Securing Sensor Networks Using A Novel Multi-Channel Architecture

Chao Gui, Ashima Gupta and Prasant Mohapatra
Computer Science Department
University of California, Davis
Davis, CA 95616
{guic,ashgupta,prasant}@cs.ucdavis.edu

***Abstract*— In many applications of sensor networks, security is a very important issue. To be resistant against the various attacks, nodes in a sensor network can establish pairwise secret keys[5], [6], [10], authenticate all communications with cryptographic functions[8], and also apply secure information aggregation schemes[13] or hop-by-hop filtering methods[14], [16]. However, these security measures can take considerable overhead in terms of storage, communication and computation, which are scarce resources in sensor nodes. Previously proposed security measures can only resist against a limited number of compromised nodes, which we define as the *resistance level*. In this paper, we propose a separate solution to any security measure. This technique either significantly reduces the overhead, or increases the resistance level without increasing overhead. The solution is based on a new "Mixed Multi-Channel" (MMC) architecture. In this design, each node can only use one fixed channel. The whole network is thus divided into multiple "planes" by the different planes. Exploiting the characteristics of multi-channel communication, a series of methods are proposed, such as MMC-1, MMC-k and MMC-r. We then present designs to integrate the methods with current security measures, and analyze their resistance level and energy conservation.**

***Index Terms*— Sensor Networks, Security, Multi-channel Communication, Key Establishment, Information Diffusion**

I. Introduction

Wireless sensor networks help people to accurately gather information, monitor and react to events from the physical world. In hostile application environments, such as battlefield surveillance, security becomes a major concern. An opponent can attack a sensor network in various ways. He can introduce "bad" nodes into the network, or, capture and reprogram original sensor nodes. The attack can be on data confidentiality, authenticity, integrity, freshness or availability. A malicious node can impersonate as if it were a large number of nodes. This type of attack in sensor network is identified as the Sybil attack [11]. In data injection attack, for example, the malicious Sybil node can multiply the amount of false data that can be injected into the network.

To resist against the attacks, secure communication between sensor nodes is necessary while maintaining scalability and flexibility to topology changes. This is usually based on pairwise key management and authentication of control packets and data reports. Several research efforts have focused on this measure[10], [5], [6]. The various key management techniques can be broadly classified into the trusted server approach [7] [12], self enforcing [2] and pre-distributed [9].

In [15], Zhang and Cao describe a scheme in which sensors are randomly divided into groups and each group shares a symmetric key. Messages are attached with multiple MACs corresponding to different group keys to protect message integrity and detect false messages. To address the data injection attacks, Przydatek *et. al.* proposed the secure information aggregation (SIA) scheme[13]. It is based on aggregation computations that give statistically approximating results and on interactive proofs at the aggregator nodes. The scheme ensures a good approximation of true values (medium or min/max of the data reports), even if a small number of nodes or the aggregator node are compromised. On the other hand, new efforts have devoted to data authentication and filtering of the injected data[14], [16]. The basic idea is that, in the example of [16], each node shares different secret keys with the base station, each of the neighbor nodes, and certain nodes multiple hops away. Then, each data report is attached with two MACs by the generating node, one is computed using the key shared with BS and one using the key shared with the node t hops away along the path towards the BS. While t reports about the same event from different nodes are packed together for delivery to the BS, each forwarding node can verify the MAC of one report. During this interleaved hop-by-hop authentication process, reports with wrong MACs or not agreed by t other reports are identified and filtered out. Finally, reports that reach the BS are verified by the it.

Secure services and operations in sensor networks take considerable overhead. In terms of storage, pre-stored secret information is needed for each shared key with other nodes. In terms of computation, a sensor node needs to compute multiple cryptographic functions for MACs of each data report or control packet. In terms of communication, extra dialogs among nodes are necessary to establish and update shared secret[15], and identify malicious nodes. For any security scheme, we can define its resistance level, which is the maximum number of compromised nodes it can resist before the scheme can be feasibly broken. To deal with this security breach, compromised nodes should be identified and the innocent nodes should update their shared keys to regain the key secrecy.

In this paper, we deal with this security breach problem by enhancing the resistance level of any established security schemes. Based on a new multi-channel architecture, we investigate several methods that can exploit the multi-channel

capacity to dramatically increase the resistance level of current security schemes, without changing the scheme. The novelty of this multi-channel exploration is that low-cost radio interface is used at each node so that it can only use one channel. This assumption not only agrees with the popular goal of minimizing the size of each motes, but also it introduces new architectural design for the whole network. Multiple groups are formed in the network by the different channels, and we name each group as one *plane*. Multiple planes in the network can be homogeneous, with each plane running the same functionality simultaneously. Or, the planes can be running different functionalities, and jointly carry out the full task of the network.

The rest of the paper is organized as the following. In Section II, we present the new Mixed Multi-Channel architecture. In Section III, we describe our proposed solution. Section IV and Section V present the implementations of scheduling the secure plane in the network, using pre-stored schedule and dynamic schedules, respectively. In Section VI, we analyze the resistance level and the energy consumption of the proposed solutions. The conclusions are presented in Section VII.

II. The Mixed Multi-Channel Architecture

We assume that the open radio media shared by the wireless links can be divided into k different channels. Here, k is a limited and constant number such that each channel is free from any interference from other channel. Thus, if the channel division is based on radio frequency, we assume non-overlapping frequency bands. Currently, the MICA motes provided by the CrossBow company can work under 900 MHz band or 2.4 GHz band. In fact, it is possible to pick several non-interfering bands in the ISM bands (900 MHz, 2.4 GHz and 5.8 GHz). Therefore, conservatively, k can have a constant value between 2 to 10.

We also assume that the sensor nodes do not have multi-channel radio interface. With each node using one specified channel, one can imagine an example deployment of mixing MICA2 motes (working on 900 MHz or 433 MHz) with MICAz motes (working on 2.4 GHz). This assumption agrees with the requirement of minimizing each sensor node, and gives us new advantages in data delivery performance and network security. This paper is focused on investigating these advantages.

Finally, we propose the Mixed Multi-Channel (MMC) architecture for sensor networks. The N nodes in such a sensor network is equally divided into k groups, with each group assigned to one of the k available channels. The nodes in any group can all communicate using the assigned channel, and no other channels. For each group, the nodes of the group is evenly distributed across the whole network field. In other words, if the area of the entire network field is A, then the density of nodes in any group is $N/(k \cdot A)$. The density of the whole network nodes is N/A.

Figure 1 shows an example of this mixed multi-channel deployment with three channels. Divided into different channels, the network forms three planes, one for each group. The nodes at each plane is dense enough to form a connected

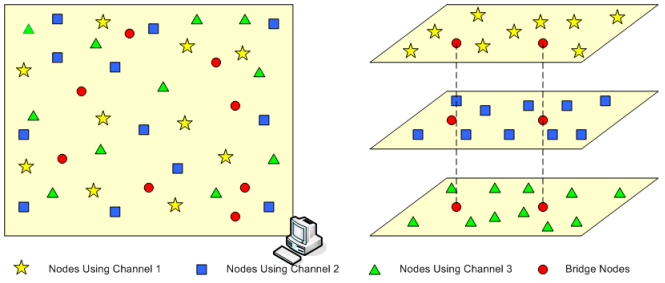

Fig. 1. An example of the mixed multi-channel deployment.

network topology for the plane separately. Since the sensor modules in current motes have relatively much smaller sensing range compared to the radio transmission range, a deployment that meets the sensing coverage requirement will be overly dense for network connectivity. Thus, in single channel setup, the media access protocol needs to deal with overly crowded mote population. On the other hand, the mixed multi-channel architecture will significantly alleviate this problem by its channel clustering technique that is therefore, energy efficient.

Other than the normal nodes, the different planes of the network are connected by certain number of bridge nodes, which are shown in Figure 1 as dark nodes. They are of more high-end form factor, and we assume each bridge node is equipped with k radio interfaces[1]. Each bridge node is capable of simultaneously communicating via different channels, thus the different network planes are connected to each other at the bridge nodes.

With regard to the security issues, we have the following assumptions: 1) For a sensor node, it is impossible to know which channel it is using until it is captured and compromised; 2) There are physical boundaries that limits a node in certain channels. Once captured, a node belonging to channel A cannot be freely changed to use channel B.[2] This physical limitation enforces the partition of the network, which is a basis of the methods presented in Section III.

III. General Scheme

In this section we propose a series of methods that utilize the MMC architecture to enhance the resistance level of current security measure. We start from the basic MMC-1 method that applies the security measure to one plane only, while nodes in other planes communicate in the normal mode. The MMC-1 method reduces the extra overhead in secure measures while increasing the resistance level. However, if each plane applies an independent copy of the secure measures, the resistance level will be further increased. This method is called the MMC-k method, where k is the number of planes. Finally, in the third method, only one plane works in secure mode at any time, but the network follows a secret schedule for deciding the secure plane. Named as the MMC-r method, it achieves same resistance level of MMC-k method, but the overhead is similar to MMC-1.

[1]In practice, a bridge node can be formed by connecting k motes to a PC-104 single-board computer

[2]For example, a MICA-2 mote cannot effectively be converted into a MICA-2z mote.

A. The MMC-1 Method

In this method, only a fixed plane employs the secure measures such as pair-wise key establishment scheme and the data authentication and filtering protocol. Data delivery protocols are running at the k planes independently, and data are collected from the planes simultaneously and independently. The data collected from the secure plane are trusted. The sink node can use the trusted data, which is sampled uniformly from the whole network field, as a reference for checking the data from other planes. While using this model of checking the data collected from the normal planes, any data that deviates too far from the model will be identified and ignored. In this manner, even though a node in normal plane can be captured and be made to inject data without resistance from any security measure, the injected data and its source will be identified and ignored by the sink.

The overhead for security measures in MMC-1 method is reduced. Instead of having N nodes setting up pairwise keys, only N/k nodes are needed to do this operation and perform the authentication computation for their packets. The majority of the nodes are freed from this computation. This will result in significant energy savings.

B. The MMC-k Method

In this method, all k planes employ the secure measures. It is important that each plane uses an independent copy of the secure measure. In this manner, if the opponent captured a node in plane A and one node in plane B, the opponent only gets two independent pieces of information. There is no value of combining information from nodes from different planes, as it only does for nodes from the same plane.

1) MMC-k with Polynomial-based Key Distribution: Polynomial-based key pre-distribution scheme by Blom[3] and Blundo[4] is a popular method for establishing pairwise keys in sensor network[10]. In the key pre-distribution scheme, the key manager first generates a bivariate t degree polynomial

$$f(x,y) = \sum_{i,j=0}^{t} c_{i,j} x^i y^j.$$

Defined over the finite field of $GF(p)$, where p is a large enough prime number, the polynomial should also satisfy the property of symmetry, i.e., $f(x,y) = f(y,x)$ for any two nodes x and y. Thus the secret key shared by nodes x and y is $f(x,y)$. For any node i, it only stores a partial share of polynomial $f(x,y)$, which is a single variate polynomial $f(i,y)$. Since $f(i,y)$ is a t degree polynomial of y, node i needs to store $t+1$ coefficients. Thus, if node i needs to communicate with node j, it computes $f(i,j)$ as the shared key with node j.

When MMC-k method is combined with this key pre-distribution scheme, the key manager needs to generate a set of k polynomials, $f_1(x,y), f_2(x,y), ...f_k(x,y)$. Nodes of plane I will only be associated with polynomial $f_I(x,y)$. Thus, the independence between different planes is achieved. In the original distribution scheme, Blundo[4] has proved that combining the information stored in t compromised nodes will not reveal the pairwise key between any other innocent nodes. Thus, the resistance level of this scheme is t. When the MMC-k method is applied, only combining information from two nodes of the same plane will be useful to the opponent.

2) MMC-k with Polynomial Pool-based Key Distribution: For the basic polynomial-based key distribution scheme, once the coefficients of the t degree polynomial is revealed, the opponent can know the public keys between any two nodes. To address this problem, a pool of multiple polynomials can be used instead of one single polynomial. In [10], a grid-based polynomial pool scheme is proposed. When MMC-k method is combined with this polynomial pool-based key distribution scheme, k grids, i.e., polynomial pools, need to be generated, to achieve plane independence. However, the number of polynomials in each pool is $m' = \sqrt{N/k}$, as compared to $m = \sqrt{N}$ in the original polynomial pool scheme. Thus, the total number of different polynomials needed is $2\sqrt{k \cdot N}$, as compared to $2\sqrt{N}$ in the original scheme.

C. The MMC-r Method

Let us compare the MMC-1 method and MMC-k method. In MMC-1, only one k-th of the nodes in the network are applying the standard security measures. The majority of the nodes do not incur the cost of storing keying information and calculating MACs for each transmitted packet. For miniature sensor motes, this extra cost can drain their power much faster. While using MMC-1, the nodes in the secure plane can be out of energy sooner than other nodes. In this case, one can add N/k nodes into the network to form a new secure plane. On the other hand, the shortcoming of MMC-1 is its dependence on one certain plane. Once it is cracked down, the data received is not trustworthy or verifiable.

The resistance level of MMC-k method is much higher than MMC-1, and the overhead is not much more than that of standard security measures. We assume that the opponent can only compromise the nodes one at a time, then the planes can only be breached one at a time. This can be detected immediately if the network is continuously cross checking the data gathered from the trusted planes. However, the shortcomings of MMC-k is still its overhead. Current security measures can be too much burden for the tiny motes, and is a limiting factor for further down-sizing the sensor motes.

Considering the MMC-1 and MMC-k method, another option emerges that combines the properties of both methods. In MMC-r, the main idea is still to have one secure plane in the network at each time, however, the secure plane is not a fixed one, but any plane in the network following a schedule. The overhead of secure measure using MMC-r will be comparable to MMC-1, with extra overhead of secure plane scheduling. On the other hand, the resistance level of MMC-r will be much higher than MMC-1, since there is no single target of attack. Up until all the planes are cracked, the network can always use one secure plane as the source of verified and trusted data. In this sense, the opponent still needs to compromise at least t nodes in every plane. Thus, the resistance level is the same as the MMC-k method.

We assume that the sensor nodes are loosely synchronized. The time is divided into slots of length Ts, and a plane in secure mode will remain in that mode for the entire length of a time slot before either being replaced by another plane, or assigned to continue to the next slot. The length of a slot, Ts, is a system parameter, and we expect its value in the order of several minutes. The time slots are sequentially numbered, denoted by integer variable s. Thus, we can define the schedule of secure plane as a function $f : \{1, 2, ..., \infty\} \rightarrow \{1, 2, ..., k\}$, where $I = f(s)$ means that at s-th time-slot, plane I should be the secure plane. At certain point before the start of slot s, all the nodes in the network has computed $f(s)$. The nodes in plane $f(s)$ will conduct the secure measure during the slot, while all other nodes will be plain communication mode. The secure plane schedule is pre-stored at all the nodes in the network. Thus, all nodes in the network can agree on the schedule without the communication overhead and issues. However, this raises a security threat, i.e., the opponent can compromise one node and get the information about the global schedule of the secure plane. If the opponent knows the full schedule of the secure plane, the MMC-r method will be no stronger than the MMC-1 method. In Section IV, we will discuss how to solve this problem using the information diffusion principles.

IV. Implementation of Secure Plane Scheduling

A. Computational Diffusion

The idea of computational diffusion is to decompose the function of scheduling, $I = f(s)$, into a set of k functions, $f_i(s) : \{1, 2, ..., \infty\} \rightarrow \{0, 1\}$, $(1 \leq i \leq k)$. For all integer slot number s:

$$f_i(s) = \begin{cases} 1 & : \quad if\ i = f(s) \\ 0 & : \quad if\ i \neq f(s) \end{cases}$$

Then, instead of distributing the schedule function $f(s)$ to every node, only function $f_I(s)$ is distributed to nodes in plane I. Before the start of the s-th time slot, all nodes will compute their share of the function in the set. For any node i in plane I, the result of $f_I(s)$ will be 1 if plane I is scheduled during the time slot, otherwise, $f_I(s)$ will return 0. Thus, the definition of function set makes all nodes in the network agree on the global schedule. We should ensure that capturing information about one function in the set will not reveal any information about any other functions. Thus, compromising one node in the network, the opponent will only be able to know the schedule for one plane only.

We now discuss one implementation of the schedule function and the corresponding function set, that can satisfy this property. We need a pseudo-random function $g(s) : \{1, 2, ..., \infty\} \rightarrow \{1, 2, ..., M\}$, where M is an integer of reasonably large value. The result of function g(s) should be equally likely to be any integer between 1 and M. If we partition the set {1,2,...,M} into k subsets of equal size, we use P_k^M to denote the k-th partition. Then, the global schedule function $f(s$ will be defined as the following for all integer slot number s:

$$f(s) \equiv i \quad iff\ g(s) \in P_i^M.$$

The corresponding function set can be shown as:

$$f_i(s) \equiv \begin{cases} 1 & : \quad g(s) \in P_i^M \\ 0 & : \quad g(s) \notin P_i^M \end{cases}$$

For any node i in Plane I, the pre-stored information about the secure plane scheduling is the pseudo-random function g(s) and the partition P_i^M. The memory for storing each partition is $\frac{M}{8k}\lceil logM \rceil$ bytes. If there are 8 planes and M=4096, the memory requirement for each partition is 768 bytes. The partition of {1,2,...,M} should avoid any pattern. In this manner, the information from one node only reveals the elements in one partition, and there is no information about how the remaining elements are partitioned. Thus, the schedules for other planes are protected.

B. Neighbor-supported Diffusion

We can apply neighbor supported diffusion [15] for diffusing the plane schedule. First, we generate k bivariate polynomials over the finite field GF[q] where q is a large prime number. The polynomials are of the following form:

$$e_I(x, y) = \sum_{0 \leq i \leq t, 0 \leq j \leq \mu} p_{i,j} x^i y^j.$$

These k polynomials are assigned to the k planes as the encryption functions. For plane I, Let $f'_I(s)$ denote the plane's encrypted version of the global scheduling function $f(s)$. The en-decryption of $f(s)$ at plane I is expressed as follows:

$$f'_I(s) = f(s) + e_I(x, y),$$
$$f(s) = f'_I(s) - e_I(x, y)$$

For node u in plane I, we assume it can establish trusted communication with μ of its neighbors in the same plane. Let $u_1, u_2, ..., u_\mu$ be the ID of the trusted neighbors. Then node u will compute the values of $e_I(u, u_i)$ for i from 1 to μ. It will give the value of $e_I(u, u_i)$ to neighbor u_i, and it will compute a $e_I(u, r)$ value for itself to keep, where r is a random value. Then, node u removes functions $e_I(x, y)$ and $f(s)$. We should note that each of node u's trusted neighbor will also give its own value to u to keep. The value given by neighbor u_i will be $e_I(u_i, u)$. This completes the pre-distribution process.

When node u needs to consult the schedule function $f(s)$, it will collect the μ values of $e_I(u, u_i)$ from its trusted neighbors $u_1, ..., u_\mu$. With its own value of $e_I(u, r)$, node u will have $\mu + 1$ different values from the polynomial $e_I(u, y)$. Since it is a μ degree polynomial on y, node u can now reconstruct the polynomial with the collected $\mu + 1$ values. With $f'(s)$ and $e_I(x, y)$, node u can decrypt the schedule function $f(s)$.

C. Combined Diffusion

We combine the previous two diffusion methods to make the plane schedule a stronger secret. We will still have the set 1,2,...,M partitioned into k subsets. The k encryption polynomials $e_I(x, y)$ will also be needed. Then, for plane I, the integer values in partition P_i^M will be encrypted using the polynomials. That is, for each integer value v in partition P_i^M, we will compute v' as $v' = v + e_I(x, y)$. For node u in plane I, it will still distribute μ values of $e_I(u, u_i)$ to its

μ trusted neighbors. This will diffuse the encryption function as a whole in the same manner described in the previous subsection. Then, node u removes the original values in partition P_i^M and only keep the encrypted value.

Hence, the opponent needs to compromise multiple nodes in one plane to break the encryption function for that plane. Even, after that, the opponent still cannot know the schedule of other planes.

V. Implementation of Dynamic Secure Plane Scheduling

In this section we present a study of dynamic scheduling of secure planes. Time is divided into superframes which is subdivided into time quanta. A time quantum is the time required for a plane to perform an observation and communicate it to the sink. We still assume that the keys (intra plane as well as group keys for communicating with the sink) in the sensors are pre-deployed. We assume that the sink is a secure and high resource device in terms of memory and processing power. The sink will be the central scheduler for time scheduling of the active planes within a superframe. The primary parameters involved in dynamic scheduling are:

1) Number of secure planes (one , r, all),
2) Superframe size (fixed or variant),
3) The key management scheme used per plane (homogeneous or heterogeneous).

We discuss the impact of each of these parameters in the following subsections.

A. Least Variants

The most simplistic case with the least number of variants is to assume a fixed superframe size of k+1 time quanta, where k represents the number of planes. A common message is sent to all the sensors on their respective planes. The key management scheme across all planes is uniform though the keys are different and the number of secure planes per time quantum is one. We assume that in every superframe every plane will be in secure mode exactly once and each in a different time quantum. We require k+1 time quanta per superframe. The superframe will comprise of k time quanta, one per plane and one more to receive and process the next scheduling message. We assume that every plane has a group key. The sink will employ a card shuffling algorithm to generate a random schedule, i.e. take the numbers from 1 through k and shuffle them. We can use the Knuth shuffle or Fisher-Yates shuffle [1], which is a linear time algorithm for this purpose. Based on the generated random schedule, the sink will send a k byte structure to all the sensors. This k byte structure could be as simple as a k byte array where the same byte is replicated k times, each byte holds some protocol information. We will discuss content of the k byte structure later. Each byte in the structure is encrypted by one of the public (group) keys of a plane. The index of the byte to be encrypted by a particular plane key is determined by the generated random schedule 2. The next scheduling message will be a permutation on the previous one, so that no consecutive scheduling messages have the same schedule. The mathematical model of the schedule

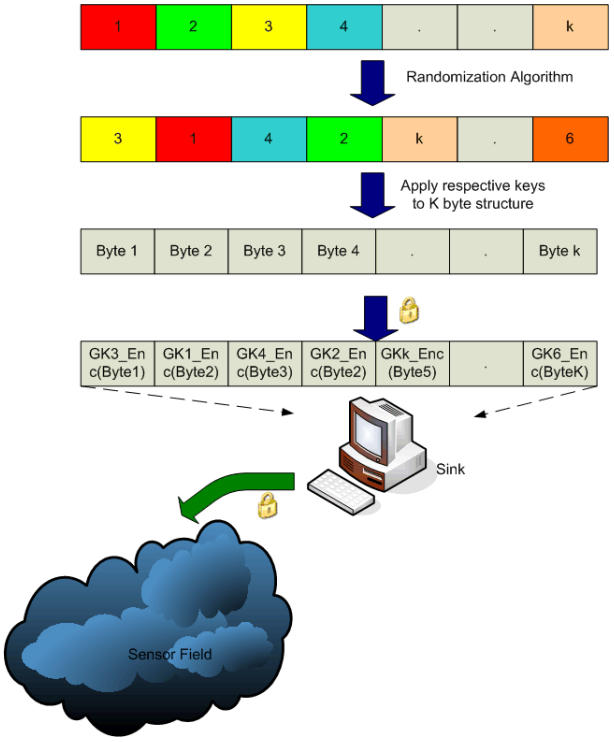

Fig. 2. Construction of a Dynamic Scheduling Message

distribution problem if a common scheduling message is being transmitted to all the sensors can be represented as a function for each plane, $f_1, f_2, ..., f_k$ such that f_i.(scheduling message) = time code i in superframe ($1 \leq i \leq k$).

For example if there were 4 planes available and the random number generated was 3412 then the first byte is encrypted by the public key of the third plane, the second byte encrypted by the public key of the fourth plane, the third byte by the public key of the first plane and the last byte by the public key of the second plane. This message is transmitted to all the sensors in the field. Each sensor will try to decrypt the message one byte at a time. The index of the byte that a sensor decrypts will be the position of the time quantum in the superframe that it has to be in secure mode. Once a sensor node has deciphered its portion of the message it need not parse any further. It is assumed that the keys are pre-deployed, but the schedule is not. When a sensor deciphers its schedule it can then perform the sensing and communicate to the sink in the required mode. Note that a sensor is sure that other sensors in its plane will also be active during that time quantum. The advantages of this scheme are:

- Schedule will vary per superframe making it difficult for eavesdroppers to capture useful information.
- The secure channel functionality is not limited to one plane. It can vary, so if one secure channel is compromised by an adversary, the sink can mark and eliminate that. It can instead use another one.
- This distributes the overhead of secure mode functioning over all planes with a good probability. It is therefore a better approach than just changing the active time of a constantly secure plane in the sensor field.

- Intrusion detection can be performed since at least one plane in every time quantum is executing in the secure mode. The sink can, therefore, compare the data received and discard corrupt data and mark its source if a plane has been breached.

The main disadvantages if this approach are:

- A common scheduling message implies the nodes will have to transmit and process extra information (that belongs to a different plane) not useful to its plane.
- The superframe structure could be too small resulting in high overhead of transmitting a scheduling message every superframe.
- As compared to the pre-deployed scheduling scheme there will be more protocol and computational overhead to communicate and decrypt schedule information. We will discuss further how to offset this overhead by increasing the superframe length in a later part of this section.
- In contrast to the static scheduling mechanism, all the nodes will spend one extra time quantum per superframe to obtain to obtain the new schedule.

B. Varying number of secure planes

The above scheme can be extended to multiple 'r' number of secure planes. Given a fixed superframe size of k, the sink can generate the schedule by performing the card shuffling algorithm on numbers 1 to k, k times. Each iteration is performed on the previously generated schedule. The result of each iteration can be truncated to r entries. Every iteration will correspond to a time quantum in the superframe. The schedule can now be disseminated by the sink to the planes in two ways. The first is using a common scheduling message. The sink can use a structure of size $X \cdot k$ bytes, where X is the minimum number of bytes required to represent the superframe. The first X bytes are for the first plane, the next X bytes for the second plane,...the last X bytes are for the kth plane. The bits corresponding to the time quanta for which a plane has to be in secure mode are turned on. The first X bytes are then encrypted using the group key of the first plane, the next X bytes using the group key for the second plane... and the last X bytes by the group key of the kth plane. An example of this process is depicted in the figure 3 with k set to 5 and and r to 3. Here X will be just one byte since 8 bits can represent 5 time quantum schedule (one bit per time quantum). The second method of dissemination is sending a separate message to each plane with its schedule. A per plane scheduling message will have lower communication overhead while a common message will be more secure against eavesdropping.

In general the probability of causing a security breach after capturing T nodes if there number of secure planes is 1, r or all will be the same as that for the MCS-1, MCS-r and MCS-k schemes as discussed in Section VI. This is because we do not assume that the attacker will pick out a node based on his previous knowledge or attempts.

The advantage of this scheme is that more number of secure planes strengthens the security of the network. The disadvantage of a common scheduling message will be the

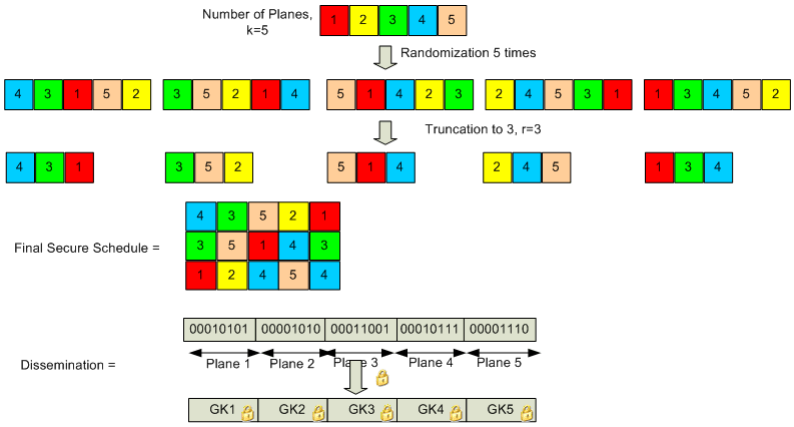

Fig. 3. Construction and Dissemination of a Dynamic Scheduling Message for Multiple Secure Planes.

communication and processing overhead incurred by the nodes in forwarding and processing data that is not required by their plane. Executing all planes in the secure mode does not provide much added advantage, results in high cost and minimizes the impact of multichannel plane scheduling paradigm. However in general it is observed that the level of security of the sensor network also increases with the increase in the number of secure planes.

It can be observed that, the protocol cost(communication and processing overhead) is directly proportional to the number of secure planes used.

C. Varying the Superframe Size

In a fixed superframe size we assumed that the scheduling message is transmitted by the sink after every $k + 1$ time quanta. if k is small the resulting overhead will be high. This overhead can be offset by increasing the superframe length and allowing duplicates in the schedule. This may increase the size of the scheduling but the resulting communication overhead will become proportionally smaller. This can be implemented in two ways. The first is by increasing the number of bytes in the k byte structure so that there is a byte for each time slot in the superframe. Using this scheme, the sensor nodes will have to process the entire message to the end to discover the secure mode schedule of their planes. The other way is to use a structure of size $X \cdot k$, where X is the number of bytes required to represent the number of time quanta in the superframe. There will be X bytes available per plane and the sink can turn on the bits that reflect the secure mode schedule of a particular plane before encrypting it with the group key of that plane. The former approach is more extensible, frees the bytes in the k-byte structure to encode other information and for shorter superframe sizes will result in a smaller scheduling message. However, in the latter scheme once a node has successfully decrypted its word, it need not process the rest of the message and the scheduling message size will be shorter as compared to the former approach for longer superframe sizes.

The cost of the protocol is, therefore, directly proportional to the scheduling message size but inversely proportional to

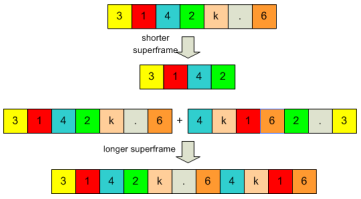

Fig. 4. Variable length Superframe

the superframe size. The security level of the sensor network would decrease with increase in the superframe size, i.e. it would be prone to eavesdrop. Hence,

$$C \propto (NP)(SMS/SS) \tag{1}$$

Here, C is the Protocol Cost, NP the number of secure planes used, SMS is the scheduling message size and SS denotes superframe size.

Furthermore, if the restriction of a fixed size superframe is removed, then the sink will also have to transmit the superframe length in the schedule structure. This information will have to be made a part of every plane's portion in the scheduling message structure thereby further increasing the length of the message. A variable superframe length implies that in one superframe there could be two observations performed while in the next one there could be five. The sink can generate a random number between 1 and a configurable maximum superframe length, to find the length. The sink will then generate a schedule as described in Section V-A and truncate it or extend it (by generating more schedules and concatenating it) based on the random number generated. This will result in a new varied length schedule, as shown in Figure 4. The added variability will strengthen the security of the network further but incur a cost in terms of computation at both the sensor and the sink ends. In this section we have assumed that only one plane is in secure mode (MCDS-1); however this is easily extensible to multiple secure planes per time quanta using the techniques described in V-B.

D. Heterogeneous vs homogeneous key management

If the key management scheme is heterogeneous, i.e., not the same across the planes, then it would be better to send a separate scheduling message to all the different planes. The distribution of the scheduling algorithm can be mathematically modeled as a function for each plane f1, f2, f3....fk such that

- f_1.(scheduling message 1)=time code 1 in superframe
- f_2.(scheduling message 2)=time code 2 in superframe
- ...
- f_k.(scheduling message k)=time code k in superframe

However synchronization will be an issue since different key management algorithms have different number of handshakes and unless all of them take the same amount of time, a time quanta will have to last as much as the largest amount of time that any of the schemes would take to perform a complete key management handshake. This would be an inefficient but a highly robust approach to secure data aggregation.

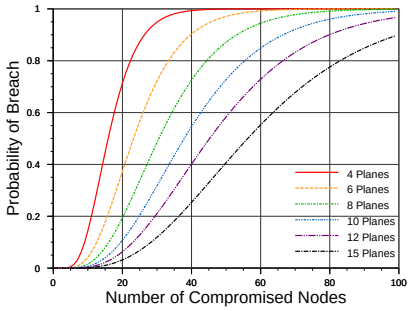

Fig. 5. Probability of breach. (t=4)

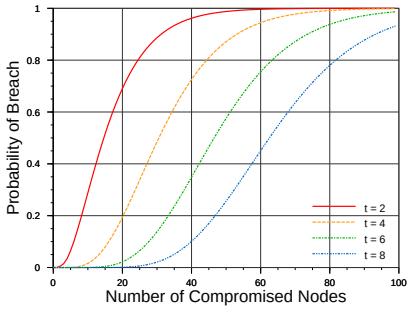

Fig. 6. Probability of breach. (k=8)

We are currently researching the dynamic scheduling scenario to quantify the proportionate impact of the parameters outlined in this section on the protocol cost.

VI. Performance Evaluation

A. Analysis of Resistance Level

1) MMC-1 Method: Let us assume the security measure applied in the secure plane has a resistance level of t. We need to determine the resistance level of the whole network. The opponent can only tell if a node is in secure plane after the node is captured and fully compromised. When a node is captured, the probability of the opponent to pick the node in the secure plane is $1/k$. The MMC-1 scheme has rendered it useless to compromise a normal node, thus the opponent must compromise t nodes in the secure plane to break the applied security measure. Suppose T nodes have been compromised, the probability that at least t of the compromised nodes are in the secure plane is the following:

$$Pr(breach) = 1 - \frac{\sum_{i=0}^{t-1} \binom{T}{i}(k-1)^{T-i}}{k^T} \tag{2}$$

In this expression, $\binom{T}{i}(k-1)^{T-i}$ is the number of cases that there is exactly i nodes in Plane 0. We sum up all cases that Plane 0 has less than t nodes. The denominator is the number of all possible cases, in which each of the T nodes can be equally probable to be in any of the k planes.

Figure 5 shows the probability of breach as the function of the number of compromised nodes, with varying number of planes. With the resistance level of the secure plane being fixed at 4, and there are 4 planes, the opponent needs to break 24 nodes in order to have 80% sureness that the secure

plane is breached. When there are 15 planes, the same number increases to 83. Thus, we can increase the resistance level from 4 to significantly higher value. Similarly, Figure 6 shows the same function with varying resistance level, t, at the single secure plane. With t=8 and 8 planes, the 80% sureness of breach requires at least 82 compromised nodes. Thus, an 82-to-8 increase of resistance level.

2) MMC-k and MMC-r Methods: Let us assume that the security measure applied in one secure plane has a resistance level of t. We need to determine the resistance level of the whole network. Let T be the number of nodes compromised by the opponent, the network can still collect secured data from any functioning plane, unless all the planes are broken. In this sense, among the T compromised nodes, there should be at least t nodes in every plane. We use $S_{k,t}(T)$ to denote the number of distributions of T distinguishable nodes into k different planes so that each plane contains at least t nodes. Then, the probability of breaking the whole network after compromising T nodes can be expressed by:

$$Pr(breach) = \frac{S_{k,t}(T)}{k^T} \tag{3}$$

When there are less than $k \cdot t$ nodes, there is no distribution that satisfy the requirement. Thus, $S_{k,t}(T) = 0$ $(T < k \cdot t)$. When $T = k \cdot t$, the number of distributions can be expressed as $S_{k,t}(k \cdot t) = \frac{(k \cdot t)!}{(t!)^k}$. Another special case is when t=1. $S_{k,1}(T)$ satisfies the following: $S_{k,1}(T) = k!S(T,k)$, where $S(T,k)$ is the number of partitions of set {1,2,...,n} into k non-empty subsets. $S(T,k)$ is also named as the *Stirling number of the second type*. For the general case of $S_{k,t}(T)$, we can use the generating function as help, which is the following theorem:

***Theorem** 1:* Let $G_{k,t}(u)$ be the exponent generating function of $S_{k,t}(T)$, i.e., $G_{k,t}(u) \equiv \sum_{n=0}^{\infty} S_{k,t}(n)\frac{u^n}{n!}$. $G_{k,t}(u)$ satisfies the following:

$$G_{k,t}(u) = (e^u - 1 - u - ... \frac{u^{t-1}}{(t-1)!})^k. \tag{4}$$

Proof: see appendix.

Figure 7 and Figure 8 show the probability of breach as the function of the number of compromised nodes, with varying number of planes and varying resistance level of single plane, respectively. Figure 7 indicates, that the curves climb up faster when the number of planes, k, is smaller. When $k \leq 6$, to reach 80% sureness of breach, the number of compromised nodes is close to $k \cdot t$, where t is the resistance level of one plane. However, when there are more planes, the required nodes are much more. Figure 8 shows that the resistance level of the whole network is significantly higher than that of one plane.

B. Analysis of Energy Savings

The MMC-1 and MMC-r methods save energy by limiting the security costs to one k-th of the nodes in the network. Using the hop-by-hop filtering methods, the injected data will be dropped in the secure plane though not in the other planes. We will use the following energy model to quantify and compare the energy consumption of both conventional and the multi-channel networks.

Assume the secure plane in the MMC network uses the same secure measures as the conventional network, and the

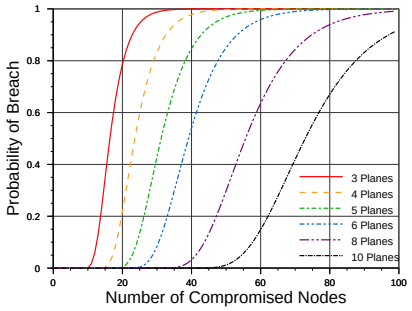

Fig. 7. Probability of breach. (t=4)

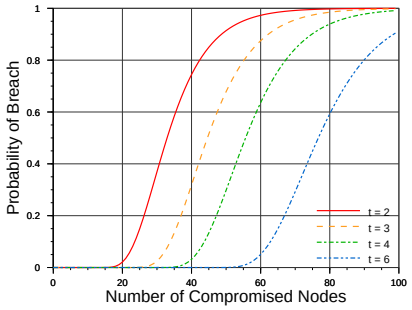

Fig. 8. Probability of breach. (k=8)

Statistical En-route Filtering (SEF)[14] is used for hop-by-hop filtering. In each data report, SEF attaches t MAC codes. Let L_r be the length of normal report without extra MACs, and L_t be length of the attached bits for each report. Thus, the length of an SEF report is $L'_r = L_r + L_t$. Let the amount of legitimate data reports and falsely injected data reports be 1 and β, respectively. For both conventional network and MMC network, we assume the routing protocols should route a report to the sink using approximately L/R_t hops. Here L is the distance to the sink and R_t is the transmission range. Thus, the average number of hops a report travels should be the same under both networks. For legitimate reports, we use H to denote the average number of hops they travel. The false injected reports will be verified and dropped while being routed. According to SEF, the probability of false report being detected after one hop is p_1. Thus, the probability that a false report will travel h hops is expressed by $p_h = (1-p_1)^{h-1}p_1$. Then, the average number of hops a false report can travel can be derived as the following:

$$H_f = \sum_{i=1}^{H} i \cdot (1-p_1)^{i-1}p_1 = \frac{1-(1-p_1)^H}{p_1}$$

Thus, the energy consumed to deliver all the reports in a conventional network using SEF is the following:

$$E_t = (L_r + L_t)(H + \beta\frac{1-(1-p_1)^H}{p_1})$$

For the MMC network, in the k-1 planes, both legitimate and injected data are forwarded to the sink, but the length of each report is only L_r. Thus, the same energy consumption in a

MMC network using SEF will be the following:

$$E'_t = \frac{k-1}{k} L_r H(1+\beta) + \frac{1}{k}(L_r + L_t)(H + \beta \frac{1-(1-p_1)^H}{p_1})$$

Our energy model should also include the computational cost in energy for authenticating the data reports. We use the following model for computational energy cost. In[8], Karlof *et.al.* have used *byte-time* to evaluate the cost of MAC computations. It refers to the duration that it takes to transmit a single byte of data over the radio, which in Mica2 motes is 0.42 *ms*. Most efficient message authentication code (MAC) algorithms uses block cyphers such as RC5. Currently, it takes 0.26 *byte-time* for a Mica mote to calculate a MAC for a packet using 64-bit RC5 algorithm[8]. Thus, we use δ to denote the ratio of per-byte computation time for calculating one MAC over the per-byte transmission time. We use c to denote the ratio of computational power consumption over the transmission power consumption. For example, the Mica2DOT motes uses 3.3V voltage for both processor and radio interface. The processor consumes 8mA of current while the radio consumes 27mA while in transmitting mode. Thus, for Mica2DOT mote, $c = \frac{8\times 3.3}{27\times 3.3} = 0.296$. Finally, we can calculate the computational energy cost relative to the transmission energy cost. For one unit amount of data reports to be relayed H hops, one MAC for each report, the computational energy cost for generating and verifying the MACs is expressed by $e_c = L_r \cdot H \cdot \delta \cdot c$. In SEF, each report has t MACs. For the injected reports, the malicious node will not use legitimate MAC algorithms, and each false report is only verified at the last hop and dropped. Thus, the total computational energy cost of authentication in a conventional network using SEF can be expressed as the following:

$$E_c = L_r \cdot H \cdot t \cdot \delta \cdot c + L_r \cdot \beta \cdot t \cdot \delta \cdot c.$$

Similarly, the same energy cost for a MMC network is the following:

$$E'_c = \frac{1}{k} E_c.$$

Finally, we need to combine the transmission energy cost and computational energy cost to derive the total energy cost. We use E and E' to denote the total energy cost of conventional network and MMC network, respectively.

$$E = E_t + E_c \quad E' = E'_t + E'_c.$$

These equations show the energy consumption as the function of number of planes (k), the amount of injected data (β) and the one-hop detection probability (p_1). Figure 9 and 10 show the energy consumption versus amount of injected data, with varying k. Shown in Figure 9, the energy consumption is always higher in a conventional network than a MMC network. It also shows a threshold of 4 planes, beyond which more planes will not yield much reduce in energy consumption. In Figure 10, when the one-hop detection probability is higher, the results are different. Conventional network consumes more energy only when injected data is less than 1.3 times of the legitimate data. The reason is that the injected data are more effectively filtered out in conventional network, while in MMC network, injected data are only filtered in the secure plane. We can even observe an increase in energy consumption with more planes. This only happens when the injected data surpasses the legitimate data.

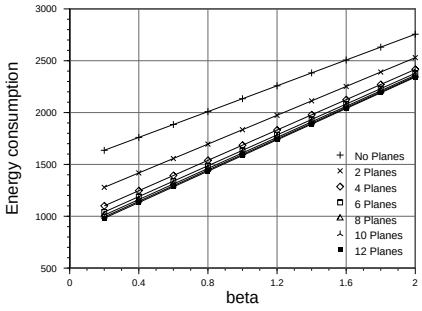

Fig. 9. Energy consumption vs. injected data ($p_1 = 0.05$).

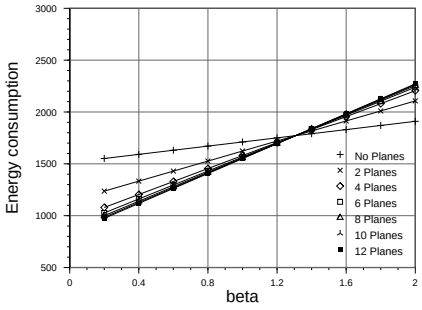

Fig. 10. Energy consumption vs. injected data ($p_1 = 0.2$).

C. Simulation Results

We use simulations to further study the performances of the MMC methods. We use 1000 nodes randomly deployed in a $800 \times 400m^2$ field. The transmission range of each node is $40m$. The sink node is located at a corner of the field. In the network, there are 200 nodes that generates data reports to be relayed to the sink. Let α portion of the reporting nodes are malicious nodes, who generates false data reports at the same rate of the legitimate nodes. We use p_1 to denote the one-hop probability that a false report gets detected. Each reporting node generates one data report every 10 seconds. Our unit of energy consumption is the energy it takes to transport one report packet over one hop. Figure 11 and 12 show the results of energy consumption versus the number of compromised nodes, and versus the one-hop detection probability, respectively.

Figure 11, shows that at 80 compromised nodes, the percentage of injected data is 40% of all network traffic. The energy consumption is significantly higher in conventional networks than the MMC networks, when the percentage of injected data is small. Also note that the difference of energy consumption when using different number of planes is not significant.

According to the SEF scheme, with each data report being attached with 5 MACs, the one-hop detection probability range from 0.05 to 0.2, based on the amount of keys captured by the opponent. The number of compromised nodes is 50 in all the cases. As shown in the figure 12, using only 4 planes

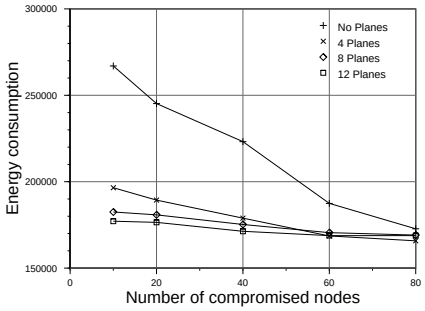

Fig. 11. Energy consumption vs. number of compromised nodes.

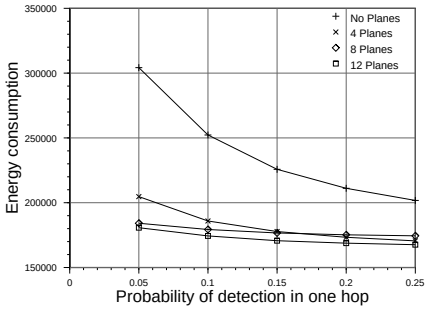

Fig. 12. Energy consumption vs. detection probability.

can significantly save energy consumption. At 0.05 detection probability, the energy savings is at 32.3% with 4 planes, and 41.9% with 12 planes.

VII. Conclusion

In this paper, we present a new Mixed Multi-Channel (MMC) architecture for sensor networks. Each node is fixed to a certain channel by its physical limitations. This partitions the network into multiple planes. The multiple independent planes are the base for the security enhancement methods that we propose in this paper. The MMC-1 method uses one plane as the secure plane, and reduces the overhead of standard secure measures while increasing its resistance level at the same time. The MMC-r further increases the resistance level of MMC-1, with comparably extra overhead. To ensure the security strength of the MMC-r scheme, we presented two schemes that implement the scheduling of the secure plane in the network. The static scheduling scheme uses the pre-stored secret information. The dynamic scheduling scheme can let the sink node determine the most suitable schedule on-line and securely deliver the updated schedule throughout the network.

Our future work involves studying heterogeneous multiplane sensor networks. We are also exploring sleep-awake duty cycling mechanisms in intra-plane and inter-plane basis in conjunction with sensor network security to improve the energy efficiency.

Appendix

Proof to Theorem 1 Consider $S_{1,t}(n)$, since there is only one plane, we can have the following:

$$S_{1,t}(n) = \begin{cases} 0 & : \quad n < t \\ 1 & : \quad n \geq t \end{cases}$$

Thus, the exponential generating function of $S_{1,t}(n)$, $G_{1,t}(u)$, can be expressed as

$$\begin{aligned} G_{1,t}(u) &= \textstyle\sum_{n=0}^{\infty} S_{1,t}(n)\frac{u^n}{n!} \\ &= \frac{u^t}{t!} + \frac{u^{t+1}}{(t+1)!} + \frac{u^{t+2}}{(t+2)!} + ... \\ &= (\textstyle\sum_{n=0}^{\infty}\frac{u^n}{n!}) - 1 - u - ... - \frac{u^{t-1}}{(t-1)!} \\ &= e^u - 1 - u - ... - \frac{u^{t-1}}{(t-1)!} \end{aligned}$$

Thus, the theorem satisfies when k=1.

Consider Plane 0, let $G^j_{k,t}(n)$ denote the number of distributions that put j nodes in Plane 0, and put at least t nodes in each of the remaining planes. We should have

$$G^j_{k,t}(n) = \binom{n}{j} S_{k-1,t}(n-j).$$

Then, $S_{k,t}(n)$ can be expressed as the following:

$$\begin{aligned} S_{k,t}(n) &= \textstyle\sum_{j=t}^{n-(k-1)t} G^j_{k,t}(n) \\ &= \textstyle\sum_{j=t}^{n-(k-1)t} \binom{n}{j} S_{k-1,t}(n-j) \\ &= \textstyle\sum_{j=t}^{n} \binom{n}{j} S_{k-1,t}(n-j) \\ &\quad (because\ S_{k-1,t}(n) = 0; n < (k-1)t) \\ &= \textstyle\sum_{j=0}^{n} \binom{n}{j} S_{1,t}(j) S_{k-1,t}(n-j) \\ &\quad (because\ S_{1,t}(n) = 0; n < t, S_{1,t}(n) = 1; n \geq t) \end{aligned}$$

Thus, by the definition of the product of two exponential generating functions, we have $G_{k,t}(u) = G_{1,t}(u) \cdot G_{k-1,t}(u)$. By induction, we have $G_{k,t}(u) = (e^u - 1 - u - ... - \frac{u^{t-1}}{(t-1)!})^k$.

References

[1] Fisher-yates shuffle. Dictionary of Algorithems and Data Structures, NIST.

[2] R. Anderson, H. Chan, and A.Perrig. Key infection: Smart trust for smart dust. In *ICNP, October 2004*, 2004.

[3] R. Blom. An optimal class of symmetric key generation systems. In *Advances in Cryptology: Proceedins of Eurocrypt 84 (LNCS vol. 209)*, 1984.

[4] C. Blundo, A. De Santis, A. Herzberg, and S. Kutten. Perfectly-secure key distribution for dynamic conferences. In *Advances in Cyptology - CRYPTO 92 (LNCS vol. 740)*, 1993.

[5] W. Du, J. Deng, Y.S. Han, and P.K. Varshney. A pairwise key predistribution scheme for wireless sensor networks. In *ACM CCS*, 2003.

[6] L. Eschenauer and V.D. Gligor. A key-management scheme for distributed sensor networks. In *ACM CCS*, 2002.

[7] K. Jamshaid and L. Schweibert. Seken (secure and efficient key exchange for sensor networks). In *IPCCC, April 2004*, 2004.

[8] C. Karlof, N. Sastry, and D. Wagner. Tinysec: A link layer security architecture for wireless sensor networks. In *ACM SenSys*, 2004.

[9] Bo-Cheng Charles Lai, David D. Hwang, Sungha Pete Kim, and Ingrid Verbauwhede. Reducing radio energy consumption of key management protocols for wireless sensor networks. In *International Symposium on Low Power Electronics and Desig*, 2004.

[10] D. Liu and P. Ning. Establishing pairwise keys in distributed sensor networks. In *ACM CCS*, 2003.

[11] J. Newsome, E. Shi, D. Song, and A. Perrig. The sybil attack in sensor networks: Analysis and defenses. In *ACM/IEEE IPSN*, 2004.

[12] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J.D. Tygar. Spins:security protocols for sensor networks. In *MobiCom, July 2001*, 2001.

[13] B. Przydatek, D. Song, and A. Perrig. Sia: Secure information aggregation in sensor networks. In *ACM SenSys*, 2003.

[14] F. Ye, H. Luo, S. Lu, and L. Zhang. Statistical en-route filtering of injected false data in sensor networks. In *IEEE Infocom*, 2004.

[15] W. Zhang and G. Cao. Group rekeying for filtering false data in sensor networks: A predistribution and local collaboration-based approach. In *IEEE Infocom*, 2005.

[16] S. Zhu, S. Setia, S. Jajodia, and P. Ning. An interleaved hop-by-hop authentication scheme for filtering false data in sensor networks. In *IEEE Symposium on Security and Privacy*, 2004.