A Pairing Identity based Key Management Protocol for Heterogeneous Wireless Sensor Networks

Manel Boujelben, Omar Cheikhrouhou, Mohamed Abid
CES research unit, National school of engineering, Sfax, Tunisia

Habib Youssef
PRINCE research unit, ISITC of Hammam Sousse, Sousse, Tunisia

Abstract— **Key management in wireless sensor networks is still a challenging problem. Most existing research considers symmetric cryptosystems to achieve key agreement assuming homogenous network architecture. However, these solutions suffer from performance bottleneck and poor scalability. Recently, deployed sensor network systems are increasingly following heterogeneous designs. In this paper, we propose a key management protocol for heterogeneous sensor networks based on an asymmetric cryptosystem named pairing identity based cryptography. We show that the proposed protocol has low communication and storage overhead. Moreover, our protocol assures key update and forward secrecy. We discuss the resilience of the proposed scheme against several types of attacks. Formal security validation, using the AVISPA tool, is used to demonstrate the privacy of the generated keys and the robustness of our protocol against several attacks.**

Keywords- Key managemet; Pairing; Sensor network

I. INTRODUCTION

Key management is an essential cryptographic primitive upon which other security primitives are built. Due to resource constraints, achieving such key agreement in wireless sensor networks (WSN) is non-trivial. Previous research on key agreement in sensor networks mainly considered homogeneous sensor networks organised in a flat architecture. However, despite the efficiency and the simplicity of flat Adhoc networks for sensor applications, recent research has indicated its limitations in terms of performance and scalability [1]. Furthermore, many security schemes designed for homogeneous sensor networks suffer from high communication and computation overhead, as well as high storage requirement [2]. Recently, hierarchical and heterogeneous networks have been proposed as an alternative topology to flat Adhoc topologies. This architecture considers a heterogeneous mix of nodes, a small number of powerful High-end sensors called H-sensors (e.g., PDAs or Imote of Intel), and a large number of Low-end sensors called L-sensors (e.g., MICA2 or TelosB). These nodes have different capabilities in terms of communication, computation, energy supply, storage space, reliability and other aspects. An heterogeneous architecture provides scalability, notable energy efficiency and security benefits [2][3]. We propose in this paper a key management scheme for this type of heterogeneous architecture. This proposed scheme uses one variant of public key cryptography, named Identity Based Cryptography (IBC), to establish pairwise keys between sensor devices. Public key cryptography has been considered too expensive for resources constrained sensor nodes. The traditional public-key algorithms (such as RSA [4]) require intensive computations which are not suitable for tiny devices. However, the recent progress on Elliptic Curve Cryptography (ECC) [5] provides new opportunities to use 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 [6], which demonstrates that the ECC public-key cryptography is feasible for sensor networks.

Since in-network processing techniques lend themselves to hierarchical network architectures, and given their several advantages, our proposed key management protocol adopts this technique. However, to support in-network processing techniques, and to secure message exchanges among sensor nodes, other types of keys must be established such as cluster keys and a group key.

So, the proposed key management protocol should not only support the establishment of pairwise keys but also other keys such as cluster keys or group key. A cluster key is a common key shared among all participants in the same cluster, and is mainly used for securing locally broadcast messages. A group key is a globally shared common key, used by the base station for the encryption of messages that are broadcast to the whole network. The proposed protocol consists then of pairwise key establishment protocol based on Pairing IBC, cluster key and group key transport protocol.

We discuss the taxonomy of attacks that threat the secure functioning of sensor networks and demonstrate how our proposed protocol can efficiently defend against several of them. However, node capture attack is still an open problem for sensor devices. To limit the impact of this attack, we preload each node with a bootstrapping time, which is the necessary time for a node to establish keys with its neighbours. Since node compromise is not a trivial task, usually a sensor node is designed to be able to sustain compromise for a certain time interval. However, the node bootstrapping phase is usually very short, and in practice it is reasonable to expect it to be shorter than the time needed to compromise the node [7]. The proposed solution for the secure functioning of the sensor devices is validated using the AVISPA tool.

The remainder of this paper is organized as follows. First, we discuss background and related work in Section 2. Section

3 presents the network model and the considered assumptions. Section 4 describes the proposed protocol for pairwise key establishment, cluster key establishment, and group key establishment. In Section 5, we discuss attacks that are imposed on sensor networks and how our protocol can efficiently defend against them. In Section 6, the validation tool AVISPA is used to formally prove the security of the proposed protocol. Finally, we conclude in Section 7 and give future work.

II. RELATED WORK

Many security proposals for WSNs have focused on efficient key management of symmetric encryption schemes for heterogeneous sensor networks. In [2], Du et al. use of the asymmetric pre-distribution which consists of pre-loading a large number of keys in each H-sensor while only pre-loads a small number of keys in each L-sensor. Similarly, Traynor et al. [3] 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.

Recent research for distributing keys in WSN goes toward the asymmetric approaches which rely on public key cryptography. RSA [4] stands for Rivest, Shamir, and Adleman algorithm is one of the most popular public-key encryption algorithms currently available. Its security is based on the difficulty of solving the Integer Factoring Problem. ECC [5] was then developed by Koblitz and Miller. ECDLP "Elliptic Curve Discrete logarithm Problem" is one variant of ECC used to compute pairwise keys. It is based on the difficulty of solving the Discrete Logarithm Problem "DLP" 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 level of security as a 1024-bit RSA key [6].

Many cryptographic applications based on elliptic curves use bilinear pairing, a powerful mathematical tool which enables efficient implementation of IBC. Bilinear pairing allow Diffie-Hellman problem, a well-known class of hard computational problems, to have an easy decisional version. Since Boneh et al. proposed an ID-based signature scheme [8] and encryption scheme [9] 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, the pairing cost is no longer a heavy burden on sensor nodes. In deed, Oliveira et al. present TinyPBC [10], a protocol for authenticated identity-based non-interactive key distribution in sensor networks based on pairing. They demonstrate that TinyPBC takes only 5.45s to compute pairings on ATmega128. Also, the requirements in terms of RAM as well as ROM are 2,687,368 and 47,948 bytes, respectively. So, this implementation demonstrated that the pairing is feasible for resource constrained sensor networks. So, given the several advantages of pairing IBC and the efficient implementation results, our proposed protocol uses pairing to establish keys between the set of sensor nodes. In addition, we adopt heterogeneous network architecture to ensure better performances and scalability.

III. NETWORK MODEL AND ASSUMPTIONS

A. Network model

We consider a heterogeneous sensor network (HSN) consisting of numerous L-sensors deployed with a small number of H-sensors (see Figure 1). The general function of an L-sensor is to collect raw data and forward it to the corresponding H-sensor, designed as cluster head. This traffic represents the intra cluster traffic. The inter cluster traffic is made by the H-sensors. H-sensors make data aggregation of information flows coming from L-sensor nodes, and forward the aggregated data toward the BS either directly or via other CHs. In addition, the network includes a BS which is the ultimate destination of data streams from all the sensor nodes. The BS may be connected to an outside network.

B. Assumptions

We assume a static sensor network, where all sensor nodes have fixed locations. 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 trusted and will never be compromised. It is equipped with tamper resistant material. Also, H-sensors should be equipped with tamper resistant hardware and intrusion detection systems to detect malicious behaviours.

IV. PROPOSED KEY MANAGEMENT AND AUTHENTICATION PROTOCOL

Different keys should be computed between the set of nodes and the base station. The proposed key management protocol is based on the difficulty of solving the DLP on elliptic curves. It builds on IBC and the Tate pairing. The Tate pairing essentially takes two points on an elliptic curve and maps them to an element of a multiplicative group of a large finite extension field. It has been shown that an IBC scheme is preferable to an ECC one because, it does not require the transmission of digital certificates and there is a simple process for generating symmetric keys. Another advantage of IBC is the possibility of scaling to the number of nodes in the network. Moreover, as in public key cryptosystems, although the number of nodes increases, each node has to store only one secret key. In addition, the proposed protocol allows for updating keys after a certain time to avoid cryptanalysis or after a node compromise. Our protocol is composed of two phases. The first phase takes place before deployment. At this phase, the BS preloads each node with corresponding information to be used to generate keys. After deployment, the second phase begins to set up the network structure and to establish pairwise keys between neighbouring nodes, cluster keys for each group of nodes in the same cluster, and a group key for all the nodes in the network.

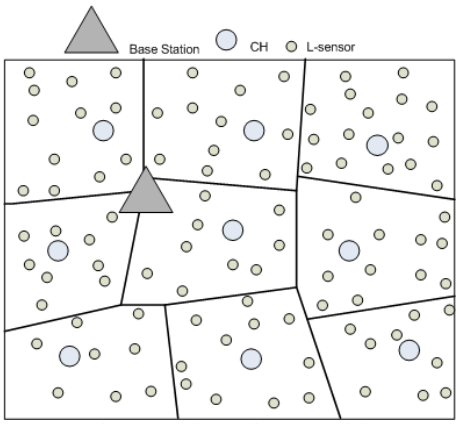

Figure 1. Clustered HSN model.

A. Pre-deployment phase

During the bootstrap of the network, the BS (a laptop or a human controller) must perform these 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. 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 is long enough such that each node can establish pairwise keys with all its neighbours and also cluster keys and group key.
- Then the BS computes a private key $S_i = sQ_i = sH_1(ID_i)$ for each node i, which is assigned to it prior to deployment. So, the system parameters are $< G_1, G_2, e, q, P, P_{pub}, H_1, H_2 >$.

B. After deployment phase

When the nodes are deployed, they must establish their pairwise keys, the cluster keys, and the group key to secure their communications. However, before doing this, each node must know with which node it may communicate. Therefore, all the nodes must go through the cluster formation phase and the neighbourhood discovery phase before establishing keys. When these two phases end, each node initialises a timer that expires when the bootstrapping time is finished.

1) Cluster formation phase: During the network initialisation, each H-sensor broadcasts a Hello message using the maximum power to nearby L-sensors with a random delay. The transmission range of the broadcast is large enough so that most L-sensors can receive Hello messages from several H-sensors. Each L-sensor sets a timer after receiving the first Hello message. When the timer expires, each L-sensor chooses the H-sensor whose Hello message has the best signal strength as cluster head and sends an acknowledgment reply message. Each L-sensor also records other H-sensors from which it receives the Hello messages, and these H-sensors serve as backup cluster heads in case the primary cluster head fails. If an L-sensor does not receive any Hello message during the initialization phase, it actively looks for a nearby H-sensor by broadcasting an Explore message. At the end of this phase, the network is divided into multiple clusters. The H-sensor acts as a cluster head (CH), and the L-sensors act as cluster members; more details about the clustering scheme can be found in [11].

2) Neighborhood Discovery Phase: After cluster formation, L-sensors go through the neighbourhood discovery phase [12]. First, each L-sensor broadcasts a Hello message for a short range in order to discover neighbors. A neighbor L-sensor, belonging to the same cluster, which receives the Hello message acknowledges with HelloReply message. Then the sender adds its ID in its neighbors list. The neighborhood discovery phase ends when all the L-sensors have obtained neighborhood information. After the network deployment, every node can then keep a list of all neighboring nodes.

3) Establishing pairwise keys: Pairwise keys should be computed between any pair of nodes N_i and N_j in the network. Node N_i generates key $V_{i,j} = e(S_i, Q_j) = e(Q_i, Q_j)^s$ which is used as a message authentication code (MAC) 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:

- 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 R_i and M_i.
- 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 R_j and M_j.

$$N_i \to N_j : R_i, M_i$$
$$N_j \to N_i : R_j, M_j$$

- Upon receiving the message, N_i verifies the MAC and computes the pairwise key $K_{i,j} = H_2(r_i R_j) = H_2(r_i r_j P)$.
- Similarly, N_j verifies the MAC and computes the pairwise key $K_{i,j} = H_2(r_j R_i) = H_2(r_i r_j P)$.

4) Establishing cluster key: The cluster key establishment phase follows the pairwise key establishment phase, and the process is very straightforward. In the beginning of this phase:

- Each cluster head CH_i generates a random key $KC_i \in Z_q^*$.
- Then, for each L-sensor node j in the range, CH_i computes a MAC $M_{i,j} = MAC\ K_{i,j}(CH_i, L_j, T_i, KC_i)$ and the encrypted cluster key $E_{i,j} = \{KC_i\}K_{i,j}$ with $K_{i,j}$ the pairwise key shared

between H_i and L_j and T_i a fresh time stamp and then sends this message:

$$CH_i \rightarrow L_j : E_{i,j}, T_i, M_{i,j}$$

- Upon receiving this message at the time T_j, L_j verifies whether $(T_j - T_i) \leq \Delta T$. If it holds, L_j accepts H_i's message, where ΔT is an expected valid time interval. Then, L_j decrypts it to obtain the key KC_i and stores it in its table.

- Next, L_j retransmits the cluster key, in the same way, using the pairwise key shared with each L-sensor in its neighborhood. This process is repeated until all the nodes in the cluster receive key KC_i.

$$L_j \rightarrow L_k : E_{j,k}, T_j, M_{j,k}$$

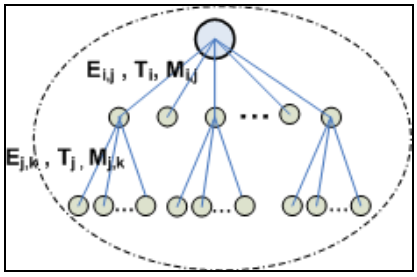

Figure 2. Establishing cluster key.

5) Establishing group key: The group key, generated by the BS, must be distributed to all the remaining nodes in a secure, reliable, and timely fashion, using hop-by-hop translation.

- The BS generates a random key $KG \in Z_q^*$.
- Then, for each CH_i in the range, the BS computes a MAC $M_{BS,i} = MAC\ K_{BS,i}(BS, CH_i, T_{BS}, KG)$ and the encrypted group key $E_{BS,i} = \{KG\}K_{BS,i}$ with $K_{BS,i}$ the pairwise key shared between BS and CH_i, and T_{BS} a fresh time stamp, then sends this message:

$$BS \rightarrow CH_i : E_{BS,i}, T_{BS}, M_{BS,i}$$

- If one CH is not in the range, it asks a neighboring CH to send him the group key in the same way using pairwise keys shared between CHs.
- Upon receiving this message at the time T_i, CH_i verifies whether $(T_i - T_{BS}) \leq \Delta T$. If it holds, CH_i accepts BS message, where ΔT is an expected valid time interval.
- Then, CH_i encrypts KG using its correspondent cluster key and broadcasts the message to all the nodes in the local cluster signed with the private key S_i of CH_i.

$$CH_i \rightarrow * : \{KG\}KC_i, T_i, signature$$

With T_i a fresh time stamp

- Upon receiving this message at the time T_j, an L-sensor node L_j verifies whether $(T_j - T_i) \leq \Delta T$. If it holds, L_j accepts the message of CH, where ΔT is an expected valid time interval. L_j decrypts the message and stores it. Then, L_j rebroadcasts the received message modifying only the time stamp and the signature. This process continues recursively until all the nodes in the cluster receive the group key.

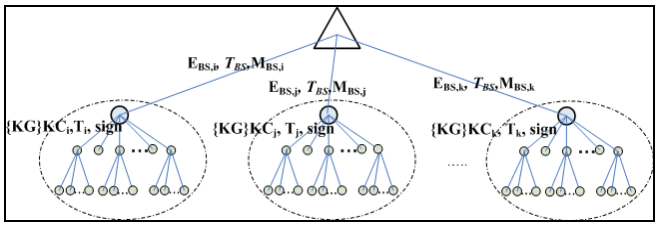

Fig.ure 3. Establishing group key.

V. 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, the transmitted cluster or group key. This operation threatens message confidentiality. Modifying messages threatens message integrity. Finally, fabricating false messages threatens message authenticity.

An adversary cannot know the value of 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 key or group key, their confidentiality is guaranteed. Modifying keying materials or keys and injecting false messages is prevented as MACs and signatures are used. Regarding the fact that every node in the network go through the key discovery and authentication phase, these attacks are also prevented for ordinary messages.

- 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 a DLP to get the random r_i or r_j. For the cluster and group key, this attack is prevented as in each transmitted message there is a time stamp 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 attack will be limited 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 [13].

After deployment, the proposed protocol goes through the cluster formation and neighborhood discovery phases. At the end of these phases, each node would keep a list of its immediate neighbours. 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 neighborhood, that it is a near one.

VI. FORMAL SECURITY VALIDATION

Many formal security validation methods and tools are proposed in the literature. In this paper we choose AVISPA, "Automated Validation of Internet Security Protocol and Application" [14], for the following reasons: (1) it is enough expressive and we can model several property like secrecy of keys, authentication, freshness, robustness against replay attacks, etc, (2) it provides a user friendly specification language, the HLPSL "High Level Protocol Specification Language", which is the input of the AVISPA tool, and (3) it is widely used by developers of security protocols and by academic researchers to analyse possible attacks on security protocols. With AVISPA we were able to successfully verify the most important properties:

- Secrecy of keys: this means that the key used during the protocol exchange must remain secret between the two parties and not revealed to a third one. This property can be specified in AVISPA with the following primitive "secret (Key, attribute_of_key, Agents)".
- Authentication of message source: this property allows a receiver to verify that the data has really been sent by the claimed sender. This property can be verified in AVISPA with the following two primitives: witness(Ni, Nj, at_ri, RPi') in the sender role and request(Nj, Ni, at_ri, RPi') in the receiver role.

The following two properties are implicitly verified in AVISPA: replay attack protection and man in the middle protection. The verification results imply that our proposed pairwise key establishment protocol can provide a significant level of security. Thus, the cluster key and the group key establishment protocols, built upon the pairwise key establishment protocol, will be secured.

VII. CONCLUSION

In this paper, we propose a key management protocol for heterogeneous sensor networks based on pairing and identity based cryptography. This protocol includes pairwise key establishment and also cluster keys and group key transport protocol. Robustness of the proposed protocol against several types of attacks is discussed and verified formally using the AVISPA security validation tool. Implementation and performance analysis are considered in future work.

REFERENCES

[1] 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.

[2] X. Du, Y. Xiao, M. Guizani, H.H. Chen, "An Effective Key Management Scheme for Heterogeneous Sensor Networks", Ad Hoc Networks, Elsevier, vol. 5, issue 1,January 2007, pp. 24–34.

[3] P. Traynor, H. Choi, G. Cao, S. Zhu, T. Porta, "Establishing pair-wise keys in heterogeneous sensor networks", in: Proceedings of IEEE INFOCOM 06.

[4] 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.

[5] N. Koblitz, "Elliptic curve cryptosystems," Mathematics of Computation 48, 1987, pp. 203-209.

[6] 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.

[7] Y. Zhou, Y. Zhang, and Y. Fang, "Access Control in Wireless Sensor Networks," Elsevier Ad Hoc Networks 5(1): 3-13 (2007)

[8] D. Bonech, 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.

[9] 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.

[10] L B. Oliveira, M. Scott, J. Lopez, Ri. Dahab. TinyPBC: Pairings for Authenticated Identity-Based Non-Interactive Key Distribution in Sensor Networks. 5th International Conference on Networked Sensing Systems (INSS'08). June 2008, Kanazawa/Japan (pages. 173-179).

[11] X. Du and F. Lin, "Maintaining Differentiated Coverage in Heterogeneous Sensor Networks," EURASIP Journal on Wireless Communications and Networking, Issue 4, 2005, pp. 565-572.

[12] F. Kausar, S. Hussain, L. T. Yang, and A. Masood, "Scalable and efficient key management for heterogeneous sensor networks," The Journal of Supercomputing, vol. 45, no. 1, pp. 44–65, Feb 2008.

[13] 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.

[14] AVISPA Project. AVISPA protocol library. Available at http://www.avispa-project.org/.