Authentication of Sensor Network Flooding Based on Neighborhood Cooperation

Tong Zhou and Krishnendu Chakrabarty
Department of Electrical and Computer Engineering
Duke University, Durham, NC 27708, USA
Email: {tzhou, krish}@ee.duke.edu

Abstract—Flooding-based information dissemination is an important communication activity in sensor networks. Security is a major concern in sensor network flooding; in particular, the authentication of sensor nodes is important because once a node is compromised, it can send fake message to the network. One solution to this problem is to set up a base station that manages the authentication of all the sensor nodes. We propose a new authentication method that is intended for use in the absence of a trusted base station. The proposed method, referred to as Predistribution-and-Local-Collaboration-Based Merkle Tree Authentication (PCMA), combines Merkle tree authentication with neighborhood collaboration. We analyze the effectiveness of PCMA in term of the probabilities of nodes being compromised and disabled by an attacker. We also estimate the additional computation and energy required for this scheme. Finally, we present simulation results to validate our analysis.

I. INTRODUCTION

Sensor networks are becoming increasingly important for applications such as military surveillance, inventory, tracking, healthcare, and environment monitoring [1]. However, these networks are difficult to deploy in practice due to harsh operational environments, resource constraints, and dynamic network topologies.

Security is a major concern for sensor networks when privacy and data integrity are important. Compared to a traditional network, a sensor network is more vulnerable to various types of intrusions and malicious attacks. For example, the limited communication bandwidth and node memory make sensor networks vulnerable to Denial-of-Service (DoS) attacks; an adversary can simply compromise one or more sensor nodes, and then use them to obtain messages or inject false data. On the other hand, the limited processing capabilities and memory on a sensor node do not allow the implementation of many widely-used security mechanisms, such as the RSA public key for digital signatures [2], and a key distribution center for key management [3].

The authentication of flooding-based information dissemination, which we also refer to as broadcast authentication, is an important security-related problem in sensor networks. In a hostile environment, an attacker may send fake messages, such that the sensor nodes in the network are either misinformed or they make incorrect decisions. An attacker may also compromise some nodes and use them to carry out insider attacks. A number of methods have been proposed in the literature to address these problems; however, most of these methods require a base station

for network monitoring [4] [5]. If the security of sensor network depends on the base station, the entire sensor network may crash or malfunction if the base station is compromised. In our proposed scheme, the base station does not participate in data authentication; instead, each sender node is authenticated by its neighbors.

The μ -TESLA method proposed in [6] is an effective authentication scheme against sensor node spoofing. It achieves authentication by delayed key release, i.e., it broadcasts the message and its MAC first, and releases the authentication key after the message is received. However, μ -TESLA cannot prevent false data injection attacks. The interleaved hop-by-hop Authentication scheme [4] attempt to prevent compromised nodes from injecting false data; however, this scheme relies on a trusted base station.

In this paper, we propose an authentication scheme where each flooding message must be authenticated by m sensor nodes. We use a watchdog scheme [7] whereby a compromised sensor node can be detected and disabled by its one-hop neighbors. We combine the key chain used in μ -TESLA with the Merkle tree concept to achieve neighbor authentication. We show that the proposed scheme dramatically reduces the risk of false data injection in the absence of a trusted base station.

The remainder of this paper is organized as follows: Section II presents the concept of the chained Merkle tree and proposes a new authentication scheme, referred to as the Predistribution-and-Local-Collaboration-Based Merkle Tree Authentication (PCMA). Section III presents an analysis of the PCMA scheme and discusses energy consumption issues. Section IV presents simulation results derived using Matlab. Section V concludes the paper and lists future directions.

II. PREDISTRIBUTION AND LOCAL COLLABORATION BASED MERKLE TREE AUTHENTICATION

The authentication method discussed in this paper is based on two key ideas: Merkle tree authentication and neighborhood collaboration. In next subsection, we first describe the concept of a Merkle tree.

A. Merkle Tree

A Merkle tree, or a Merkle hash tree, is composed of a complete binary tree and a one-way hash function, where each node of the binary tree is the hash function of the concatenation of all its children [8]. For example, Fig. 1(a)

Fig. 1. Merkle tree and chained Merkle tree.

shows a Merkle tree with 4 leaves, where s_i are the keys, $m_i = f(s_i)$, $i = 1, \dots, 4$; $m_{1-2} = f(m_1|m_2)$, $m_{3-4} = f(m_3|m_4)$, $m_{1-4} = f(m_{1-2}|m_{3-4})$. A Merkle tree can authenticate a leaf without releasing its other leaves. For example, if we need to authenticate s_1 , we can release s_1 , m_2 , m_{3-4} and verify that $F((F(s_1)|m_2)|m_{3-4}) = m_{1-4}$.

B. Chained Merkle Tree

In order to use the Merkle tree to authenticate many broadcast packets, we introduce the concept of a chained Merkle tree by combining the Merkle tree and the key chain used in μ -TESLA. The chained Merkle tree is composed of a chain of small Merkle trees, which we call “sub-Merkle trees”. Each sub-Merkle tree has a relatively small number of leaves, e.g., 4 or 8 leaves. All the sub-Merkle tree are then concatenated as shown in Fig. 1(b). Each sub-Merkle tree is used in one authentication instance, and the authentication only involves a small cluster of sensor nodes, where the cluster size equals the number of leaves of the sub-Merkle tree. The amount of computation and communication required is thus small enough to be implemented on resource-constrained sensor nodes.

C. Predistribution and Local-Collaboration-Based Merkle Tree Authentication (PCMA)

We next introduce the Predistribution and Local-Collaboration-Based Merkle tree Authentication (PCMA) scheme. We assume that nodes have acquired information about their one-hop neighbors in the initialization stage, as in [9] [10].

Before deployment, each node stores the chained part of the chained Merkle tree. For example, for the Merkle tree shown in Fig. 1(b), the sender node n_0 stores the key chain $\{s_{0,0}, s_{0,1}, s_{0,2}, \dots\}$. It then selects 3 neighbors n_1 , n_2 and n_3 as *authentication neighbors*. The nodes n_1 , n_2 and n_3 store the neighbor authentication key chain $\{s_{1,0}, s_{1,1}, s_{1,2}, \dots\}$, $\{s_{2,0}, s_{2,1}, s_{2,2}, \dots\}$ and $\{s_{3,0}, s_{3,1}, s_{3,2}, \dots\}$, respectively. From Fig. 1(b), we see that each key in the last 3 key chains is independent of its precedent and successive keys. Therefore, in order to

Fig. 2. An example of a chained Merkle tree distributed to 3 neighbors.

include authentication in the key sequences, we make these key sequences hash key chains, i.e., we generate the key sequences $\{s_{i,0}, s_{i,1}, s_{i,2}, \dots\}_{i=1,2,3}$ using one hash function H such that $s_{i,j} = H(s_{i,j-1})$. In this case, on receiving the keys $\{s_{1,j}, s_{2,j}, s_{3,j}\}$, the three keys can be immediately authenticated by verifying that the hash function of them are the previously received three keys $\{s_{1,j-1}, s_{2,j-1}, s_{3,j-1}\}$.

During packet transmission, similar to μ -TESLA, all the nodes are loosely synchronized, and communication time is divided into time intervals $t_0, t_1, t_2, \dots$. Assume that each node has the key chain $k_{0,j}, j \in \{0, 1, 2, \dots\}$, and the node has m neighbors who hold part of the chained Merkle tree $\{k_{i,j}, i \in \{0, \dots, m-1\}, j \in \{0, 1, 2, \dots\}\}$. During each time interval (t_j, t_{j+1}) , the sender node starts with the key $k_{0,j}$ as the current authentication key and sends authentication requests $(M|MAC(M, kp_{0,1})), \dots, (M|MAC(M, kp_{0,m-1}))$, where M is a message generated with a one-way hash function. On receiving and authenticating the authentication requests, the sender's authentication neighbors send the key $(k_{1,j-1}|MAC(k_{1,j-1}, kp_{0,1})), \dots, (k_{m-1,j-1}|MAC(k_{m-1,j-1}, kp_{0,m-1}))$ to the sender node. The sender node then broadcasts the keys $k_{0,j-1}, F(k_{1,j-1}), F(F(k_{2,j-1})|F(k_{3,j-1})), \dots$. In the following sections, we refer these keys as $K_{0,j-1}, \dots, K_{\log m, j-1}$. Other nodes can authenticate $k_{0,j-1}$ using the method described in Section II.A.

Fig. 2 shows the key distribution on one sender node and its neighbors in case of a Merkle tree with four leaves. Note that although a Merkle tree must have 2^n leaves, where n is an integer, the number of authentication neighbors need not be 2^n , but it can be an arbitrary integer m . For any m , we construct the chained Merkle tree where each sub-Merkle tree has $\lceil \log_2 m \rceil + 1$ levels, and two or more leaves may be given to a single authentication neighbor node.

A watchdog mechanism is used in the sensor network to detect if a node is compromised. If a sensor node n_0 detects that one of its neighbors n_1 has been compromised, and n_0 holds the authentication key chain of n_1 , it refuses to authenticate n_1 . Fig. 3 shows a flowchart illustrating

Fig. 3. Flowchart illustrating PCMA.

PCMA.

III. PERFORMANCE ANALYSIS: ANALYTICAL RESULTS

For the purpose of analysis, we first make the following assumptions:

- 1) The sensor nodes are uniformly distributed, and each node has n neighbors. The sender nodes are distributed uniformly among all the nodes.
- 2) The sub-Merkle tree of the chained Merkle tree has m leaves, i.e., each sensor node has $m-1$ authentication neighbors.
- 3) Each node can be compromised by an attacker with the same probability p .

In the PCMA scheme, an attacker can use sender node n_0 to carry out a false data injection attack if n_0 is compromised and all the authentication neighbors of n_0 are compromised.

A node is said to be disabled if either it is unresponsive, or any of its authentication neighbor refuses to authenticate it. We next analyze the security and performance of PCMA using three criteria: the risk of a false data injection attack, the risk of nodes being disabled, and the computation and communication overhead compared to flooding without PCMA.

A. The Risk of Nodes Being Compromised and Used to Inject False Data

Here we develop a probabilistic model to characterize the risk that nodes are compromised and used to inject false data. We first define the following events:

- $\mathcal{E}_{1i}$: Sender node n_i is compromised and can be used to inject false data.
- $\mathcal{E}_{2k}$: For a sensor network with N nodes, k senders are compromised.
- $\mathcal{E}_{3k}$: Given k compromised senders, the attacker can use at least one of them to inject false data.
- $\mathcal{E}_4$: For a sensor network which contains N senders, the attacker can successfully carry out a false data injection attack.

Assuming that the events $\mathcal{E}_{1i}$, $1 \leq i \leq N$, are independent, we have $P[\mathcal{E}_{1i}] = p^m$, which implies that the node n_i is compromised and can be used to inject false data if

Fig. 4. Variation of $P[\mathcal{E}_4]$, the risk of false data injection attack, with m .

all its authentication neighbors and the node itself are compromised. For $k \geq m$, we have

$$\begin{aligned} P[\mathcal{E}_{3k}|\mathcal{E}_{2k}] &= 1 - \prod_{i=0}^{k-1} (1 - P[\mathcal{E}_{1i}]) \\ &= 1 - (1 - p^{m-1})^k \end{aligned} \quad (1)$$

The assumption that the events $\mathcal{E}_{1i}$ are independent is valid only when the compromised nodes are sparsely distributed. Therefore, we are estimating the value of $P[\mathcal{E}_4]$ in this section. Note that $P[\mathcal{E}_{3k}|\mathcal{E}_{2k}] = 0$ if $k < m$. Therefore,

$$\begin{aligned} P[\mathcal{E}_4] &= \sum_{k=m}^N P[\mathcal{E}_{2k}] P[\mathcal{E}_{3k}|\mathcal{E}_{2k}] \\ &= \sum_{k=m}^N \binom{N}{k} p^k (1-p)^{N-k} (1 - (1 - p^{m-1})^k) \\ &= \frac{B(p; m, N - m + 1)}{B(m, N - m + 1)} \\ &\quad - (1 - p^m)^N \frac{B(\frac{p-p^m}{1-p^m}; m, N - m + 1)}{B(m, N - m + 1)} \end{aligned} \quad (2)$$

where $B(a, b)$ is the well-known Beta function, and $B(x; a, b)$ is the incomplete Beta function [11]. From (2) it can be seen that if $p \rightarrow 0$, $P[\mathcal{E}_4] \rightarrow 0$. Fig. 4 shows how $P[\mathcal{E}_4]$ varies with m where all the nodes are senders. We assume a total of 10,000 nodes, where each node has 36 neighbors. These two parameters are used throughout for performance analysis in this paper. If p is fixed, increasing m can improve the tolerance to false data injection attacks. It may also be seen that, for $p < 0.5$, setting m between 8 and 16 makes $P[\mathcal{E}_4] \leq 0.1$. However, larger value of m may reduce the sensor network's availability, as shown in the next subsection.

B. The Risk of Nodes Being Disabled

Here we study the likelihood of nodes being disabled if there exist unresponsive nodes. We first consider the extreme case where each sensor node is a sender, and define the following events:

- $\mathcal{E}_{5i}$: For a compromised sensor node n_i , n_i is chosen by one specific neighbor as the its authentication neighbor.
- $\mathcal{E}_{6ij}$: For a compromised sensor node n_i , j of its neighbors are disabled.

Since each sensor node has n neighbors, and $m - 1$ of the neighbors are randomly chosen as authentication neighbors, we have $P[\mathcal{E}_{5i}] = (m - 1)/n$. Moreover, it can be easily shown that

$$\begin{aligned} P[\mathcal{E}_{6ij}] &= \binom{n}{j} (P[\mathcal{E}_{5i}])^j (1 - P[\mathcal{E}_{5i}])^{n-j} \\ &= \binom{n}{j} \frac{(m-1)^j}{n^j} \left(1 - \frac{m-1}{n}\right)^{n-j} \end{aligned} \quad (3)$$

From (3), we can easily infer that if one node is unresponsive, the average number of disabled nodes is m . The next step in the analysis is to estimate the total number of disabled nodes N_D in the sensor network. We estimate N_D in an incremental function for each unresponsive node. Let $N_D(k)$ be the number of disabled nodes due to k unresponsive nodes. For the first unresponsive node, $N_D(1) = m$. Now we add another unresponsive node, and it disables another cluster of nodes. Note that this time, the newly disabled nodes may overlap with the previously disabled nodes, and the number of disabled nodes equals the number contributed by two clusters of nodes minus the overlap. Therefore,

$$\begin{aligned} N_D(2) &= m + m - m \frac{m}{N} \\ &= m + m \frac{N-m}{N} \end{aligned} \quad (4)$$

As more nodes are rendered unresponsive, the number of disabled nodes increases at a slower rate.

Theorem 1: The expected number of disabled nodes $N_D(k)$ for k unresponsive nodes is given by:

$$N_D(k) = \sum_{i=0}^{k-1} m \left(\frac{N-m}{N} \right)^i.$$

Proof: We use the principle of mathematical induction to prove this theorem. The statement trivially holds for $k = 1$. Assume that for $k - 1$ unresponsive nodes,

$$N_D(k-1) = \sum_{i=0}^{k-1} m \left(\frac{N-m}{N} \right)^i$$

We show that the statement also holds for k unresponsive nodes. As we add one more unresponsive node,

$$\begin{aligned} N_D(k) &= \sum_{i=0}^{k-1} m \left(\frac{N-m}{N} \right)^i + m - m \frac{\sum_{i=0}^{k-1} m \left(\frac{N-m}{N} \right)^i}{N} \\ &= \sum_{i=0}^k m \left(\frac{N-m}{N} \right)^i. \end{aligned} \quad (5)$$

This completes the proof of the theorem. $\blacksquare$

Since the expected number of compromised nodes is Np , we conclude that

$$\begin{aligned} N_D(Np) &= \sum_{i=0}^{Np-1} m \left(\frac{N-m}{N} \right)^i \\ &= N \left(1 - \left(\frac{N-m}{N} \right)^{Np} \right). \end{aligned} \quad (6)$$

Equation (6) provides an estimate because while we are considering the overlap between the neighbors of the disabled nodes, we do not consider the overlap between the original unresponsive nodes. If $N \rightarrow \infty$, $N_D(Np) \rightarrow 1 - e^{-pm}$.

Another extreme case is that the sender nodes are sparsely distributed, such that none of the two senders share an authentication neighbor. In this case, a sender is disabled when it itself is unresponsive or one of its authentication neighbors is unresponsive. Therefore, we define the following events:

- $\mathcal{E}_{7i}$: Sender node n_i is disabled.

Obviously, $P[\mathcal{E}_{7i}] = 1 - (1-p)^m$. Since we assume that the senders are disabled in an independent fashion, $N_D(Np) = N(1 - (1-p)^m)$. We further conclude that the two extreme cases provide upper and lower bounds on the number of disabled senders, and it can be seen easily that both of these bounds are exponential in m .

C. Amount of Extra Computation and Communication

Comparing to the simple flooding method, PCMA requires additional computation and communication. The energy consumption due to the additional computation and communication are evaluated in this section. For the convenience of calculation, we first define the following parameters:

- The size of each message is S_M bits, the size of the key is S_K bits, and the size of the MAC is S_{MAC} bits.
- Each authentication requires n_a cycles, and each hash function in the Merkle tree uses n_h cycles.
- For the sensor node, the clock cycle is F_p (in Hz), the data rate is d bps, the power rate for transmission, reception, and computation are p_t , p_r and p_c (in W), respectively.

1) *Computation necessary for PCMA:* In order to implement PCMA, the following computations are necessary:

- 1) Sender calculates the MAC of each message, i.e., there is one MAC calculation per message.
- 2) During key release at the beginning of a time interval, each authentication neighbor sends out the authentication key to the sender, and the sender needs to verify that all the keys are correct and sends the $\lceil \log m \rceil + 1$ keys. Therefore, the sender needs $2m$ hash function calculations.
- 3) On receiving the released key, each receiver needs to authenticate the buffered messages. Therefore, one MAC calculation is needed per message.
- 4) After receiving the disclosed keys $K_{1,j}, K_{2,j}, \dots, K_{\log m+1,j}$, each receiver node authenticates the disclosed authentication key by verifying that $K_{\log m,j-1} = F(K_{1,j}, F(K_{2,j}, \dots, F(K_{\log m,j}, F(K_{\log m+1,j}))) \dots)$. Therefore, each receiver node needs to carry out a total of $\lceil \log m \rceil + 1$ hash function calculations.

Besides, each node has to carry out one authentication calculation for each packet it receives. Therefore, assuming that there are m_p packets to be flooded per time interval, the overhead for one packet in cycle is

$$n_a + 2mn_h/(Nm_p) + n_h(\lceil \log m \rceil + 1)/m_p.$$

Thus the energy consumption in Joules for PCMA can be estimated to be

$$\frac{p_c}{F_p} \left(n_a + \frac{2mn_h}{Nm_p} + \frac{n_h(\lceil \log m \rceil + 1)}{m_p} \right) \text{ J}$$

2) Amount of Communication Required for PCMA:

Comparing to flooding without authentication, the following additional communication is required in the PCMA scheme:

- 1) For each packet sent, the MAC is sent together with the packet. This amounts to S_{MAC} more bits for each packet.
- 2) During packet authentication in each time interval:
 - a) The sender sends $m - 1$ authentication requests, i.e., $(S_M + S_{MAC})(m - 1)$ extra bits.
 - b) On receiving the authentication requests, the receiver sends the authentication key, i.e. $(S_K + S_{MAC})(m - 1)$ bits.
 - c) The sender then floods the $\lceil \log m \rceil + 1$ keys over the sensor network, i.e., a total of $(\lceil \log m \rceil + 1)S_K$ bits are flooded.

Therefore, the extra amount of communication for each flooded packet and each sensor node is

$$S_{MAC} + (m - 1)(S_M + S_K - 2S_{MAC})/(Nm_p) + (\lceil \log m \rceil + 1)S_K/m_p \text{ bits.}$$

The energy cost of the communication per packet for each node is

$$(p_t + p_r) (S_{MAC} + (m - 1)(S_M + S_K - 2S_{MAC})/(Nm_p) + (\lceil \log m \rceil + 1)S_K/m_p) / d \text{ J.}$$

The energy cost in Joules of broadcast without authentication per node per broadcast packet is estimated as

$$\frac{p_t + p_r}{d} S_M \text{ J.}$$

3) *Energy Efficiency:* From the above analysis, we can calculate the percentage energy overhead for PCMA. Assume that we are using MICA2 sensor nodes [12], and MD5 (64 calculations, each involves 3 and/or xor calculations of 128 bits) is used as both the Merkle hash function and authentication function. Each message is 512 bits long, and each key/MAC is 128 bits. The parameters of the MICA2 nodes are shown in Table I.

We estimate the total energy consumption and the energy consumption overhead for $m = 1, \dots, 16$, and $m_p = 8$. The results are shown in Fig. 5. The contribution of PCMA to the total energy consumption is approximately 30%. Since as much as 20% of the total energy is spent on the MAC transmission [6], which cannot be avoided in any authentication scheme, the additional overhead due to

TABLE I
SOME PARAMETERS OF THE MICA2 NODE [12].

Chip type	7.37 MHz, 8 bit
max data rate (kbits/sec)	38.4
full operation current (mA)	12
sleep current (mA)	0.010
radio receive current (mA)	7
radio transmit current (mA)	10
flash memory write current (mA)	15
flash memory read current (mA)	4

Fig. 5. Energy overhead due to PCMA.

PCMA is quite low.

IV. SIMULATION RESULTS

In this section, we present simulation results using Matlab. We make the following assumptions:

- 1) There are a total of 10,000 nodes.
- 2) The nodes are evenly distributed, and each node has 36 neighbors – this number is chosen to ensure that there are enough neighbors available to serve as authentication neighbors.

In order to calculate the risk of false data injection attack, we simulated random node compromise 100 times, and obtain the number of successful compromise instances¹. During each random compromise simulation, a biased coin with heads up probability p is tossed for each node to decide whether the node is compromised. We consider the two cases corresponding to all nodes being senders, and a total of 626 uniformly distributed senders. The comparison between analytical and simulation results are shown in Fig. 6 and Fig. 7. It can be seen that the simulation results and the analytical results match each other very well.

We also calculated the number of disabled nodes using the same simulation setup. For each random compromise, we counted the total number of disabled nodes, and finally used the average number as the number of disabled nodes. The result is shown in Fig. 8. The simulation results and analytical results of ratio of disabled nodes also match very well.

From the analytical and simulation results, we conclude that:

¹Success in this context is defined as the scenario where an attacker can compromise a node and all its authentication neighbors, thereby injecting false data.

Fig. 6. All nodes are senders: $p = 0.1$

Fig. 7. 626 uniformly distributed senders: $p = 0.1$

Fig. 8. Analytical and simulation results: $p = 0.1$

- 1) In order to prevent false data injection attacks, m should be considerably large.
- 2) In order to prevent too many nodes being disabled due to a small number of unresponsive nodes, $m > 10$ is not acceptable.

We therefore need a trade-off between the two conflicting objectives. We define a weighted cost function T as

$$T = \alpha P[\mathcal{E}_4] + (1 - \alpha) \frac{N_D}{N} \quad (7)$$

and try to find the value of m that minimizes T , where $0 \leq \alpha \leq 1$. The variation of the cost function T with m is shown for $\alpha = 0.1$ and $\alpha = 0.5$ in Figures 9-10. The results show that the best value of m is 5 for $p = 0.1$ and 10 for $p = 0.3$ ($\alpha = 0.5$ in both cases).

V. CONCLUSION AND FUTURE WORK

We have proposed a broadcast authentication scheme that prevents false data injection attacks without base-station monitoring. This method is based on a combination of Merkle tree authentication and neighborhood collaboration. The only requirement for this scheme is that the sender nodes must have sufficient memory to hold a key chain. This may be achieved by adding a small amount of external flash memory to the sender nodes. We have presented theoretical analysis as well as simulation results to evaluate this method in terms of probabilities of false data injections and the disabling of nodes. We have also shown that compared to

Fig. 9. Trade-off: $\alpha = 0.1$

Fig. 10. Trade-off: $\alpha = 0.5$

other authentication methods such as μ -TESLA, the energy overhead for the proposed scheme is quite low.

Future work includes the use of cascaded PCMA to prevent a “smart” attack whereby an attacker attempts to compromise one sender node and all of its neighbors, the addition of redundant authentication neighbors to improve robustness, and the implementation of PCMA on TinyOS.

REFERENCES

- [1] C.-Y. Chong and S. P. Kumar, “Sensor networks: Evolution, opportunities and challenges,” *Proceedings of IEEE*, vol. 91, no. 8, pp. 1247–1256, 2003.
- [2] R. L. Rivest, A. Shamir, and L. M. Adelman, “A method for obtaining digital signatures and public-key cryptography, Tech. Rep. MIT/LCS/TM-82, 1977.
- [3] J. Kohl and C. Neuman, “The kerberos network authentication service (v5), Tech. Rep. IEEE RFC 1510, 1993.
- [4] S. Zhu, S. Setia, S. Jajodia, and P. Ning, “An interleaved hop-by-hop authentication scheme for filtering of injected false data in sensor networks,” in *IEEE Symposium on Security and Privacy*, May 2004, pp. 259–271.
- [5] M. Bohge and W. Trappe, “An authentication framework for hierarchical ad hoc sensor networks,” in *ACM Proceeding of Fourth International Conference on Web Information Systems Engineering*, 2003, pp. 79–87.
- [6] A. Perrig, R. Szewczyk, V. Wen, D. E. Culler, and J. D. Tygar, “SPINS: security protocols for sensor networks,” in *ACM Mobile Computing and Networking*, 2001, pp. 189–199.
- [7] S. Marti, T. Giuli, K. Lai, and M. Baker, “Mitigating routing misbehavior in mobile ad hoc networks,” in *Proc. 6th Ann. Intl. Conf. Mobile Computing and Networking (Mobicom 2000)*. New York: ACM Press, 2000, pp. 255 – 265.
- [8] A. Perrig, *Secure Broadcast Communication in Wired and Wireless Networks*. Boston: Kluwer Academic Publishers, 2003.
- [9] L. Eschenauer and V. D. Gligor, “A key-management scheme for distributed sensor networks,” in *Proceeding of ACM Conference on Computer and Communications Security*, 2002, pp. 41–47.
- [10] D. Liu and P. Ning, “Establishing pairwise keys in distributed sensor networks,” in *Proceedings of the 10th ACM conference on Computer and Communications Security*, 2003, pp. 52–61.
- [11] “mathworld.wolfram.com,” website.
- [12] “www.xbow.com,” website.