A Low-Energy Key Management Protocol for Wireless Sensor Networks

Gaurav Jolly, Mustafa C. Kuşçu, Pallavi Kokate, and Mohamed Younis

Computer Science and Electrical Engineering
University of Maryland, Baltimore County
Baltimore, Maryland 21250
{jolly1, kuscu1, pallavi2, younis}@umbc.edu

[*]**Abstract.** *Wireless sensor networks have a wide spectrum of civil and military applications that call for security, e.g., target surveillance in hostile environments. Typical sensors possess limited computation, energy, and memory resources; therefore the use of vastly resource-consuming security mechanisms is not possible. In this paper, we propose a cryptographic key management protocol, which is based on the IBSK scheme, but only two symmetric keys are required to be pre-deployed at each sensor. The protocol supports the eviction of the compromised nodes. Simulation shows that the energy consumption overhead introduced by the key management is remarkably low thanks to the multi-tier network architecture in which only sensor-to-gateway secure sessions are allowed, and reports order-of-magnitude improvement in energy saving as compared to the original IBSK scheme, and Kerberos-like schemes.*

1. Introduction

Many compelling applications like distributed information gathering and distributed micro sensing in radiology, military, and manufacturing drive the research in sensor networks. Typically, sensor networks comprise of a large set of distributed low power sensors scattered over the area to be monitored. The sensors have the ability to gather data, and process and forward it to a central node for further processing. A major challenge for the sensor networks is the limitations on the sensor hardware.

Contemporary wireless sensors have limited battery, computation, and memory capacity. Such resource-constrained environment has motivated extensive research that addresses energy-aware hardware and software design issues [1][2]. Much effort has been on the energy-efficient communication protocols [3][4][5]. The comparative progress in making these networks secure has been insignificant. This is despite the fact that in certain applications of sensor networks, like military applications, security becomes important.

The energy-constrained nature of the sensor networks makes the problem of incorporating security very challenging. Many well-known security mechanisms introduce significant computational/memory-wise overhead. The design of the security protocols for sensor networks should be geared towards conservation of the sensor resources. The level of security versus the consumption of energy, computation, and memory resources constitute a major design trade-off.

In this paper, we focus on the design of a very low energy key management scheme for a sensor network. Our protocol, which is an extension of the Identity-Based Symmetric Keying (IBSK) scheme [10], introduces flexibility to IBSK by supporting the addition of sensors and the revocation of the network nodes (their keys), as well as key renewals. Our scheme inherits the advantages of the pre-deployed keying nature of the IBSK. In order to further reduce the energy consumption by the sensors, we rule out the direct end-to-end communication among sensors feature, while preserving the hop-by-hop routing functionality. We also provide simulation results for the energy consumption (Section 4).

The organization of the rest of the paper is as follows. Section 2 describes the sensor network architecture that we study. In Section 3, we present a hierarchical key management protocol that consists of sub-protocols for the key distribution and the initialization steps, as well as for the addition of sensors and the revocation of the network nodes, and a key renewal mechanism for our sensor network. We list the assumptions and design trade-offs in Section 3. Section 4 describes the simulation environment and results. Section 5 summarizes the related work on sensor network security and key management. Finally Section 6 concludes the paper and discusses open problems.

[*] Mustafa C. Kuşçu is the contact author. He can be reached by phone at (410) 455-3968, by FAX at (410) 455-3969 and by e-mail as kuscu1@umbc.edu. The authors would like to thank Dr. Alan Sherman and William Byrd for their valuable comments.

Proceedings of the Eighth IEEE International Symposium on Computers and Communication (ISCC'03)
1530-1346/03 $17.00 © 2003 IEEE

2. Sensor network architecture

We adopt the sensor network model proposed by Younis, et al. [3][6]. In this model, a sensor network consists of a large number of sensors distributed over an area of interest. There is a command node in charge of the network's mission. The model also introduces super-nodes, called gateways, in addition to the sensor nodes. The gateways have considerably high energy resources compared to the sensors, and are equipped with high performance processors and more memory. As shown in Figure 1, the gateways partition the sensors into distinct clusters, using a clustering algorithm, e.g. [7]. Each cluster is composed of a gateway and a set of sensor nodes (distinct from other sets), which gather information and transmit to the gateway of their cluster. The gateway fuses the data from the different sensors, performs mission-related data processing, and sends it to the command node via long-haul transmission.

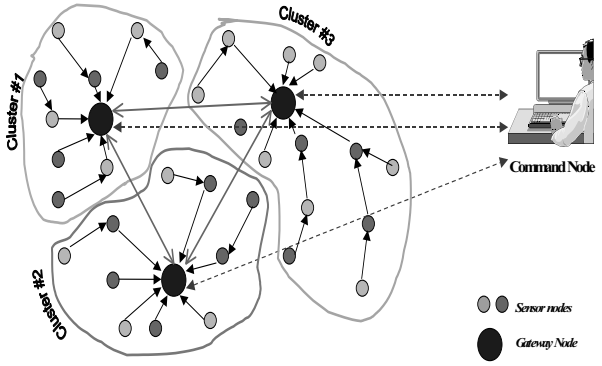

Figure 1. Multi-gateway, clustered sensor network.

On the other hand, typical sensors are extremely constrained in resources. For example, a MICA sensor [9] is quite restrictive: TinyOS operating system, 4K RAM, 128K flash memory, and Atmel ATmega processor (8-bit, 4MHz), and runs with 2 AA batteries.

In our model [3][6], the energy-aware operation of the sensors is mainly achieved by keeping the quiescent sensors in the *sleep* mode by using a Time Division Multiple Access (TDMA) based Media Access Control, wherein the gateway assigns distinct activity slots to the sensors. Other than the sleep mode, the sensor can be either *active* or *idle*. In the active mode, the sensor is transmitting, receiving, or sensing. In the idle mode the sensor is idle but its circuitry is on. The sensors transmit in the slots assigned to them and thereafter, in order to conserve energy, make a transition to sleep mode. The computation part of the sensor is always on except in the sleep mode. In the sleep mode the circuitry is switched off and hence the sensor can save energy and extend its lifetime.

In the next section, we focus on the key management aspects for a sensor network used in military reconnaissance setup.

3. Key management

A key management procedure is an essential constituent of network security. Symmetric key systems require the keys to be kept out of reach of the adversary. Moreover, sensor networks have energy-wise and computational constraints; therefore it is necessary to maintain a balanced security level with respect to those constraints. In this section we propose a key management scheme for sensor networks, whose objective being the minimization of the sensor's computational, communications-wise and storage overhead due to the key management operations.

In military reconnaissance scenarios, the sensors and the gateways are likely to be deployed in the enemy territory. The sensors in each cluster will report the sensed enemy movements to the gateway that in turn will process that data and forward it to the command node. The command node is deployed in the friendly environment. Therefore, we can assume that the command nodes are secure while the gateways and sensors can be compromised.

3.1 Assumptions

In a data gathering setup like ours, the end-to-end communications between sensors is not common. The semantic of the sensor-to-sensor communication can be viable in system models that require the sensors to perform data fusion and setup the routes[1]. In our model, this is not the case, instead, all traffic within a cluster flows over the gateway of the cluster. Therefore, the high overhead of establishing secure sessions between individual sensors can be avoided. Moreover, we assume that each sensor uses direct communications with the gateway during key management operations.

We assume that the functionality of intrusion detection is available (e.g., tamper detection, network intrusion detection) to the command node, although we are not aware of any intrusion detection for sensor networks and although we do not specify how one is going to work.

The sensors and the gateways are randomly distributed and are not aware of the topology prior to the deployment.

Some assumptions on the network nodes are as follows:

i. Sensors. We are not making any trust assumptions on sensors or any assumptions on the capabilities of the adversary. Each sensor is capable of determining its location by using GPS during bootstrapping. Sensors remain stationary during the operation of the network.

ii. Gateways. Each gateway can directly communicate with every other gateway in the network. Moreover, we

[1] During bootstrapping, sensors may need to communicate with each other. We are not addressing the security of network bootstrapping.

Table 1: Notation used in the key management protocols.

Notation	Description
C	command node
G_i	gateway i
S_i	sensor node i
G	set of all gateways in the network
S	set of all sensors
id_i	identifier for node i
nonce	random nonce value
sdata	Sensor location and energy level data
$K_{A,B}$	secret key shared between A and B (A and B each can be S_i, G_i, or C)
E(K, ...)	symmetric encryption function using key K
$\|\|$	concatenation operator
G_h	head gateway (used for revocation)

assume that broadcast communication among gateways is available. We also assume that there is secure group communications among the gateways. The clustering algorithm, e.g. [7], can be easily extended to bring in the setting up of the secure communications. The gateways can establish the group key using a group key agreement protocol. Carman et al discuss those protocols in [10].

iii. Command node. The command node is assumed to be secure and is trusted by all of the nodes in the sensor network. The intrusion detection mechanism will operate integral to the command node, and the eviction of a compromised node will be triggered by the intrusion detection mechanism.

3.2 Proposed approach

Our key management protocol is a symmetric-key mechanism, and consists of the sub-protocols that define how keys are distributed, added, revoked, and renewed during the lifetime of the sensor network. The approach does not call for any sensor to generate keys, or to perform any extensive computation associated with key management.

Distribution of the keys. We utilize a secret-key mechanism, and each sensor stores only two keys. One key is shared with a gateway, and the other is shared with the command node. Since the sensors are not trusted and are memory-constrained, storing a small number of keys is advantageous for the security of the network, as well as it saves the memory. The gateways have rich memory resources and can store large number of keys; however, they cannot be completely trusted. Assigning all the keys to the gateway will compromise the entire network, even in the case of compromise of a single gateway. The command nodes are assumed to be secure, and have sufficient memory. Therefore the command node can store all of the secret keys in the network. The sensor keys are programmed into the memory of the sensors before they are deployed. They can be stored in the flash RAM, and be erased when necessary.

Table 1 displays the notation used in the protocol descriptions. The number of keys stored by the command node is equal to $|G|+|S|$, where $|G|$ is the number of gateways and $|S|$ is the number of sensors. Each gateway stores the keys it shares with the sensors in its cluster and the key it shares with the command node. It also shares one key with exactly one other gateway in the network, along with the group key shared with all the gateways. The keys are pre-deployed, therefore there is no key transmission/reception overhead (consumes energy) at the sensor side during the bootstrapping.

Initialization phase. At the time of deployment, each gateway is randomly assigned $|S|/|G|$ keys. Then each gateway forms clusters using a cluster formation algorithm, and thereafter acquires the keys of the sensors in its cluster from the other gateways. After the key exchange at the gateway level, each gateway keeps the keys of the sensors that in its cluster, others keys are erased. This is essential since if a gateway is captured then the keys of only its cluster become available to the adversary. The protocol for the initialization phase is as follows:

$$S_i \rightarrow \text{broadcast} \quad \mathrm{id}_{G_j} \,||\, \mathrm{id}_{S_i} \,||\, \mathrm{E}(K_{S_i,G_j}, \text{nonce} \,||\, \text{sdata}) \qquad \text{(i1)}$$

Each sensor is preloaded with the identifier of the gateway (id_{G_j}) that contains its shared key. The sensor includes this identifier in the "hello" message it broadcasts after its deployment (i1).

$$\text{(Clustering process)} \qquad \text{(i2)}$$

$$G_i \rightarrow G \quad \mathrm{id}_{G_i} \,||\, \mathrm{E}(K_G, \text{nonce} \,||\, \{\mathrm{id}\}_i) \qquad \text{(i3)}$$

After the cluster formation, each G_i identifies the set of sensors $\{\mathrm{id}\}_i$ that reside in its cluster and broadcasts to the other gateways. (The broadcast helps reduce the volume of inter-gateway traffic.)

$$G_i \leftarrow G_j \quad \mathrm{E}(K_{G_i,G_j}, \text{nonce} \,||\, \{(K_{S_k,G_j}, \mathrm{id}_{S_k})\}_i) \qquad \text{(i4)}$$

Each G_j replies to G_i with its set of keys $\{(K_{S_k,G_j}, \mathrm{id}_{S_k})\}_i$, where $\{\mathrm{id}\}_i$ is a subset of $\{\mathrm{id}\}_j$. Then, each sensor S_l in G_i's cluster receives a message from G_i that assigns G_i as its gateway:

$$S_l \leftarrow G_i \quad \mathrm{id}_{G_i} \,||\, \mathrm{E}(K_{S_l,G_j}, \mathrm{nonce} \,||\, \mathrm{id}_{G_i} \,||\, \mathrm{msg}) \qquad \text{(i5)}$$

Addition of sensors. The new sensors are arbitrarily deployed; they cannot be pre-assigned to a cluster. However, they are preloaded with two keys as other sensors.

The command node transmits the list of (identifier, key) pairs to a randomly selected gateway G_h (not to the whole gateway group, to reduce the risk of compromise), which becomes the gateway that shares the keys of the new sensors:

$$C \rightarrow G_i \qquad \mathrm{E}(K_{G_i,C}, \mathrm{nonce} \,||\, \{(K_{S_k,G_i}, \mathrm{id}_{S_k})\}_i) \qquad \text{(a1)}$$

Besides, each added sensor node broadcasts a 'hello' message as in the initialization step i1. The clustering mechanism adjusts itself (possibly by reconfiguring the clusters). Each gateway broadcasts the sensors in its range to the gateways in *G,* requesting the keys for those sensors.

$$S_l \rightarrow \text{broadcast} \qquad \mathrm{id}_{G_h} \,||\, \mathrm{id}_{S_l} \,||\, \mathrm{E}(K_{S_l,G_j}, \mathrm{nonce} \,||\, \mathrm{sdata}) \qquad \text{(a2)}$$

(Clustering process) (a3)

$$G_i \rightarrow G \qquad \mathrm{id}_{G_i} \,||\, \mathrm{E}(K_G, \mathrm{nonce} \,||\, \{\mathrm{id}\}_i) \qquad \text{(a4)}$$

G_h responds to those requests. Then each new sensor S_l is assigned its gateway G_i.

$$G_i \leftarrow G_h \qquad \mathrm{E}(K_{G_i,G_h}, \mathrm{nonce} \,||\, \{(K_{S_k,G_h}, \mathrm{id}_{S_k})\}_i) \qquad \text{(a5)}$$

$$S_l \leftarrow G_i \qquad \mathrm{id}_{G_i} \,||\, \mathrm{E}(K_{S_l,G_h}, \mathrm{nonce} \,||\, \mathrm{id}_{G_i} \,||\, \mathrm{msg}) \qquad \text{(a6)}$$

Revocations. Key revocations (and node evictions) are performed after detecting the compromised nodes. The (hypothetical) intrusion detection mechanism informs the command node of the compromised nodes. If a group of sensors are compromised, they can be trivially evicted from the command node's sensor list by the command node, as well as from their cluster by the gateway. (Note that the gateway reconfigures its intra-cluster hop-by-hop routing, ignoring the evicted sensors.)

In the case of the revocation of a gateway (G_j) key, the command node evicts G_j from G, and chooses an uncompromised gateway G_h as a *head* gateway. To G_h, it sends the identifiers each sensor and its new gateway (G_i), and the new key to be shared with G_i. Also, the new gateway-sensor secret keys are sent to G_i through group broadcast. After this, a re-clustering step takes place.

$$C \rightarrow G_h \qquad \mathrm{E}\Big(K_{G_h,C}, \mathrm{nonce} \,||\, \big\{(\mathrm{id}_{S_k} \,||\, \mathrm{id}_{G_i} \,||\, \mathrm{E}(K_{G_i,C}, \mathrm{nonce}' \,||\, \mathrm{id}_{S_k} \,||\, K_{S_k,G_i}) \,||\, \mathrm{E}(K_{S_k,C}, \mathrm{nonce}'' \,||\, \mathrm{id}_{G_i} \,||\, K_{S_k,G_i}))\big\}_j\Big) \qquad \text{(r1)}$$

(Clustering process) (r2)

On successfully decrypting the message the sensor receives its new key (the last component in the r1 step) shared with G_i. Thus, it accepts G_i as its new gateway, and ignores the further messages from G_j.

$$G_i \rightarrow S_k \qquad \mathrm{E}(K_{S_k,C}, \mathrm{nonce} \,||\, \mathrm{id}_{G_i} \,||\, K_{S_k,G_i}) \qquad \text{(r3)}$$

Key renewals. Using the same encryption key for extended periods may incur a cryptanalytic risk. A remedy can be to ignore this threat, which can be suitable for short-living networks, e.g., where the sensor battery drains quickly [8]. For other networks, it will be necessary to renew the encryption keys occasionally [11]. In order to accomplish the renewal of the sensor keys, the command node generates the new keys, and pushes the keys to the gateways, as in the case of the revocation.

The time interval between subsequent renewals may depend on the data traffic volume, the strength of underlying cryptographic primitives, and the extra processing load incurred at the gateways.

4. Simulation

We simulate the key management energy consumption, by using a Visual C++ based sensor network simulator [3][6].

We model the sensor energy dissipated during the key management. The computational component is a small fraction as compared to the communication therefore it is not considered. From the previous section, we can see that the sensors transmit in the steps i1, a2, and receive in i5, a6, r3. The clustering process is not considered. We are interested in the initialization part, as each sensor performs i1 and i5.

Our metrics are the energy consumption per node, the average energy consumption within overall network, and the overhead of security in terms of the energy consumption.

The average energy consumption for a uniformly 10-cluster network is less than 50μJ, as can be seen in Figure 1. In Figure 2, we can see that fewer than 30% of the sensors consumed more than 50μJ. The density of gateways

can be increased to further reduce the average energy consumption.

5. Related work

During the past few years, the architecture and design of sensor networks and hardware have progressed significantly [4][5][12][13].

Recently, security has become a topic of interest in sensor networks research. The detailed survey by Carman et al [10] analyzes the existing key management technologies on the sensor networks. They observed that *pre-deployed keying*, which requires deployment of keys to the sensors prior to their use, is very energy efficient; however, it poses inflexibility to configuration changes and requires storage of high number of keys. One way to do pre-deployed keying is to use a *pre-deployed group keying* approach. However, compromise of a single node results in the compromise of the entire network, because all nodes share the same key. In the *node-specific pre-deployment* method, a unique key is stored for each node pair. For large networks, the number of stored keys at each node can be too high.

A protocol proposed in [10] is the Identity-Based Symmetric Keying (IBSK), which is a node-specific pre-deployment technique, is very similar to our approach as well. This protocol requires the distinct shared keys to be pre-deployed in the sensor nodes. Therefore, the scalability in the number of stored keys is a major concern with IBSK. Moreover, post-deployment key management operations are sophisticated and energy-inefficient. For instance, consider the key renewal operation on a single sensor. Since the sensor shares keys with a high number of remote nodes (possibly as large as the size of a cluster), each sensor has to be updated with a new key; and the original sensor has to be updated with a large number of keys. The updates require excessive communication, which has adverse effects on the sensor energy level and lifetime. Quantitatively speaking, renewing a single node in an n-node cluster will require an $(n - 1)$-key-long transfer to that node, and $(n - 1)$ single key transfers to the other nodes in the cluster. Total key transfers add up to $2(n - 1)$. This means, each sensor consumes approximately (average) 8.6mJ per sensor, which is quite high. This is mainly because of the fact that IBSK assumes that node-to-node communication is a necessary feature. This assumption is valid, because sensor networks can stage ad-hoc behavior. However, for many data gathering applications, a clustered architecture and a sensor-to-gateway traffic pattern are sufficient. In fact, the data gathering applications that utilize sensor networks eventually have an inherent centralized architecture (flow of data towards a central node). This observation was already made by Perrig et al [8] to reduce the key management overhead. By restricting IBSK, its energy consumption would be significantly reduced, and the keys at the sensors will be manageable, without sacrificing network functionality.

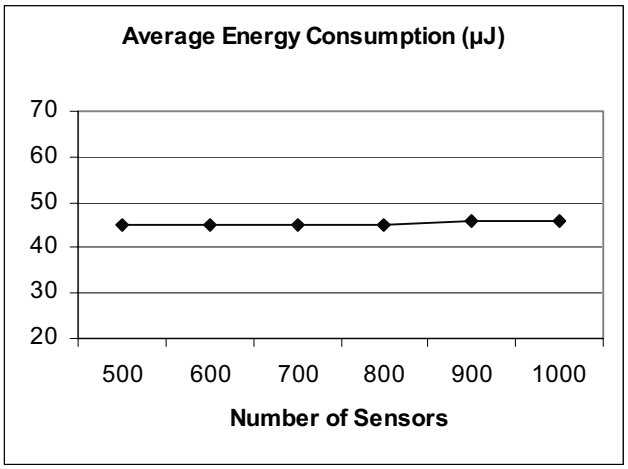

Figure 1. The average energy consumed by sensors in a network with 10 clusters.

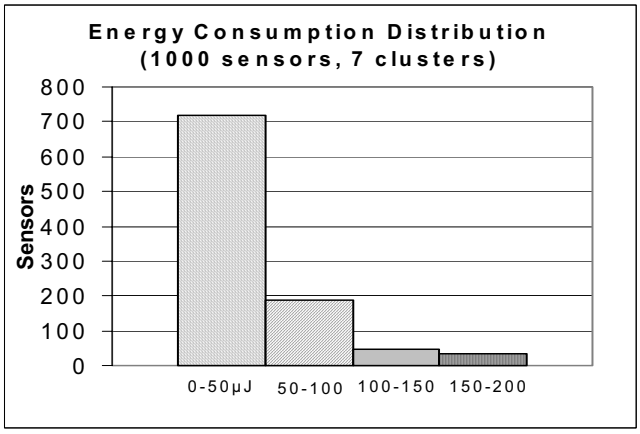

Figure 2. Distribution of sensor energy consumption with our approach.

A trusted third party Key Distribution Center (KDC) such as Kerberos is an alternative to pre-deployed keying. However KDC schemes are not energy-conserving. Because, for every secure session, sensor message transmission and reception are necessary for the session key establishment. After simulating Kerberos, we observe that the majority of the sensors in Figure 3 dissipate energy in the 1.5 - 3.0mJ range, which is too high. (Remember that our approach consumed energy in the µJ range.)

6. Conclusions and future work

We have presented an energy conserving method to provide key management for sensor networks. The method

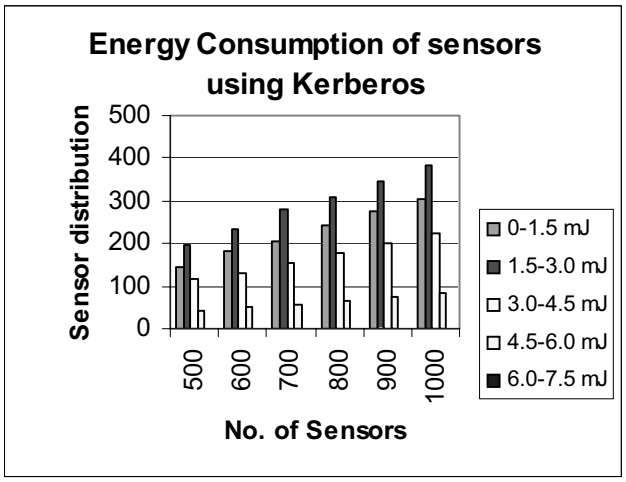

Figure 3. Distribution of sensor energy consumption using Kerberos

uses pre-deployed symmetric keying. A critical observation is that sensor-to-sensor secure channel establishment is not necessary for many monitoring applications. Therefore, pre-deployed keying has become sufficient, cost-effective approach to provide a keying infrastructure for security protocols that use those keys. The overhead per sensor appears to be feasible, and none of the sensors are required to store large numbers of keys. Moreover, our approach supports key revocation and renewal mechanisms, as well.

Our work is an improvement in the energy-aware design of the key management functionality for limited environments like sensor networks.

Formal analysis about the security strength of the proposed scheme remains as future work. The bootstrapping phase of the network may require some additions to the protocol, as proposed by Bobba, et al in [16]. An open problem is to design a lightweight intrusion detection mechanism to detect compromised nodes in the network.

References

[1] J. Rabaey, et al., "PicoRadios for Wireless Sensor Networks: The Next Challenge in Ultra-Low-Power Design," in Proceedings of the International Solid-State Circuits Conference, (San Francisco, CA), February 2002.

[2] S. Hollar, "COTS Dust," Master's thesis, Electrical Engineering and Computer Science Department, UC Berkeley, 2000.

[3] M. Younis, M. Youssef, and K. Arisha, "Energy-Aware Routing in Cluster-Based Sensor Networks," in Proceedings of the 10th IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS2002), (Forth Worth, TX), October 2002.

[4] V. Raghunathan, C. Schurgers, S. Park, and M. B. Srivastava, "Energy Aware Wireless Microsensor Networks," IEEE Signal Processing Magazine, March 2002.

[5] E. Shih, B. Calhoun, S.-H. Cho, and A. Chandrakasan, "Energy-Efficient Link Layer for Wireless Microsensor Networks," in Proceedings of the Workshop on VLSI 2001 (WVLSI '01), (Orlando, Florida), April 2001.

[6] K. Arisha, M. Youssef, and M. Younis, "Energy-Aware TDMA-Based MAC for Sensor Networks," in Proceedings of the IEEE Workshop on Integrated Management of Power Aware Communications, Computing and Networking (IMPACCT 2002), May 2002.

[7] G. Gupta, M. Younis, "Performance Evaluation of Load-Balanced Clustering of Wireless Sensor Networks," in the Proceedings of the 10th International Conference on Telecommunications (ICT'2003), Tahiti, Papeete – French Polynesia, February 2003 (to appear).

[8] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J. D. Tygar, "SPINS: Security Protocols for Sensor Networks," Wireless Networks, vol. 8, no. 5, pp. 521-534, 2002.

[9] M. Horton, et al., "Mica: The commercialization of microsensor motes," Sensors Online Magazine, April 2002. http://www.sensorsmag.com/articles/0402/40/main.shtml.

[10] D. Carman, P. Kruus, and B. Matt, "Constraints and approaches for distributed sensor network security," Tech. Rep. 00-010, NAI Labs, September 2000. http://download.nai.com/products/media/nai/zip/nailabs-report-00-010-final.zip.

[11] W. Fumy and P. Landrock, "Principles of key management," IEEE Journal of Selected Areas in Communications, vol. 11, pp. 785-793, June 1993.

[12] E. Shih, S.-H. Cho, N. Ickes, R. Min, A. Sinha, A. Wang, and A. Chandrakasan, "Physical Layer Driven Algorithm and Protocol Design for Energy-Efficient Wireless Sensor Networks," in the 7th ACM Annual International Conference on Mobile Computing and Networking 2001), July 2001.

[13] S.-H. Cho and A. Chandrakasan, "Energy Efficient Protocols for Low Duty Cycle Wireless Microsensor Networks," in International Conference on Acoustics, Speech, and Signal Processing 2001, May 2001.

[14] A. Perrig, et al.,, "SPINS: Security Protocols for Sensor Networks," in Seventh Annual International Conference on Mobile Computing and Networks (MobiCOM 2001), (Rome, Italy), 2001.

[15] L. Eschenauer and V. D. Gligor, "A Key Management Scheme for Distributed Sensor Networks," in 9th ACM Conference on Computer and Communications Security—CCS 2002, November 2002.

[16] R.B. Bobba, L. Eschenauer, V. Gligor, W. Arbaugh, "Bootstrapping Security Associations for Routing in Mobile Ad-Hoc Networks," Institute for Systems Research, ISR Technical Report 2002-44, May 2002.