SeGrid: A Secure Grid Infrastructure for Sensor Networks

Fengguang An[1], Xiuzhen Cheng[2,⋆], Qing Xia[3], Fang Liu[2], and Liran Ma[2]

[1] Institute of Computing Technology,
Chinese Academy of Sciences, Beijing 100080, P.R. China
[2] Department of Computer Science,
The George Washington University, Washington DC 20052, U.S.A
[3] Kellar Institute, George Mason University, Fairfax, VA 22030, U.S.A

Abstract. In this paper, we propose SeGrid, a secure grid infrastructure for large scale sensor networks. The basic idea relies on the availability of a low-cost public cryptsystem (e.g. Blom's key management scheme [4]) that can be used for shared key computation between the source and the destination, as long as the public shares are known to each other. In SeGrid, each sensor resides in a grid computed from its physical location. Within a grid, one or a few number of sensors, with one of them the grid head, are active at any instant of time and all other sensors fall asleep for energy conservation. We intend to compute a shared key for two grids instead of two nodes, such that the grid heads can securely communicate with each other. The public shares of a grid are stored at designated locations based on our public share management protocol such that the closer two grids, the shorter distance to obtain each other's public shares. We instantiate SeGrid based on Blom's key management scheme [4] to illustrate the computation of a grid key. To our best knowledge, this is the first work that simultaneously considers both key management and network lifetime extension, which explores along the dimension of network density.

Keywords: Blom's key management scheme, secure grid infrastructure.

1 Introduction

Security provisioning is a critical service for many sensor network applications. However, the severely-constrained resources (memory, processor, battery, etc.) within a sensor render many of the very popular security primitives inapplicable. Therefore, much research effort [1, 5, 9, 10, 12, 14, 16] has been placed on how to establish a shared key between two sensors such that their communications can be secured with low-cost symmetric encryption techniques.

Most existing schemes [9, 10, 12, 16] for distributed key agreement in sensor networks intend to design light weight (in computational complexity) algorithms

⋆ The research of Dr. Xiuzhen Cheng is supported by the NSF CAREER Award No. CNS-0347674.

X. Jia, J. Wu, and Y. He (Eds.): MSN 2005, LNCS 3794, pp. 422–432, 2005.
© Springer-Verlag Berlin Heidelberg 2005

to compute pairwise keys for communicating nodes. The induced key graph containing only edges incident at two sensors sharing common keys should be globally connected in order for the network to function well. Another constraint considered by these techniques is the memory budget allocated for pre-deployment key information storage. The tradeoff between memory cost and security has been well-studied in most of these works.

As understood by the research society, the utmost problem in a sensor network is its operation time elongation. Even though the above-mentioned works do take resource (especially energy) consumption into consideration, none of them explores the *density* dimension for further energy conservation. In this paper, we propose an idea of establishing a secure grid infrastructure (termed SeGrid) for sensor networks. We envision that all sensors within a grid are equivalent in routing and thus a secret key is needed between two grids (instead of two nodes) that demand communication. In this secure grid infrastructure, only one or a few number of sensors (for fault-tolerance) within a grid are active at any instant of time and all other sensors fall asleep to conserve energy. This design explores the fact that sensors are low-cost and are densely deployed in a typical network. When a new sensor becomes active, or an active sensor dies due to energy depletion, the shared grid keys should be recomputed. We instantiate this idea by applying Blom's key management scheme [4] to demonstrate the grid key computation. Note that putting redundant sensors to sleep for energy conservation is a popular method in all layers of the protocol [19, 20, 17] design for sensor networks. However, to the best of our knowledge, this work is the first to combine it with key management.

The basic idea of SeGrid is outlined in the following. We assume that there exists a public cryptosystem with low computation overhead (e.g. Blom's key management scheme [4]) such that each sensor can be preloaded with a crypto pair containing a public share and a private share before deployment. In SeGrid, sensors compute the grids they are residing in and choose to sleep or wake-up based on some schedule (e.g. the wake-up schedule proposed in [20]). Each grid has a *grid head*, an active node for message transmission. The grid head stores the public shares of all active nodes within its grid at designated locations and queries the nearest grid that stores the public shares of another grid based on our public share management protocol. After obtaining the public shares of the destination grid, source grid computes a key k_s that will be used to secure all transmissions between these two grids. The destination grid can follow the same procedure to compute the grid key k_s. The public share management protocol ensures that the closer two grids, the shorter the query distance to obtain each other's public shares. This protocol involves only simple algebraic (shift and addition) operations, thus has very low computation overhead. We finally instantiate SeGrid based on Blom's key management scheme [4] to demonstrate how the grid key can be computed based on the underlying public cryptosystem.

This paper is organized as follows. We elaborate the network model and the underlying assumptions in Section 2, then propose our secure grid infrastructure

for sensor networks (SeGrid) in Section 3. An example instantiation of SeGrid is outlined in Section 4. We conclude this paper with a discussion in Section 5.

2 Network Model and Assumptions

We are considering a large-scale sensor network deployed in outdoor environments. Each sensor is able to position itself through any of the techniques proposed in literature (eg. [6, 15, 18]). A virtual grid will be computed based on position information and each sensor resides in one grid. The id of a grid is denoted by (X, Y). At any instant of time, one or t number of sensors, where t is a small integer, are active and all other sensors fall asleep for energy conservation. A sleeping sensor wakes up periodically in order to replace a sensor with depleted energy. An active sensor is in full operation and all active sensors collaborate together to guarantee the functioning of the network. Sensors within neighboring grids can communicate directly. The wakeup/sleep schedule, the active/inactive status transition, and the underlying routing protocol for message dissemination, are out of the scope of this paper. We just simply assume that they are available for us to employ. Existing works related to these topics can be found in [3, 20].

We will explore a public cryptosystem that contains public and private crypto pairs. The public share can be disseminated as plain text while its corresponding private share must be kept secret. By exchanging their public shares, two nodes can compute a shared secret key based on their private shares and the exchanged public share. Example cryptosystems satisfying these conditions include the Diffie-Hellman key exchange protocol [11] and Blom's key management scheme [4]. In Section 4, we are going to instantiate a secure grid sensor network infrastructure based on Blom's method.

We assume each sensor is preloaded with a crypto pair before deployment. The operation of the sensor network is unattended after the initial bootstrap procedure for sensor localization and key management is done. Each grid may have more than one public shares, if it has more than one active sensors. An update message will be directed to all locations storing the public shares of the grid such that the public shares of newly introduced active (old inactive) sensors can be inserted (removed). A grid demanding the public shares of another grid can just query the nearest grid storing the corresponding information. We will propose a simple protocol for public share management in Subsection 3.2.

We envision that in a sensor network all nodes within a grid are equivalent. Therefore we only consider the secure communication between two grids. The computation of the shared key k_s between the two grids depends on the underlying public cryptosystem. We will show how to compute k_s based on Blom's key management scheme in Section 4. Note that intra-grid secure communication may be needed when more than one sensors are active simultaneously within a grid. The shared keys between these active nodes can be computed based on the underlying public cryptosystem too.

Note that even though $t > 1$ number of sensors may be active at any instant of time, we assume that only one sensor within a grid is in charge of transmission.

This sensor is the *grid head*. All active nodes other than the grid head listen to the messages directed from neighboring grids. This assumption is realistic since in a sensor network, a measurement from one sensor may not be attractive due to dynamics. Usually sensor readings in close neighborhood need to be combined for fault-tolerance [7] and an aggregated summary will be reported to the base stations [7, 8].

3 SeGrid: The Secure Grid Infrastructure

In this section, we propose the basic idea of our secure grid infrastructure for outdoor sensor networks. Note that this elaboration does not depend on any public cryptosystem. We will instantiate this idea in Section 4 based on Blom's key management scheme [4].

We will first describe a simple algorithm for each sensor to locally and independently compute the id of the grid it resides in. Then we give a novel protocol for each grid to determine where to store its public shares. In the last, we propose how to apply the secure grid infrastructure for protecting the unicast communications between two grids.

3.1 Grid Determination

In GAF [19], the size of a grid is determined based on node equivalence for routing. In other words, any node within a grid can communicate directly with any other node in any neighboring grid. This constraint specifies that the size of a grid, denoted by r, can be at most $\frac{R}{\sqrt{5}}$, where R is the nominal transmission range. In our study, we adopt this idea since we also intend to turn off most of the sensors within a grid for energy conservation in order to extend network lifetime. GAF specifies the length of the grid edge but does not specify how to determine the grid a node resides. In the following, we propose a very simple algorithm to allow each node independently and locally determine its grid.

Let (x, y) be the location of any sensor residing at grid (X, Y). Then we have $X = \lfloor x \div 2^{\lfloor \log_2 r \rfloor} \rfloor$, $Y = \lfloor y \div 2^{\lfloor \log_2 r \rfloor} \rfloor$. Note that X and Y can be computed through shift operations only, as long as $2^{\lfloor \log_2 r \rfloor}$ is computed off-line and uploaded into each sensor before deployment. This is a reasonable assumption since r depends only on the nominal transmission range R, which can be made available before deployment. Therefore we can simply shift the binary representations of x and y to the right for k positions, where $k = \lfloor \log_2 r \rfloor$, to obtain X and Y.

Remark: The procedure of computing the grid id of a sensor based on its physical location does not require precise location information. We may treat k as an error bound. In other words, as long as the position error in both x and y directions are upper-bounded by 2^k, we can still determine the grid id. In this case, the constraint that two nodes within neighboring grids should communicate directly may be violated.

3.2 Public Share Management

In this subsection, we propose a simple protocol for storing and querying the public shares of a grid. We need to answer two questions. First, for any grid (X_0, Y_0), where shall we store its public shares? Second, if grid (X_1, Y_1) would like to securely communicate with (X_0, Y_0), where to find out the latter's public shares? Our protocol is based on the following assumption: the closer two grids, the higher the probability they may communicate. Therefore, the public shares of a grid will be stored at designated locations such that the closer to the grid, the shorter the query distance involved in public share acquisition. In our protocol, the density of the grids storing the public shares of a grid drops logarithmically as the distance to the grid increases. Fig. 1 gives a simple example to illustrate the storage locations of the public shares for the grid $(4, 4)$.

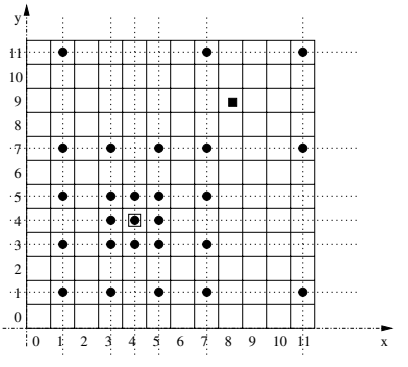

Fig. 1. The public shares of the grid $(4, 4)$ are stored at the grids denoted as "." in the figure. If the grid $(8, 9)$ needs the public shares of $(4, 4)$, it can query either $(7, 7)$ or $(7, 11)$ since they are closer.

The answer to the first question is very simple. The public shares of the grid (X_0, Y_0) will be stored at (x, y) if $x = X_0 \pm (2^i - 1)$ or $x = X_0 \pm (2^{i+1} - 1)$, and $y = Y_0 \pm (2^i - 1)$ or $y = Y_0 \pm (2^{i+1} - 1)$, where $i = 0, 1, 2, \cdots$. To identify the nearest grid that stores the public shares of (X_0, Y_0), the grid (X_1, Y_1) will compute

$$\Delta X_L = 2^{\lfloor \log_2 |X_1 - X_0| \rfloor} - 1\ ,\ \ \Delta Y_L = 2^{\lfloor \log_2 |Y_1 - Y_0| \rfloor} - 1,$$
$$\Delta X_H = 2^{\lceil \log_2 |X_1 - X_0| \rceil} - 1\ ,\ \ \Delta Y_H = 2^{\lceil \log_2 |Y_1 - Y_0| \rceil} - 1,$$

to figure out the following four grids that contain the public shares of (X_0, Y_0): $(X_1 - \Delta X_L, Y_1 - \Delta Y_L)$, $(X_1 - \Delta X_L, Y_1 + \Delta Y_H)$, $(X_1 + \Delta X_H, Y_1 - \Delta Y_L)$, $(X_1 + \Delta X_H, Y_1 + \Delta Y_H)$, and then choose the nearest one to query.

Note that if Manhattan distance instead of Euclidean distance is used as a routing metric for public share queries and updates, the computation overhead is very low since only simple addition and subtraction operations are involved. For example in Fig. 1, $\Delta X_L = 1$, $\Delta X_H = 3$, $\Delta Y_L = 2$, and $\Delta Y_H = 2$. The

Manhattan distance from $(8, 9)$ to $(7, 11)$ is $\Delta X_L + \Delta Y_H = 3$. Similarly we can compute the Manhattan distance to the other three points. We choose either $(7, 7)$ or $(7, 11)$ to query since they are the closest among the four grids that are close to $(8, 9)$.

Remarks:

- The computation of the storage locations for a grid contains only shift and addition operations. However, the identification of the nearest grid for public share query involves the complicated log functions. Nevertheless, this can be done easily through a lookup table.
- This protocol guarantees that closer grids obtain the public shares within shorter distance. Therefore, the farther away a grid, the higher the communication overhead for public share query. In reality, closer grids intend to communicate securely with higher probability.
- The update of the public shares for a grid always take the same number of messages, as long as the routing protocol remains unchanged.

3.3 Secure Grid Communication

Now we are ready to propose our secure grid communication framework. We assume there exists a routing protocol, either geography-based (e.g. [13]) or topology-based (e.g. [3]), such that we can employ directly.

Let (X_A, Y_A) and (X_B, Y_B) be the two grids that require a secure communication. The following procedure will be conducted at both ends:

1. Query the nearest grid that contains the public shares of the other party based on the procedure proposed in Subsection 3.2 to obtain the public shares.
2. Compute a secret key K_s shared by these two grids and secure the future communication with this key.

Note that Step 2 depends on the underlying public cryptosystems. In the following section, we will show how to apply Blom's key management scheme [4] to compute the shared key between two grids.

4 A Simple Realization

In this section, we implement a secure grid infrastructure for sensor networks based on Blom's key management scheme [4]. For completeness, we give a brief overview on Blom's scheme first. Then we describe how to compute a grid key based on Blom's scheme.

4.1 Preliminary: Blom's Key Management Scheme

Blom's λ-secure key management scheme [4] has been well-tailored for light-weight sensor networks by [9]. In the following, we will give an overview on Blom's scheme based on [9].

Let G be a $(\lambda+1)\times M$ matrix over a finite field $GF(q)$, where q is a large prime. The connotation of M will become clear latter. G is public, with each column called a public share. Let D be any random $(\lambda+1)\times(\lambda+1)$ symmetric matrix. D must be kept private, which is known to the network service provider only. The transpose of $D\cdot G$ is denoted by A. That is, $A=(D\cdot G)^T$. A is private too, with each row called a private share. Since D is symmetric, $A\cdot G$ is symmetric too. If we let $K=(k_{ij})=A\cdot G$, we have $k_{ij}=k_{ji}$, where k_{ij} is the element at the ith row and the jth column of matrix K, $i,j=1,2,\cdots,M$.

The basic idea of Blom's scheme is to use k_{ij} as the secret key shared by node i and node j. D and G jointly define a *key space* (D,G). Any public share in G has a unique private share in A, which form a so-called *crypto pair*. For example, the ith column of G, and the ith row of A form a crypto pair and the unique private share of the ith column of G, a public share, is the ith row of A. Two sensors whose crypto pairs are obtained from the same key space can compute a shared key after exchanging their public shares. From this analysis, it is clear that M is the number of sensors that can compute their pairwise keys based on the same key space.

In summary, Blom's scheme states the following protocol for nodes i and j to compute k_{ij} and k_{ji}, based on the same key space:

- Each node stores a unique crypto pair. Without loss of generality, we assume node i gets the ith column of G and the ith row of A, denoted by g_{ki} and a_{ik}, where $k=1,2,\cdots,\lambda+1$, respectively. Similarly, node j gets the jth column of G and the jth row of A, denoted by g_{kj} and a_{jk}, where $k=1,2,\cdots,\lambda+1$, respectively.
- Node i and node j exchange their stored public shairs drawn from their crypto pairs as plain texts.
- Node i computes k_{ij} as follows: $k_{ij}=\sum_{k=1}^{\lambda+1}a_{ik}\cdot g_{kj}$; Similarly, node j computes k_{ji} by $k_{ji}=\sum_{k=1}^{\lambda+1}a_{jk}\cdot g_{ki}$.

Blom's key management scheme ensures the so-called λ-secure property, which means the network should be perfectly secure as long as no more than λ nodes are compromised. This requires that any $\lambda+1$ columns of G must be linearly independent. An interesting method of computing G is proposed by Du *et. al* in [9]. This idea is sketched as following. Let *len* be the number of bits in the symmetric key to be computed. Choose q as the smallest prime that is larger than 2^{len}. Let s be a primitive element of $GF(q)$ and $M<q$. Then

$$G=\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ s & s^2 & s^3 & \cdots & s^M \\ s^2 & (s^2)^2 & (s^3)^2 & \cdots & (s^M)^2 \\ & & \vdots & & \\ s^\lambda & (s^2)^\lambda & (s^3)^\lambda & \cdots & (s^M)^\lambda \end{bmatrix}$$

Note that G is a Vandermonde matrix. Each column of G represents the public share of some sensor node storing that column. In Blom's key management

scheme, public shares need to be exchanged between sensors that require secure peer-to-peer communication. Based on the structure of G, we observe that only the second element of each column, the *seed* of the column, needs to be stored and exchanged. Thus both storage and communication overheads can be greatly decreased.

4.2 Shared Key Computation

Assume a large key space (D, G) following Blom's key management scheme has been computed off-line. Before deployment, each sensor receives a crypto pair from the key space. Note that the crypto pairs to different sensors may not be unique, as the key shared by two grids will be computed based on multiple public shares. But we require that all active sensors within one grid have different crypto pairs.

Let t_A (t_B) be the number of active sensors in a grid (X_A, Y_A) ((X_B, Y_B)). Following the public share management protocol proposed in Subsection 3.2, all these t_A (t_B) public shares will be stored at designated grids, and are available to other grids. If (X_A, Y_A) and (X_B, Y_B) need secure communication, they acquire the public shares of the other party first in order to compute a shared key.

In Blom's key management scheme, two sensors can compute a shared key as long as they know each other's public share. We can derive a shared key k_s between two grids from the keys shared by all pairs of sensors within the two gride, as shown in Fig. 2.

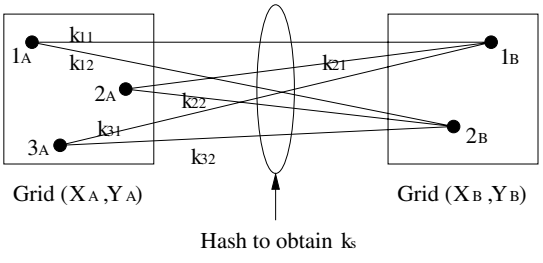

Fig. 2. The grid $(X_A, Y_A)((X_B, Y_B))$ contains three (two) active sensors with node 1_A (1_B) as the grid head. After obtaining the public shares of (X_B, Y_B), each node i_A in grid (X_A, Y_A) first computes k_{i1} and k_{i2}, the shared keys with the two nodes in grid (X_B, Y_B), then transmits securely to node 1_A the value $k_{i_A} = Hash(k_{i1}, k_{i2})$. Finally, node 1_A computes k_s as $k_s = Hash(k_{1_A}, k_{2_A}, k_{3_A})$. Similarly, node 1_B computes k_s based on the public shares of (X_A, Y_A).

Let's use grid (X_A, Y_A) as an example to demonstrate the procedure of computing a shared key with the grid (X_B, Y_B). After obtaining the public shares of grid (X_B, Y_B), each node i in (X_A, Y_A) computes a shared key with each node j in grid (X_B, Y_B). These pairwise keys are denoted by k_{ij}, where $i = 1, 2, \cdots, t_A$ and $j = 1, 2, \cdots, t_B$. Then i computes $k_i = Hash(k_{i1}, \cdots, k_{it_B})$. This value will

be securely transmitted to the grid head h of (X_A, Y_A). After receiving all k_t's, where $t \neq i$, h derives the grid key k_s by computing $Hash(k_1, k_2, \cdots, k_{t_A})$.

Remarks:

- The private shares of each sensor must be kept secret. Therefore if one node within a grid is compromised, grid keys can still be computed. Further, the grid key remains secure if an active node other than the grid head is compromised.
- When the active nodes within a grid changes, the public shares of the grid can be modified easily since only the public share of the node with role change needs to be updated.
- The hash function exploited must be linear, and must be able to take any number of inputs. The XOR function is a simple example.
- Two nodes within a grid can compute a shared key based on Blom's method easily after exchanging their public shares as plain texts. This key can be used to secure the intra-grid communication.
- The security of the grid key computation protocol based on Blom's key management scheme [4] is determined by the λ-secure property of the key space (D, G). Therefore if the crypto pairs of more than λ sensors are exposed to the adversary, the security of the whole network is compromised. This is the major drawback of applying Blom's key management scheme for grid key computation since the memory budget within a sensor for security information storage is limited.
- The space consumed for storing the crypto pairs within a sensor is exclusively determined by λ. The larger the λ, the higher the security level, and the larger the storage space.
- The computation overhead of a grid key is determined by λ too. Each shared key computation between two active nodes takes $\lambda + 1$ number of modular multiplications.

5 Conclusion and Future Research

In this paper, we have proposed SeGrid, a secure grid infrastructure based on public cryptosystems for large scale sensor networks. We have instantiated SeGrid based on Blom's key management scheme to demonstrate how to compute a grid key shared by two grids. To our knowledge, SeGrid is the first work that targets key management in a level that is above the physical sensors. This is a more practical consideration since sensors may stay in sleep mode most of the time for network lifetime extension.

As a future work we will explore the applicability of ID-Based Cryptosystems [2] in SeGrid. In an ID-based encryption system, the public key can be any string (e.g. an email address), and the private key needs to be computed from the public key and other system parameters. The idea of using the grid ID as a public key in SeGrid is very attractive since public key management can be totally avoided.

References

1. F. An, X. Cheng, M. Rivera, J. Li, and Z. Cheng, PKM: A Pairwise Key Management Scheme for Wireless Sensor Networks, to appear in *2005 International Conference on Computer Networks and Mobile Computing (ICCNMC'05)*, Zhangjiajie, Hunan, China, August 2-4, 2005.
2. D. Boneh and M. Franklin, Identity-Based Encryption from the Weil Pairing, *Proceedings of CRYPTO'01*, Spinger-Verlag, 2001.
3. A. Boukerche, X. Cheng, and J. Linus, Energy-Aware Data-Centric Routing in Microsensor Networks, *MSWiM 2003*, pp. 42-49, 2003.
4. R. Blom, An optimal class of symmetric key generation systems, in *Proc. of the EUROCRYPT 84 workshop on Advances in cryptology: theory and application of cryptographic techniques*. Springer-Verlag New York, Inc., pp. 335-338, 1985.
5. H. Chan, A. Perrig, and D. Song, Random Key Predistribution Schemes for Sensor Networks, *IEEE SP 2003*.
6. X. Cheng, A. Thaeler, G. Xue, and D. Chen, TPS: a time-based positioning scheme for outdoor sensor networks, *INFOCOM 2004*, Vol. 4, pp.2685-2696, HongKong, March 7-11, 2004.
7. M. Ding, D. Chen, K. Xing, and X. Cheng, Localized Fault-Tolerant Event Boundary Detection in Sensor Networks , *IEEE INFOCOM 2005*, 13-17 March 2005.
8. M. Ding, D. Chen, A. Thaeler, and X. Cheng, Fault-Tolerant Target Detection in Sensor Net works, *IEEE WCNC 2005*, Vol. 4, pp. 2362-2368, 13-17 March 2005.
9. W. Du, J. Deng, Y. S. Han, and P. K. Varshney, A pairwise key pre-distribution scheme for wireless sensor networks, in *CCS '03: Proceedings of the 10th ACM conference on Computer and communications security*. ACM Press, pp. 42–51, 2003.
10. W. Du, J. Deng, Y.S. Han, S. Chen, and P.K. Varshney, A Key Management Scheme for Wireless Sensor Networks Using Deployment Knowledge, *IEEE INFOCOM 2004*.
11. W. Diffie and M.E. Hellman, New Directions in Cryptography, *IEEE Transactions on Information Theory*, Vol. 22, No. 6, pp. 544-654, 1976.
12. L. Eschenauer and V.D. Gligor, A Key-Management Scheme for Distributed Sensor Networks, *CCS'02*, pp.41-47, November 18-22, 2002, Washington DC, USA.
13. B. Karp and H.T. Kung, GPSR: Greedy Perimeter Stateless Routing for Wireless Networks, *ACM MOBICOM 2000*, pp. 243-254.
14. D. Liu and P. Ning, Establishing Pairwise Keys in Distributed Sensor Networks, *ACM CCS'03*, pp. 52-60, 2003.
15. F. Liu, X. Cheng, D. Hua, and D. Chen, TPSS: A Time-based Positioning Scheme for Sensor Networks with Short Range Beacons, to appear in *2005 International Conference on Computer Networks and Mobile Computing (ICCNMC'05)*, Zhangjiajie, China, August 2-4, 2005.
16. F. Liu, L. Ma, X. Cheng, D. Hua and J. Li, S-KMS: A Self-configured Key Management Scheme for Sensor Networks, submitted to *IEEE INFOCOM 2006*.
17. L. Ma, Q. Zhang, and X. Cheng, A Power Controlled Interference Aware Routing Protocol for Dense Multi-Hop Wireless Networks, *submitted*.

18. A. Thaeler, M. Ding, and X. Cheng, iTPS: An Improved Location Discovery Scheme for Sensor Networks with Long Range Beacons, Special Issue on *Theoretical and Algorithmic Aspects of Sensor, Ad Hoc Wireless, and Peer-to-Peer Networks* of *Journal of Parallel and Distributed Computing*, Vol. 65, No. 2, pp.98-106, Feburary 2005.
19. Y. Xu, J. Heidemann, and D. Estrin, Geography-Informed Energy Conservation for Ad Hoc Routing, *Proceedings of the 7th annual international conference on Mobile computing and networking*, pp. 70-84, Rome, Italy, 2001.
20. W. Ye, J. Heidemann and D. Estrin, An energy-efficient MAC protocol for wireless sensor networks, *Proceedings of IEEE INFOCOM*, pp. 1567-1576, 2002.