Establishing pairwise keys in heterogeneous two-tiered wireless sensor networks

Manel Boujelben, Omar Cheikhrouhou, Mohamed Abid
CES research unit, National school of engineering, Sfax, Tunisia
boujelben_manel@ieee.org, Omar.cheikhrouhou@isetsf.rnu.tn, mohamed.abid@enis.rnu.tn

Habib Youssef
PRINCE research unit, ISITC of Hammam Sousse, Sousse, Tunisia
habib.youssef@fsm.rnu.tn

Abstract— **Key management is an important cryptographic primitive upon which other security services are built. Most of the existent pairwise key management protocols consider homogenous Adhoc sensor networks. However, recent researches have demonstrated the limitation of this architecture in terms of performance and scalability. Therefore, this paper focuses on proposing a pairwise key management protocol for heterogeneous multi-tiered wireless sensor network that leverages the existence of a small percentage of more capable sensor nodes. The heterogeneity among sensor nodes helps provide scalability, notable energy efficiency and security benefits. The proposed protocol uses both probabilistic key pre-distribution in the lower tier of the network architecture and public key cryptography in the upper tier to distribute session keys. The performance of our solution is evaluated analytically with respect to communication, computation, and memory overhead. We demonstrate that the proposed key establishment protocol provides a high level of security while at the same time minimizes the resource consumption of the sensor devices. These results are highlighted when comparing our proposed protocol with three well known key establishment protocols implemented for WSN.**

Keywords- ***Security, Heterogeneous Sensor networks, Random key pre-distribution, Pairing***

I. INTRODUCTION

Wireless Sensor Networks (WSNs) are widely used in several applications such as military, healthcare, automotive, research, and so on. Many applications are often deployed in potentially adverse or even hostile environments, so they are dependent on the secure operation of a sensor network, and have serious consequences if the network is compromised or disrupted. To achieve security in wireless sensor networks, it is important to be able to encrypt and authenticate messages sent among sensor nodes. Keys for encryption and authentication purposes must be agreed upon by communicating nodes. Due to resource constraints, achieving such key agreement in wireless sensor networks is nontrivial. Sensor nodes are battery powered and equipped with integrated sensors, low data processing capabilities, and short-range radio communications. So, efficient key management mechanisms are needed besides lightweight ciphers.

Previous research on key agreement in sensor networks mainly considered homogeneous sensor networks organised in a flat architecture. 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 networks for sensor applications, recent researches have indicated the limitations of the flat topology in terms of performance and scalability [1]. Furthermore, many security schemes designed for homogeneous sensor networks suffer from high communication and computation overhead, and high storage requirement [2]. Recently, hierarchical or multi tiered 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 [3]) and a large number of Low-end sensors called L-sensors (e.g., MICA2 or TelosB [4]). L-sensors are in charge of performing simpler tasks, such as making physical measurements, while resource-rich, high-power H-sensors take on more complex tasks. A multi-tiered heterogeneous architecture provides scalability, notable energy efficiency and security benefits [5][6].

To exploit these advantages, we consider a two tiered heterogeneous sensor network when designing the proposed pairwise key management protocol. The process of establishing pairwise key at the lower tier of our architecture presents fundamental differences with the process that should be used at the upper tier. The lower tier is composed of many L-sensors clustered around few H-sensors. On the other hand, the upper tier is composed of many H-sensors designed as cluster heads "CH" communicating with the base station "BS". Given the difference between the two tiers in terms of capabilities, energy, and security requirements, we propose to use a different key management protocol at each tier. In the lower tier, in order to deal with resource limited devices, we use the random unbalanced key distribution method[7] combined with Blom scheme [8] to improve network resilience against node capture while respecting the sensor nodes resource constraints. In the upper tier, nodes are powerful and can perform the very secure asymmetric cryptographic

algorithms. So, for the upper tier, we apply the pairing Identity based cryptography to distribute pairwise keys.
We evaluate analytically the resource requirements of our protocol in terms of storage overhead, computation overhead, and communication overhead and we compare it with three key establishment protocols implemented specially for WSN. The remainder of this paper is organized as follows. First, we discuss background and related work in Section 2. In Section 3, we present the network model and the considered assumptions. Section 4 describes two proposed protocol for pairwise key establishment, one for each tier, and give results of overhead analysis with respect to memory, communication and computation. In Section 5, we present a comparison of our proposed protocol with three well known key establishment protocols. Finally, we conclude in Section 6.

II. RELATED WORK

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.
Blom [8] and Blundo et al. [9] proposed respectively a matrix and a polynomial key generation schemes. These schemes are λ-secure, meaning that the coalition of less than λ+1 sensor nodes can not reveal the pairwise keys of others. Another important key establishment scheme relying on probabilistic key pre-distribution technique was first published by Eschenauer and Gligor [10]. This approach consists of assigning to each sensor node a random subset of keys from the key pool before deployment. After deployment, any two nodes will be able to find one common key used as their shared secret to initiate communication. Moreover, Du et al. developed a pairwise key management scheme [11]. This scheme combines the random key pre-distribution scheme [10] and the Blom scheme [8] to substantially improve network resilience against node capture over existing schemes, without increasing the memory overhead. However, one of the important mechanisms in sensor networks, in-network processing, is not considered in the previous schemes. So, hierarchical key management solutions are proposed as LEAP [12]. All previously described solutions consider a network with a homogenous set of nodes. Researches have shown the limitation of this architecture and its high overhead. Recently deployed sensor network systems are increasingly adopting heterogeneous designs. In [2], Du et al. considered key management for Heterogeneous Sensor Network (HSN). Their basic idea is the use of the asymmetric pre-distribution which consists of pre-loading a large number of keys in each high end sensor while pre-loads only a small number of keys in each low end sensor. Similarly, Traynor et al. [7] 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 does not only provide a significant level of security but also reduces the consequences of node compromise.
Another major approach for distributing keys in WSN is the asymmetric approaches which rely on public key cryptography. RSA 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 "Elliptic curve cryptography" was then developed. 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 DLP on elliptic curves. ECC can obtain the same security level as RSA while using smaller keys [13]. Many cryptographic applications based on elliptic curves use bilinear pairing, a powerful mathematical tool which enables efficient implementation of Identity Based Cryptography "IBC". ECC-based schemes such as IBC seem to be the best approach for distributing symmetric keys in a WSN. In spite of all its advantages, they are still public key cryptosystems and thus they are orders of magnitude more complex than symmetric cryptosystems. To profit from the multi tiered architecture of our network and from the heterogeneity of nodes, we can use the IBC in the upper tier which is composed of powerful nodes. These nodes can support the intensive calculations required by this public key scheme. Also, this scheme provides a significant level of security needed in the upper tier. However, in the lower tier, using this scheme will affect the resources of L-sensors. Therefore, symmetric key pre-distribution schemes are more suitable to establish keys in this tier. Giving the relevant advantages of the unbalanced key pre-distribution cited before, we apply this approach combined with the matrix key based scheme in the lower tier to take advantage of the security level while respecting the nodes constraints [14].

III. NETWORK MODEL & ASSUMPTIONS

A. Network Model

The two-tiered heterogeneous WSN model assumed is shown in Figure 1. In this architecture, numerous L-sensors are deployed together with a small number of H-sensor. In addition, the network includes a BS which is the ultimate destination of data streams from all the sensor nodes. The BS is connected to an outside network. Each H-sensor has more memory ability than an L-sensor. It is equipped with a high-end embedded processor and is capable of communicating over long distances. The general functions of H-sensor are: (1) data aggregation of information flows coming from L-sensor nodes, and (2) forwarding the aggregated data to the next hop H-sensor toward the administrative H-sensor designed as CH.

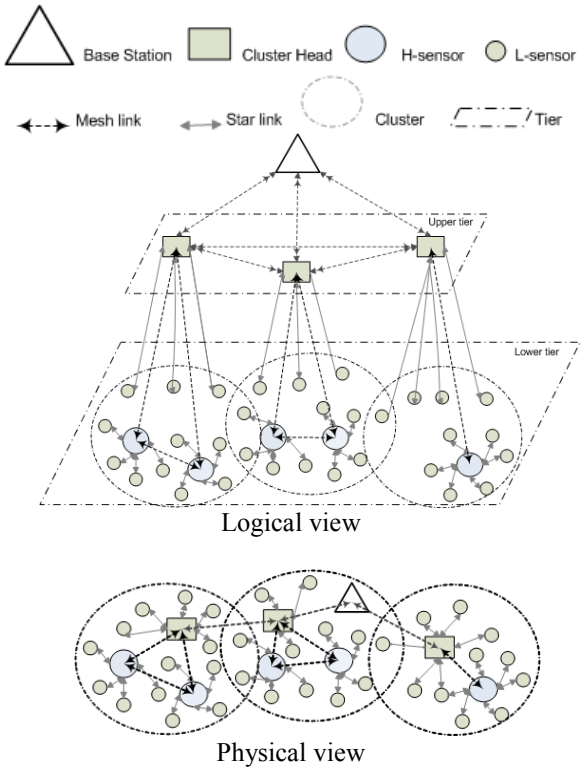

Figure 1. Two-tiered heterogeneous WSN model.

An L-sensor is a battery-powered node equipped with a low-end processor and mechanisms for short-range radio communications at low data rates. The general function of the L-sensor is to collect raw data and forward it to the corresponding H-sensor. The lower tier of the network consists of multiple clusters, where each cluster is composed of numerous nodes clustered around an administrative CH. The network of CHs forms the upper tier. The inter-clusters communication will be done by CHs which aggregate data from their local clusters and send them to the BS either directly or via other CHs in the range.

B. *Assumptions*

We assume a static sensor network, where all sensor nodes have fixed locations. 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 to protect their supplementary keying materials from compromise. In addition, intrusion detection systems should be integrated in H-sensors to detect malicious behaviours. An L-sensor communicates only with H-sensors and never with another L-sensor. For precision, each L-sensor can establish a link only with two H-sensors, one by default and another as back up in the case where it can not establish a link with the first. Sensor nodes are organised in clusters and each node knows to which cluster it belongs and have a list of its immediate neighbours.

IV. PAIRWISE KEY MANAGEMENT PROTOCOL

Pairwise keys must be established between the pairs (L-sensor, H-sensor) or (H-sensor, H-sensor) in the same cluster to ensure intra-cluster secure communication and between the pair (CH_i, CH_j) to ensure inter-cluster communication and finally between (CH_i, BS) to report the final aggregated data. In the first subsection, the proposed protocol uses the combination of the unbalanced key pre-distribution method and the Blom scheme to establish pairwise key among L-sensors and H-sensors nodes at the lower tier. In the second subsection, we demonstrate how IBC is used for key management between CHs and BS at the upper tier.

A. *Tier 1 key agreement protocol*

Blom's scheme was first proposed for key establishment in wired networks and then applied for sensor networks. Recently, the unbalanced key pre-distribution method was proposed for heterogeneous sensor networks. Therefore, the proposed key agreement protocol designed for this tier builds upon the Blom scheme and applies on it the unbalanced key pre-distribution method. Hence, the proposed protocol can not only respect the memory and processing constraints on nodes but also give a significant level of security by reducing the consequence of node compromise.

1) *Overview of Blom's scheme*

In [8], Blom proposed a key pre-distribution scheme that allows any pair of nodes to find a common secret. This scheme is λ-secure, i.e., as long as the number of nodes compromised by an adversary is less then or equal to λ, the uncompromised nodes remain secure. When an adversary compromises more than λ nodes, all pairwise keys of the entire network are compromised. Compared to the full pairwise scheme, this scheme is efficient in terms of memory consumption. Each node in a network of size N needs to store only λ+1 elements, with λ<< N. However, this scheme is not perfectly resilient against node capture. To increase its resilience, the security parameter λ must be high, resulting in a large memory overhead.

2) *Overview of unbalanced key pre-distribution method*

Heterogeneous sensor networks present a number of advantages. First, the presence of nodes with additional capabilities reduces the difficulty of implementing secure systems. The ability of the more capable nodes to store extra keying data allows for critical memory to be saved. Based on these arguments, Traynor et al. [7] and Du et al. [2] proposed that security can be provided by leveraging an unbalanced distribution of symmetric keys. In so doing, this solution for key distribution allows systems to operate securely while minimizing the security burden placed upon the least capable nodes in the system.

3) Multiple space unbalanced random key pre-distribution

The proposed scheme is composed of three phases: the key pre-distribution phase, the shared key discovery phase and the path key establishment phase. These three phases will be applied to each cluster independently, so that each cluster will have its own spaces and can choose different level of security and connectivity depending on the chosen parameters (λ, k, m). This fact can improve the security of our scheme by reducing the impact of attacks to the local cluster.

a) Key Pre-distribution Phase: This phase is composed of three steps:

-*Step 1 (Generating G matrix):* To generate the matrix G of size (λ +1)×N, we must chose a primitive element from a finite field F_q, where q is the smallest prime larger than the size of the key, and q > N. Figure 2 shows an example of matrix G. Let s be a primitive element of F_q. A node k in the network must carry the k^{th} column of G. Using this generator matrix each node will only carry the seed s^k.

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ s & s^2 & \dots & s^k & \dots & s^N \\ s^2 & (s^2)^2 & \dots & (s^k)^2 & \dots & (s^N)^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ s^\lambda & (s^2)^\lambda & \dots & (s^k)^\lambda & \dots & (s^N)^\lambda \end{pmatrix}$$

Figure 2. Example of matrix G.

-*Step 2 (Generating D matrix):* This step consists of generating the ω spaces. Each space consists of a tuple $S_i=(D_i.G)$, i=1… ω, where D_i is a random symmetric matrix of size (λ+1)×(λ+1). Then, ω matrices $A_i= (D_i.G)^T$, must be computed. Let $A_i(j)$ represent the j^{th} row of A_i. According to Blom's scheme, each space has the "λ -secure" propriety.

-*Step 3 (Selecting spaces):* Each L-sensor node and H-sensor node must choose respectively k spaces and m spaces randomly from the space pool of size ω (m>>k). For any chosen space i, the node will carry a row from the matrix A_i, and this row must be secret and should not be revealed to other nodes.

b) Shared key discovery phase: After deployment, each node must carry the corresponding seed of matrix G as node ID, the different chosen spaces, and the corresponding secret rows of matrix Ai. According to Blom's scheme, two nodes can find a common secret key if they have both picked a common space. Therefore, to communicate with its neighbours, each node must broadcast a message containing its ID and the spaces it carries. If a neighbour receiving the message finds that it shares a common space with the sender, they can compute their shared secret key using Blom's scheme (See Figure 3).

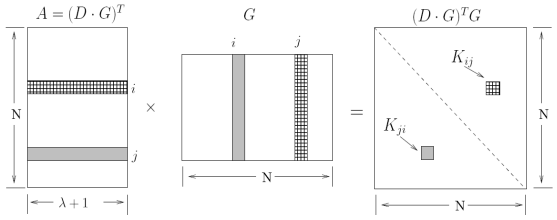

Figure 3. Blom's scheme.

c) Path key establishment phase: A link exists between two nodes only if they share a common space.In the other case, two neighboring nodes must establish a pairwise key by using a multi-hop path through secure and trusted H-sensors. The key can be generated at one of the endpoints and passed through this path such that the two end nodes can communicate directly. At the completion of this final phase, all nodes within transmission radius of each other should be able to communicate securely.

4) Overhead analysis

a) Computational Overhead: Computing a pairwise key requires 2λ modular multiplications and λ modular additions in a finite field F_q, where q is a prime number that is large enough to accommodate a cryptographic key. To do so, each node must, in the first step, regenerate the corresponding column of matrix G from a seed. So λ-1 modular multiplications must be computed. In our scheme, the exponentiation can be evaluated in a much cheaper way. This is mainly due to the observation that the identity of the node is used to seed this operation rather than a large random number from F_q. So IDs can be chosen from a different finite field $F_{q'}$ where q' is a prime number that is larger than the maximum number of sensors but much smaller than a typical q. In the second step, the node must compute the product of the corresponding row of matrix A_i with the generated column of G. This operation involves λ+1 additional modular multiplications and λ additions. Since the seed is the ID of a sensor, the modular multiplication is always performed between an integer in F_q and another integer in $F_{q'}$. For typical cryptosystems such as RC5, q should be at least 64 bit long. For example, when we choose λ=30 and q'=16 bits, the total number of multiplication is 60. So, 30 16-bit modular multiplications and 30 modular multiplications of a 64-bit integer and a 16-bit integer must be computed. Compared to RSA 1024 signature signing scheme, where the exponentiation requires at least 1024 multiplications, the proposed scheme introduces a very low computation overhead.

In this paragraph, we compare the computational overhead in L-sensor and H-sensor nodes. In the case where a key match exists, an L-sensor node, which is resource constrained, makes this operation only once with a neighboring H-sensor with which it carries a common space. An H-sensor does this key agreement phase with its d neighbors. Let's compute the probability of an H-sensor and an L-sensor having at least one space in common, using ω

the size of the space pool and k (respectively m) the size of the space ring in L-sensor (respectively H-sensor). We obtain the following equation [7]:

$$P[match] = 1 - \frac{(\omega - k)!(\omega - m)!}{\omega!(\omega - m - k)!} = U \qquad (1)$$

If there is not a key match between two nodes, communication and encryption overhead will occur in the secure path to lead the key to the intended node. Communication is detailed in the next subsection and the encryption overhead depends on the cryptographic algorithm and the number of neighbours.

We consider a discrete uniform distribution of L-sensor nodes using an H-sensor node in the path key establishment phase. The probability that any H-sensor node is used by an L-sensor in the path key establishment phase is given by:

$$P = (1 - U) \qquad (2)$$

Because there are a fixed number of L-sensors in the neighbourhood of an H-sensor, we can compute the average number of encryptions/decryption done by an H sensor, and given t the number of L-sensors in the range:

$$M = \sum_{t} t(1 - U) \qquad (3)$$

So, the total cost for computing a pairwise key is:

In L-sensor:

$$C = 2\lambda Mul + \lambda Add \qquad (4)$$

In H-sensor:

$$C = d(2\lambda Mul + \lambda Add) + MEnc / Dec \qquad (5)$$

b) Communication Overhead: In terms of communication overhead, there is none for direct key establishment. In this subsection, we study the communication overhead of our key pre-distribution scheme when two neighboring nodes cannot find a common key. As we have discussed before, when this situation occurs, these two nodes have to find an indirect path between them using one or many H-sensors. The shorter the length of the path the better it is. We use P(g) to denote the probability that the number of H-sensors needed to connect two neighboring nodes is g.

Because the number of keys in H-sensor nodes is very high, both theory and practice demonstrate that H-sensor nodes are able to communicate directly with each other with a far greater than five-nines reliability (0.99999). Therefore, direct secure links between H-sensors will always be assumed. In either case, if an L-sensor has not a direct match with the intended H-sensor, the probability that an L-sensor uses g H-sensors in the indirect path is:

$$P(g) = 1 - (1 - U)^{g+1} \qquad (6)$$

With g is the number of H-sensors in the path.

Two H-sensors have a high probability to find a key match. So communication overhead required for establishing a pair-wise key between two neighboring H-sensors is negligible. Regarding the fact that there is a lower probability to find a key match between a pair (L-sensor, H-sensor), the communication overhead will be more important but it will not affect the resources of the constrained L-sensor nodes. Given the adopted network connectivity model, if an L-sensor node can't find a common space with the local H-sensor, it may establish a session key through a path of one or more of the other H-sensor nodes. So the communication burden will be almost placed on H-sensors.

This overhead (communication and computation) occurs only once in the first installation of the network or after a node is revoked, so the cost is not significant. We can also choose a larger k and m to lower the communication overhead but this can degrade the resilience of the network.

c) Memory Overhead: For each chosen space, a node must carry (λ+1) elements to be able to compute a key according to Blom's scheme. Given that the network is composed of a heterogeneous set of nodes and that each L-sensor node carries k spaces while each H-sensor node carries m spaces, the total memory usage T is:

For L-sensor node:

$$T = k(\lambda + 1) \qquad (7)$$

For H-sensor node:

$$T = m(\lambda + 1) \text{ with } m >> k \qquad (8)$$

B. Tier 2 key agreement protocol

In tier 2, pairwise keys should be computed between CHs and the BS. The proposed key agreement protocol for this tier 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. Since Boneh et al. proposed an ID-based signature scheme [15] and encryption scheme [16] from the pairing; many schemes using pairing have been proposed. It has been shown that an IBC scheme is preferable than an ECC one because it does not require the transmission of digital certificates. In fact, by using the IBC, the wireless sensor node identity can be used as the public key and hence there is no need for a certificate to bind a wireless sensor node identity to its public key.

1) Initialization

During the bootstrap of the network, the BS must perform these extra tasks. First, 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 a collision resistant cryptographic hash function H_1 is determined, where $H_1 : \{0,1\}^* \to G_1$, mapping from arbitrary-length strings to points in G_1. The BS picks a random number $s \in Z_q^*$ and then sets $P_{pub} = sP$. The base station keeps *s* which is known only to it and no other node in the network will know the value of s. The BS uses the

value of s to generate secret keys of CHs. For each cluster head CH_i, it computes a private key $S_i=s\ Q_i=s\ H_1(ID_i)$, which is assigned to it prior to deployment. So, the system parameters are <G1, G2, e, q, P, P_{pub}, H_1 >.

2) Key establishment between BS and CH_i

The BS generates key $K_{BS,CHi} = e(P,Q_i)^{s^2}$ which is used as a pairwise key between BS and CH_i. In the same way, CH_i generates $K_{BS,CHi} = e(P_{pub},S_i) = e(P,Q_i)^{s^2}$.

3) Key establishment between CH_i and CH_j

Cluster head CH_i generates the key $K_{CHi,CHj} = e(S_i,Q_j) = e(Q_i,Q_j)^s$. This key is used as a pairwise key between CH_i and CH_j. In the same way, CH_j generates $K_{CHi,CHj} = e(Q_i,S_j) = e(Q_i,Q_j)^s$.

4) Overhead analysis

This key management introduces significant computational overhead, but this will not affect the node resources as it is implemented on powerful nodes (BS, H-sensors).

a) Computational overhead: The main component which must be computed to execute this protocol is the Tate pairing. The Tate pairing e(P,Q) calculation is a very computationally demanding component. It requires a big number of arithmetic operation (addition, multiplication, and inversion) done on large integers. Evaluating pairing was a very complex and costly operation. However, since Barreto et al [17] proposed an efficient pairing algorithm working on super singular curves, the pairing cost is no longer a heavy burden on sensor nodes. In deed, authors of [18] demonstrated the possibility of efficiently implementing the Tate pairing even in L-sensor nodes. In fact, they present results of computing pairing over the constrained Micaz nodes [4]. The memory cost in terms of RAM and ROM spaces are respectively 1,831 and 18,384 bytes. The average execution time for computing the pairing is 30,21s. In our proposed protocol, we apply the Tate pairing in H-sensor nodes. Therefore, it will not result in a high computational overhead on these powerful nodes. Moreover, if a software implementation requires too much energy with unacceptable latency, a software/hardware implementation of this scheme should be investigated.

b) Communication overhead: To establish a pairwise key between two nodes (CH_i, CH_j) or (CH_i, BS), there is zero message to be sent. Most of the energy consumed in a sensor network is due to the message transmitting process. The proposed protocol does not involve any key distribution messages and therefore significantly reduces the energy consumption profile.

c) Memory Overhead: Each CH_i must store its corresponding pair of key (public key Q_i, secret key S_i) and the public key of the BS. Assuming that the size of the key is 64 bits, each node has to store 192 bits which is a little amount of data to be stored in H-sensor nodes.

V. COMPARISON WITH OTHER PROTOCOLS

In this section, we will deal with presenting the differences between the proposed protocol and the random key pre-distribution protocol [10], the multiple space random key pre-distribution protocol [11], and the unbalanced random key pre-distribution protocol [7] in terms of network model, key establishment approach, and security and scalability levels in one hand. On the other hand, a comparison study of these protocols is given with respect to computation, memory, and communication overheads.

Table 1 shows these comparative results. We can notice that our proposed protocol gives a high level of security and scales well while at the same time respects the storage, computation and communication constraints of sensor nodes. In fact, the heavy burden is always placed upon the more capable H-sensor nodes. Hence, it is well suited for heterogeneous wireless sensor network applications.

VI. CONCLUSION

Key management poses a main concern for security operation in sensor network. Many key management protocols are proposed for homogeneous sensors networks. However these networks have limited performance and security. Heterogeneous sensor network are proposed to outcome these drawbacks. This paper proposed a pairwise key management for a two tiered HSN. The observation that we have made is that each tier has different security requirements and node constraints. Therefore, our proposed protocol uses two well known key management approaches: the probabilistic key pre-distribution in the lower tier and public key cryptography in the upper tier. Overhead analysis and comparison with other protocols are performed to demonstrate the feasibility and the security efficiency of our solution for wireless sensor networks. Implementation and actual measurement of this protocol 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, 2007, pp. 24–34.

[3] 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.

[4] Crossbow Technology Inc., www.xbow.com

[5] S. Zhao, K. Tepe, I. Seskar, and D. Raychaudhuri, "Routing protocols for self-organizing hierarchical ad-hoc wireless networks," in IEEE Sarnoff 2003 Symposium.

[6] 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.

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

[8] 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.

[9] 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.

[10] 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.

[11] 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, 2003, pp. 42–51.

[12] 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.

[13] 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.

[14] M. Boujelben, H.Youssef, M.Abid, "An efficient scheme for key pre-distribution in wireless sensor networks", in proceeding of the 4th IEEE International conference on wireless and mobile computing: Networking and communication, France, 2008.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

TABLE I. A COMPAPISON SUMMARY BETWEEN KEY ESTABLISHMENT PROTOCOLS IMPLEMENTED FOR WSN

Evaluation metrics	**Protocols**			
	[10]	***[11]***	***[7]***	***Proposed protocol***
Network model	Homogeneous distributed network	Homogeneous distributed network	Heterogeneous hierarchical network	Heterogeneous hierarchical two-tiered network
Key establishment approach	Probabilistic key pre-distribution	Probabilistic key pre-distribution + Matrix based key generation	Probabilistic key pre-distribution	Probabilistic key pre-distribution + Matrix based key generation + Identity based cryptography
Security level	L	M	M	H
Scalability level	H	M	H	H
Computation overhead	0	d*(2 λ Mul + λ Add)	0	LS: 2 λ Mul+ λ Add HS: d*(2 λ Mul + λ Add) + M Enc/Dec + pairing function
Memory overhead	k * key size	k (λ+1) * key size	LS : k * key size HS : m * key size	LS : k(λ +1)* key size HS: m(λ+1)*key size + memory for the pairing function
Probability of communication overhead	$\frac{((\omega-k)!)^2}{\omega!(\omega-2k)!}$	$\frac{((\omega-k)!)^2}{\omega!(\omega-2k)!}$	$\frac{(\omega-k)!(\omega-m)!}{\omega!(\omega-m-k)!}$	$\frac{(\omega-k)!(\omega-m)!}{\omega!(\omega-m-k)!}$

L: Low, M: Meduim, H: High

LS: Low-end Sensor, HS: High-end Sensor

Mul: Multiplication, Add: Addition, Enc/Dec: Encryption/Decryption