus, adaptations have been made in order to fit MIRACL into the platform.

To evaluate the effect of cluster size on the latency of the cluster key delivery, we use the TOSSIM simulator [30] integrated with TinyOS. To evaluate the average node energy consumption to compute a pairwise key, we use PowerTOSSIM simulator [31]. It simulates the wireless network at the bit level and uses a simulation model based on the different TinyOS components. Also, this simulator is used to study the impact of cluster size on the total energy consumption. During the experiment, each test is repeated 20 times. We present in the next subsections the results of pairwise key protocol implementation and then results of cluster key protocol implementation. Each protocol is implemented and evaluated separately as the memory of TelosB mote cannot simultaneously support both protocols.

(ii) Evaluation of pair-wise key management protocol

Our pairwise key management protocol is implemented using nesC language in TinyOS environment. It uses several TinyOS components and interfaces. With all security components implemented, the program has a code size (ROM) of 43 486 bytes and a data size (RAM) of 2305 bytes. This is a relatively large space in ROM, due mainly to the use of the MIRACL library, which requires a large programming space. Concerning the RAM space, we have used only 23% of the total available space.

Measurements made are summarized in Table 2. Experimental results show that our proposed protocol takes on average 20.1 sec to establish a pair-wise key between two TelosB motes. This time includes the computation delay, especially that of the pairing function, which is about 9 sec, and also the communication delay.
The energy consumed by our proposed pairwise key establishment protocol has been measured using the PowerTOSSIM simulator. The energy overhead introduced by our key management protocol is about 1707 mJ, which is negligible compared to the total energy of a mote (< 1%). Therefore, the proposed key management solution is efficient and suits well the severe constraints of sensor nodes. Such efficiency is necessary for any security solution in wireless sensor devices.

(iii) Evaluation of cluster key management protocol

The proposed cluster key establishment protocol implemented on TelosB motes requires a code size (ROM) of 14120 bytes and a data size of 537 bytes.

We evaluate also the performance of the cluster key establishment protocol in terms of the effect of scaling the number of nodes on the latency of cluster key delivery and on the total energy consumed by all the nodes in the cluster to obtain this key. To do this, we use the TOSSIM and the PowerTOSSIM simulators provided with TinyOS.

TABLE II.
EXPERIMENTAL EVALUATION OF THE PROPOSED PAIRWISE KEY ESTABLISHMENT PROTOCOL YPE SIZES FOR CAMERA-READY PAPERS

	Time (sec)	Energy (mJ)	ROM (bytes)	RAM (bytes)
Pairwise key establishment scheme	20,1	1707	43 486	2305

Fig. 7 shows the average latency of the cluster key delivery when the cluster size is varied from 10 to 100 nodes. We see that even for a cluster of size 100, the cluster key delivery time remains below 3 minutes, which is still acceptable.

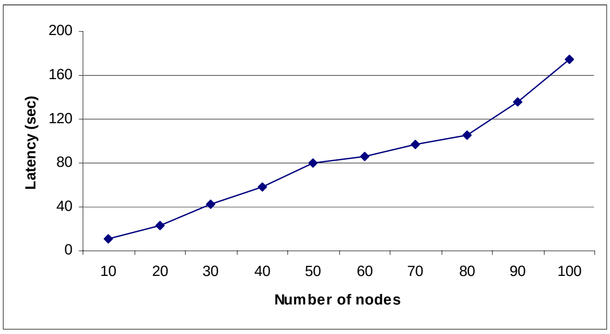


Figure 7: Impact of cluster size on the latency of cluster key delivery.

In wireless sensor networks, protocol energy overhead must be minimized. Fig. 8 shows the total energy consumed by a cluster to distribute the cluster key to the various nodes. We notice that our protocol doesn't exhaust the energy resources of sensor nodes as it only consumes about 20 joules when the cluster contains 100 nodes.

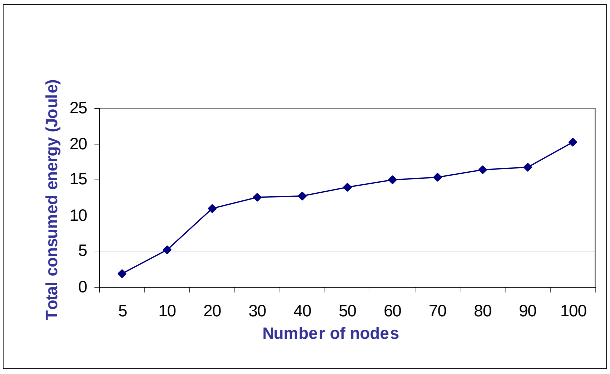


Figure 8: Overall cluster energy consumption as a function of the number of nodes in the cluster.

VI. CONCLUSION

This paper introduced a key management scheme for heterogeneous sensor networks. It supports the establishment of pairwise keys and cluster keys to enable different communication patterns. The establishment of keys is based on Pairing Identity based Cryptography, a variant of public key cryptography. Security analysis of the proposed scheme showed its resilience to various types of attacks, especially the node compromise attack. The pairing based key management scheme was shown to provide a low storage cost compared to well known key management schemes and a relatively high communication and computation overhead. However, despite this overhead, the TinyOS implementation showed that the proposed scheme can be efficiently implemented in real sensor networks, running security critical application.

REFERENCES

[1] L. Eschenauer, V.D. Gligor, “A key management scheme for distributed sensor networks”, in: Proceedings of the 9th ACM Conference on Computer and Communication Security. November 2002, pp. 41–47.
[2] S. Zhu, S. Setia, and S. Jajodia, “LEAP: Efficient security mechanisms for large-scale distributed sensor networks”, In ACM CCS 2003, October 2003, pp. 62–72.
[3] P. Gupta and P. Kumar, “The capacity of wireless networks,” IEEE Transactions on Information Theory IT 2000, vol. IT-46(2), pp. 388–404, 2000.
[4] X. Du, Y. Xiao, M. Guizani, H.H. Chen, “An Effective Key Management Scheme for Heterogeneous Sensor Networks”, AdHoc Networks, Elsevier, vol. 5, issue 1, January 2007, pp. 24–34.
[5] R. Adler, M. Flanigan, J. Huang, R. Kling, N. Kushalnagar, L.Nachman, C.-Y. Wan, and M. Yarvis, “Intel mote 2: an advanced platform for demanding sensor network applications,” in SenSys ’05, New York, NY, USA: ACM Press, 2005, pp. 298–298.
[6] Crossbow Technology Inc., www.xbow.com
[7] M. Bohge and W. Trappe,"An authentication framework for hierarchical ad hoc sensor networks", In Proc. of 2003 ACM workshop on Wireless Security (WiSE '03), August 2003.
[8] R. L. Rivest, A. Shamir, and L. M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM, 21(2):120–126, 1978.
[9] N. Koblitz, “Elliptic curve cryptosystems,” Mathematics of Computation 48, 1987, pp. 203-209.
[10] N. Gura, A. Patel, A. Wander, H. Eberle, S.C. Shantz, “Comparing elliptic curve cryptography and RSA on 8-bit CPUs”, in: Proceedings of the 6th International Workshop on Cryptographic Hardware and Embedded Systems, Boston, Massachusetts, August 2004.
[11] W. Du, J. Deng, Y.S. Han, S. Chen, P.K. Varshney, A key management scheme for wireless sensor networks using deployment knowledge, in: Proceedings of IEEE INFOCOM 2004
[12] F. Anjum, Location dependent key management using random key predistribution in sensor networks, in: Proceedings of WiSe’06.
[13] R. Blom, “An optimal class of symmetric key generation systems,” Advances in Cryptology: Proceedings of EUROCRYPT 84, Lecture Notes in Computer Science, Springer-Verlag, 1985, 209:335–338.
[14] C. Blundo, A. D. Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung, “Perfectly-secure key distribution for dynamic conferences,” Lecture Notes in Computer Science, 1993, 740:471–486.
[15] H. Chan, A. Perrig, D. Song, Random key predistribution schemes for sensor networks, in: Proceedings of the 2003 IEEE Symposium on Security and Privacy, May 11–14, pp. 197– 213.
[16] W. Du, J. Deng, Y. S. Han, and P. K. Varshney, “A pairwise key pre-distribution scheme for wireless sensor networks”, in Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS), Washington DC, USA, October 27-31 2003, pp. 42–51.

[17] P. Traynor, H. Choi, G. Cao, S. Zhu, T. Porta, “Establishing pair-wise keys in heterogeneous sensor networks”, in: Proceedings of IEEE INFOCOM 06.
[18] D. Boneh, B. Lynn, H. Shacham, “Short signatures from the weil pairing”, in: Proc. Advances in Cryptology, Asiacrypt 2001, Springer-Verlag, LNCS 2248, 2001, pp. 514–532.
[19] D. Boneh, M. Franklin, “Identity-based encryption from the weil pairing”, in: Proc. Advances in Cryptology, Crypto 2001, Springer-Verlag, LNCS 2139, 2001, pp. 213–229.
[20] P. Barreto, S. Galbraith, C. O’hEigeartaigh, M. Scott, “Efficient pairing computation on supersingular abelian varieties”, Designs, Codes and Cryptography, 42(3)(2007) 239–271.
[21] L. B. Oliveira, D. F. Aranha, E. Morais, F. Daguano, “TinyTate: computing the tate pairing in resource-constrained sensor nodes”, in: Proc. IEEE International Symposium on Network Computing and Applications NCA 2007, pp. 318–32.
[22] Lawrence C. Washington Elliptic Curves: Number Theory and Cryptography, Second Edition Series: Discrete Mathematics and Its Applications Volume: 50, CRC 2008
[23] D. Hankerson, A. Menezes, and S. Vanstone. Guide to Elliptic Curve Cryptography. Springer, 2004.
[24] P. S. L. M. Barreto, H. Y. Kim, B. Lynn, and M. Scott. Efficient algorithms for pairing-based cryptosystems. In Advances in Cryptology –Crypto’2002, volume 2442 of Lecture Notes in Computer Science, pages 354–368. Springer-Verlag, 2002 .
[25] I. Duursma and H.-S. Lee. Tate pairing implementation for hyperelliptic curves $y^2= x^p-x+d$. In Advances in Cryptology – Asiacrypt’2003, volume 2894 of Lecture Notes in Computer Science, pages 111–123. Springer-Verlag, 2003
[26] C. Karlof and D.Wagner. Secure routing in wireless sensor networks: Attacks and countermeasures. Elsevier’s AdHoc Networks Journal, Special Issue on Sensor Network Applications and Protocols, 1(2–3):293–315, September 2003.
[27] X. Du, M. Guizani, Y. Xiao, S. Ci, H.H. Chen, “A Routing-driven Elliptic Curve Cryptography based Key Management Scheme for Heterogeneous Sensor Networks,” IEEE Transactions on Wireless Communications, accepted for publication, Apr. 2007.
[28] D. Gay, P. Levis, R. V. Behren, M. Welsh, E. Brewer, and D. Culler, “The nesC Language: A Holistic Approach to Networked Embedded Systems”, In Proceedings of Programming Language Design and Implementation (PLDI) 2003, June 2003.
[29] M. Scott. MIRACL – Multiprecision Integer and Rational Arithmetic C/C++ Library, 2007. http://www.shamus.ie.
[30] P. Levis, N. Lee, M. Welsh, and D. Culler, "TOSSIM: Accurate and scalable simulation of entire TinyOS applications", in Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys) 2003, Nov. 2003.
[31] V. Shnayder, M. Hempstead, B. Chen, G. Werner-Allen, and M. Welsh. “Simulating the Power Consumption of Large-Scale Sensor Network Applications”, In Proceedings of the Second ACM Conference on Embedded Networked Sensor Systems (SenSys'04), 2004.
[32] V. S. Miller. Short programs for functions on curves. Unpublished manuscript, 1986.
[33] M. Boujelben, O. Cheikhrouhou, H. Youssef, and M. Abid. “A Pairing Identity based Key Management Protocol for Heterogeneous Wireless Sensor Networks”, In Proceedings of the First IFIP/IEEE International Cnference on Network and Service Security (N2S), Paris, France, June. 2009.
[34] P. Szczechowiak and M. Collier. “TinyIBE: Identity-Based Encryption for Heterogeneous Sensor Networks”, In Proceedings of the 5th International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP ‘09), pages 319-354, Melbourne, Dec. 2009.

Manel Boujelben was born in Sfax, Tunisia 1982. She received the MS Degree in new technologies of computer dedicated systems (2006) from the National School of Engineers of Sfax (Tunisia). Currently, she is a PhD student at the Computer & Embedded System (CES) laboratory at the National School of Engineers of Sfax. She has authored more than 7 papers. Her research interests are data security, key management, adhoc and sensor networks.

Habib Youssef received a Diplôme d'Ingénieur en Informatique from the Faculté des Sciences de Tunis, University of El-Manar, Tunisia in June 1982 and a Ph.D. in computer science from the University of Minnesota, USA, in January 1990. From September 1990 to January 2001 he was a Faculty member of the computer engineering department of King Fahd University of Petroleum & Minerals (KFUPM), Saudi Arabia (Assistant Professor from 1990 to 1995 and Associate Professor from September 1995 to January 2001). From February 2001 to June 2002, he was a Maitre de Conférences en informatique at the Faculté des Sciences de Monastir (FSM), University of Monastir, Tunisia. From July 2002 to August 2005, he served as the Director of the Institut Supérieur d’Informatique et Mathematiques of the University of Monastir. He is currently serving as a Professor of computer science and Director of the Institut Supérieur d’Informatique et des Technologies de Communication, Hammam Sousse, University of Sousse, Tunisia.
Dr. Youssef has over 150 publications to his credit in the form of books, book chapters, and journal and conference papers. He is the author with S. Sait of two books, (1) "VLSI Physical Design Automation: Theory and Practice", McGraw-Hill 1995, (also co-published by IEEE Press 1995), and reprinted with corrections by World Scientific in 1999, and (2) "Iterative Computer Algorithms with Applications in Engineering", IEEE CS Press 1999, and since 2003 published by John Wiley & Sons, which has also been translated into Japanese. His current research interests are computer networks, performance evaluation of computer systems, and algorithms for combinatorial optimization.

Rania MZID was born in Sfax, Tunisia 1985. She obtained her MS degree in new technologies of computer dedicated systems (2010) from the National School of Engineers of Sfax (Tunisia). She is currently a PhD student at the French Atomic Energy commission (CEA) Saclay and Computer embedded systems (CES) in

Tunisia (Sfax). She currently works on model driven engineering in real time systems.

Mohamed ABID received the Ph. D. degree from the National Institute of Applied Sciences, Toulouse (France) in 1989 and the “thèse d’état” degree from the National School of Engineering of Tunis (Tunisia) in 2000 in the area of Computer Engineering & Microelectronics. His current research interests include: hardware-software co-design, System on Chip, Reconfigurable System, and Embedded System, etc. He has also been investigating the design and implementation issues of FPGA embedded systems.
Actually, he occupies the post of director of doctoral school “Sciences & Technologies”, University of Sfax. He is member and Head of the research laboratory «Computer Embedded System » CES-ENIS, since 2006 (http://www.ceslab.org). He was member of “System on Chip at Computer, Electronic and Smart engineering system” Laboratory at ENIS 2001-2005. He was also responsible of «Hardware-Software co-design» research Group at EµM-Lab-FSM, Monastir-Tunisia, 1991-2000.
He is member of scientific committee at ENIS, since 2008, member of quality committee at ENIS, 2006-2007 and member of national committee of engineering pedagogy, since 2006. He was member and responsible of doctoral degree «computer system engineering» at ENIS, 2003-2010. He served in national or international conference organization and program committees at different organizational levels. He was also Joint Editor of Specific Issues in two International Journals and Joint editor of many conference’s articles nationals and internationals.
Dr. Abid is joint coordinator or an active member of several International Research and Innovation projects. He is Author or co-author of more than 30 publications in Journals and author or co-author of more than 180 papers in international conferences. He is also author or co-author of many guest’s papers, Joint author of many book’s chapters. Dr. Abid has served also as Guest professor at several international universities and as a Consultant to research & development in Telnet Incorporation.