Multi-path Key Establishment under Byzantine Attacks in Wireless Ad Hoc Networks

Tian Lan, Ruby Lee, and Mung Chiang
Department of Electrical Engineering, Princeton University, NJ 08544, USA
{tlan, rblee, chiangm}@princeton.edu

Abstract— **Secure communications in wireless ad hoc networks are protected by setting up end-to-end secret keys for communicating node pairs. Secret keys can be provided by pre-loading shared secrets (e.g. a set of potential keys or keying information) into nodes prior to deployment. However, due to physical limitations of nodes and network scalability requirements, this key pre-distribution approach is not able to achieve full key-connectivity for all communicating pairs. Therefore, on-demand key establishment, in which pairwise secret keys are derived by exchanging multiple keying messages among wireless nodes, becomes necessary.**

In this paper, we develop a theoretical framework for the on-demand key establishment approach. Our contributions include a novel security metric, which we call a Byzantine resilience vector, to quantify the resilience of any key establishment scheme against Byzantine (arbitrary) attacks. Our analysis shows that previous key establishment schemes are vulnerable under Byzantine attacks. We prove a universal bound on achievable Byzantine resilience vectors for any on-demand key establishment scheme. We show that this bound, which characterizes the optimal security analytically, is tight, by proposing a Byzantine-resilient key establishment scheme which achieves any vector within this bound. In addition, we also propose a class of low complexity key establishment schemes which achieves nearly-optimal Byzantine attack resilience. The security and complexity of the proposed schemes are analyzed.

I. Introduction

In wireless ad hoc networks such as wireless sensor networks, symmetric key cryptography is attractive due to its efficiency under extreme node resource constraints (e.g. computation, memory and power). Currently, there exist three different approaches for providing pairwise secret keys: key assignment using trusted third parties, key pre-distribution before initial node deployment, and key establishment by exchanging keying messages. In particular, the key assignment schemes rely on trusted servers for key agreement among nodes [1], [2]. These schemes may not be practical for ad-hoc networks or large-scale sensor networks, which do not have the infrastructure of trusted servers or base-stations. The second approach, key pre-distribution, has attracted a lot of attention recently due to its efficiency in small or local networks. In key pre-distribution schemes, a large amount of secret keys or keying information can be preloaded into nodes prior to deployment. Neighboring nodes then discover shared keys after deployment to achieve a certain level of key-connectivity probability. The pioneering work on key pre-distribution was by Eschenauer and Gligor in [3]. The work has since been generalized by [4] to a q-composite scheme utilizing the existence of multiple shared keys and by [5] to a random matrix based scheme to improve resilience against node captures. In a separate work, a location-aware key pre-distribution scheme was proposed in [6], [7], [8]. This scheme puts strong requirements on deployment location knowledge, but achieves better scalability and local key-connectivity compared to basic key pre-distribution schemes. Later, a general framework for key pre-distribution was presented in [9].

As pointed out in [10], [11], [12], key pre-distribution schemes have to struggle with the conflicts among node resource constraints, desired key-connectivity probability, scalability in network size, and resilience against malicious attacks. Due to the limitation of node memory and computation ability, key pre-distribution schemes scale poorly to very large networks and the resulting pairwise key-connectivity probability is relatively low. In addition, most key pre-distribution schemes are designated to protect only the confidentiality of secret keys, while two other security components, integrity and availability, are not accounted for. Key pre-distribution schemes are vulnerable when various attacks occur simultaneously.

To address these issues, a key establishment approach that employs pre-distributed keys as local link keys has been proposed in [4], [13], [14], [15]. In this approach, to set up an end-to-end secret key between two nodes, the source node generates a set of keying messages, from which a secret key can be derived. Each keying message is sent through a separate communication path from the source node to the destination node which then computes the secret key locally. The transmission is protected by existing link keys at each hop. Since it is difficult to attack a large fraction of keying messages simultaneously in an ad hoc network, the key establishment approach using multi-path is able to guard against various attacks efficiently. In particular, an XOR-based key establishment scheme was proposed in [4], [13], where a secret key is derived by the XOR of all keying messages. This scheme prevents malicious attackers from deriving the secret key if not all keying messages are revealed. In [14], Shamir showed that there exists a scheme to divide a secret key into m keying messages in such a way that the key is easily reconstructable from any $v+1$ pieces, but even complete knowledge of $v-1$ pieces reveals no information about the key. This technique enables the construction of a key establishment scheme that can guard both revealing and erasure of keying messages. In a separate work [15], Huang et al proposed a Reed-Solomon

code based scheme that allows node pairs to derive secret keys when both erasure and modification of keying messages occur.

However, all of these previous key establishment schemes only deal with a subset of the following three attacks and thus, are vulnerable to a Byzantine attack model, in which malicious nodes (i.e. compromised or attacker-fabricated nodes) can (a) reveal the keying messages passing through them to make secret keys computable to the attackers; (b) erase and stop forwarding keying messages to prevent other nodes from establishing secret keys; or (c) cheat the receivers by modifying the forwarded keying messages to prevent other nodes from deriving the correct secret keys. The main contributions of this paper are summarized as follows:

- We define a novel security metric, called a Byzantine resilience vector, to quantify the resilience of any key establishment scheme against Byzantine attacks. The security of previous key establishment schemes [4], [13], [14], [15] is evaluated with respect to the proposed security metric. Our analysis and simulation show that previous key establishment schemes are vulnerable under Byzantine attacks.
- We develop a unifying theoretical framework, which includes all previous key establishment schemes as special cases. The entire set of Byzantine resilience vectors achievable by any key establishment scheme is characterized by proving a security performance bound in a closed-form expression. The bound is tight, since we propose an optimal key establishment algorithm that is able to achieve any Byzantine resilience vector within the bound. This proposed algorithm is the first that can simultaneously guard against all three attacks defined in the Byzantine attack model.
- The 3-dimensional region consisting of all feasible Byzantine resilience vectors (as plotted in Fig.2 in Section III), not only provides a quantitative measurement for the optimal security of the key establishment approach but also gives a benchmark for the design and analysis of key establishment protocols given statistical attack patterns. In particular, it is proven that a secret key can always be established securely if less than one third of keying messages are attacked.
- In addition, we propose a class of low complexity key establishment algorithms with nearly-optimal Byzantine attack resilience. The algorithms only require XOR of keying messages and simple table lookups. Complexity and security performance analysis of the proposed algorithms is presented in detail. We also implement the two proposed algorithms with the Zone Routing Protocol [16] in a wireless ad hoc network and compare their performance with previous key establishment schemes. A significant security improvement is observed in simulations.

II. A New Security Metric for Byzantine Attacks

We consider a wireless ad hoc network consisting of N nodes without using any infrastructure such as access points or base stations. Secret and reliable communications in the network are protected by pairwise secret keys. In our threat model, nodes are not tamper resistant. Compromised or fabricated nodes reveal all their forwarding keying messages to attackers and also try to disrupt normal key establishment in the network.

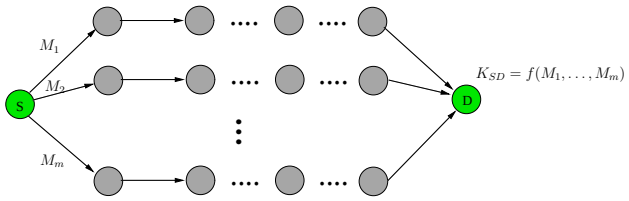

Fig. 1. This figure illustrates how to set up an end-to-end secret by sending m messages from the source node S to the destination node D, in the key establishment approach.

In the key establishment approach, multiple keying messages and communication paths between two nodes are used to set up an end-to-end secret key. Suppose that two nodes S and D in Fig 1 want to set up a pairwise key. The source node S generates m keying messages, denoted by $M_1, \ldots, M_m$, and sends each of the keying messages via a separate communication path toward the destination node D. To secure keying messages during transmission, encryptions by existing link keys are performed at each intermediate node before forwarding keying messages, and nodes at next hop decrypt the messages by the same link keys. In this approach, the link keys are provided by key pre-distribution schemes. If $S \rightarrow R \rightarrow D$ is a communication path used for sending message M_i, the following steps are performed:

$$S - E_{K_{SR}}(M_i) \rightarrow R \quad \text{and} \quad R - E_{K_{RD}}(M_i) \rightarrow D,$$

where K_{SR} and K_{RD} are existing link keys shared by node pairs S-R and R-D, respectively. Upon receiving the keying messages, node D employs a given function $f(\cdot)$ to derive an end-to-end secret key $K_{SD} = f(M_1, \ldots, M_m)$ used to secure future communication with node S. Since end-to-end secret keys can be set up based on demand, the key establishment approach allows rekeying or key refreshing to be easily implemented in wireless ad hoc networks.

Existing key establishment schemes [4], [13], [14], [15] are limited to dealing with a subset of possible attacks and become vulnerable under Byzantine attacks. Our Byzantine attack model consists of a combination of the following three independent attacks, each targeted at a security component:

- *Revealing attacks on keying message confidentiality*: Compromised or fabricated nodes reveal to attackers the content of keying messages traveling through them. To quantify the resilience against this attack, we define a threshold value $v \geq 0$, such that if no more than v messages are revealed to attackers, the end-to-end secret key remains *unconditionally secret* even if all attackers collude, i.e. when keying messages are random, we have

$$\begin{aligned} &\text{Prob}\left\{f(M_1, \ldots, M_m) = K_{SD} \middle| M_{i_1}, \ldots, M_{i_v}\right\} \\ &= \text{Prob}\left\{f(M_1, \ldots, M_m) = K_{SD}\right\} = \frac{1}{2^k}. \end{aligned} \quad (1)$$

for any $i_1, \ldots, i_v \in \mathbb{Z}_m$ and any k-bit key, K_{SD}. This implies that knowing any set of no more than v keying messages reveals zero information to attackers.

- *Erasure attacks on keying message availability*: In an attempt to prevent the end-to-end secret key from being established, compromised or fabricated nodes make keying messages unavailable to the destination, by stop-forwarding keying messages or jamming forwarding link. We define $e \geq 0$ to be a threshold such that the secret key can be recovered at the destination node if no more than e messages are erased.
- *Modification attacks on keying message integrity*: Since complicated authentication methods (e.g. digital signature using public-key-based cryptography) are impractical in ad hoc networks, keying messages are subject to modification attacks, in which compromised or fabricated nodes forward modified keying messages to cause confusion. A threshold value $d \geq 0$ is chosen to denote the maximum number of modified messages that can be corrected by a key establishment scheme.

Although erasure and modification attacks can also be regarded as transmission erasures and errors from a classical error control coding perspective, where much research has been done on how to correct transmission erasures and errors efficiently, our Byzantine attack model in this paper is different, because providing confidentiality (which is irrelevant to error control coding applications) is a must for establishing end-to-end secret keys in wireless ad hoc networks. In the following, we will provide a unifying framework and analysis for resilience of any key establishment schemes under Byzantine attacks.

Byzantine attack is defined as any arbitrary combination of the revealing, erasure, and modification attacks. Given that m keying messages are used for establishing a secret key in a key establishment scheme, we quantify its Byzantine attack resilience by a three tuple $(v, e, d)_m$, denoted as a Byzantine resilience vector. The vector $(v, e, d)_m$ is a new metric that measures the security performance of any key establishment scheme under Byzantine attacks. More precisely, we say that a key establishment scheme achieves a Byzantine resilience $(v, e, d)_m$, if an end-to-end secret key can be successfully established under no more than e erasure attacks and d modification attacks, while attackers have absolutely no information about the key even if they obtain the content of v keying messages and collude. For a key establishment scheme using m keying messages, the set of achievable Byzantine resilience vectors lies in a 3-dimensional region, which illustrates security of the particular scheme along three axis, confidentiality, availability, and integrity (see Figure 2).

We can analyze the security of previous key establishment schemes within our new unifying framework. In [4], [13], secret keys of length k are derived at destination nodes by the bitwise XOR of all keying messages, each being exactly k bits, i.e. $K_{SD} = M_1 \oplus \ldots \oplus M_m$. It can be verified that the end-to-end secret key remains confidential if not all keying messages are revealed to attacks. Thus, this scheme achieves Byzantine resilience $(v = m-1, e = 0, d = 0)_m$. In [14], the author proposed another key establishment scheme, in which a secret key is regarded as an integer coefficient of a degree t random polynomial in GF_{2^k}, such that it can be recovered from any $t+1$ evaluations of the polynomial and remains undetermined if only t evaluations are known. By varying the degree t and assigning $v = t$, this scheme is able to achieve $(v+e = m-1, d = 0)_m$. The last scheme [15] employs Reed-Solomon codes (a special class of error control codes) to deal with keying message erasure and modification. Considering the secret key as an input, keying messages are constructed by dividing the output codeword into m pieces, such that the key can be recovered if no more than e and d keying messages are erased and modified respectively, given $2d + e \leq m - 1$. Extending this scheme to general error control codes achieves a Byzantine resilience of $(v = 0, e + 2d = m - 1)_m$, since any revealed keying message can be used as a constraint to reduce the search space of the secret key and thus violates unconditional secrecy. Table I summarizes the security analysis of previous key establishment schemes, whose vulnerabilities under Byzantine attacks (i.e. entries with zero resilience) are marked by * in the table.

Previous Schemes	Resilience vector $(v, e, d)_m$		
	v	e	d
XOR [4], [13]	$v = m - 1$	$e = 0^*$	$d = 0^*$
Polynomial [14]	$v + e = m - 1$		$d = 0^*$
RS code [15]	$v = 0^*$	$2d + e = m - 1$	

TABLE I

THE SECURITY ANALYSIS IN THIS TABLE SHOWS THAT PREVIOUS KEY ESTABLISHMENT SCHEMES ARE VULNERABLE UNDER BYZANTINE ATTACKS, SINCE THEY ARE DESIGNED TO DEAL WITH ONLY A SUBSET OF POSSIBLE ATTACKS.

III. PROVING THE OPTIMAL BYZANTINE RESILIENCE

In this section, we prove a universal bound on achievable Byzantine resilience vectors for key establishment. This bound is shown to be tight, as we propose an optimal key establishment scheme which can achieve any vector within this bound. To our knowledge, our proposed algorithm is the first that provides a solution against all three attacks defined in the Byzantine attack model simultaneously.

Theorem 1: With the use of m keying messages, each of k bits, a Byzantine resilience vector $(v, e, d)_m$ can be achieved if and only if $v + e + 2d \leq m - 1$.

Proof: The theorem states that the bound $v + e + 2d = m - 1$ is both optimal and tight. In the following, we start by showing the optimality and then propose a new key establishment scheme to prove the achievability.

To show $v + e + 2d = m - 1$ is optimal. If $e = d = 0$, then we have $v \leq m - 1$, since the secret key becomes deterministic given all m keying messages. The upper bound of $v+e+2d = m-1$ in this case is trivial. For $e+d > 0$, we denote $[M_1, \ldots, M_m]$ as a *feasible message vector*, in which $M_1, \ldots, M_m$ are a set of allowable keying messages that can

be used to establish a secret key $K_{SD} = f(M_1, \ldots, M_m)$. Now, without loss of generality, we assume that the first v keying messages $M_1, \ldots, M_v$ are revealed to attackers who are able to collude. Then, with this information, the attackers can rule out any feasible message vector whose first v keying messages are not equal to $M_1, \ldots, M_v$. To guarantee unconditional secrecy of the secret key, all possible keys must be feasible. It is necessary that the number of remaining feasible message vectors with the first v messages in common must be no less than 2^k, i.e. the number of all possible secret keys of length k. Formally, if $H(\cdot)$ denotes the entropy function and feasible message vectors are random, we derive

$$\begin{aligned} & H([M_1, \ldots, M_m] | M_1, \ldots, M_v) \\ & \geq H(f(M_1, \ldots, M_m) | M_1, \ldots, M_v) \\ & = H(K_{SD} | M_1, \ldots, M_v) \\ & = H(K_{SD}) \\ & = k \end{aligned} \tag{2}$$

where K_{SD} is the secret key. The second step is from the information processing inequality and the fourth step holds because all keys are equally likely due to the definition of unconditional secrecy (1). Equation (2) implies that with the first v messages fixed, there exists at least 2^k feasible message vectors. These 2^k feasible message vectors are different only in the last $m-v$ messages, each of length k. Thus, the minimum Hamming distance of these feasible message vectors (i.e. the minimum number of different messages in any two feasible message vectors) can be no more than $m - v$. According to error control coding theory, given e erasures and d modifications, two feasible message vectors with a Hamming distance of $m - v$ remain distinct and separable only if

$$2d + e + 1 \leq m - v \quad \Leftrightarrow \quad v + e + 2d \leq m - 1 \tag{3}$$

This gives the optimality of bound $v + e + 2d \leq m - 1$.

For achievability of the bound, we propose a new key establishment scheme that achieves any Byzantine resilience vector $(v, e, d)_m$ satisfying the upper bound $v+e+2d+1 = m$. The proposed algorithm for generating m keying messages is similar to the polynomial evaluation used in [14]. However, we employ a different decoding strategy and show that the algorithm can deal with revealing, erasure, and modification attacks at the same time. Let $p > 2^k$ be a prime number. Thus the desired secret key can be regarded as an integer in the field GF_p, i.e. $K_{SD} \in [0, 2^k - 1]$. We generate a random degree v polynomial in GF_p as follows:

$$q(z) = K_{SD} + A_1 z + \ldots + A_v z^v \tag{4}$$

where $A_i \in GF_p$ for $i = 1, \ldots, v$ are randomly chosen integers. Then m keying messages are computed at the source node by evaluating $q(x)$ at m distinct points for $z = 1, \ldots, m$, i.e.

$$[M_1, M_2, \ldots, M_m] = [q(1), q(2), \ldots, q(m)] \tag{5}$$

Since the polynomial has degree v, it has been shown in [14] that revealing no more than v keying messages would leave the secret key K_{SD} completely unknown. So we only need to show that the destination node can recover key K_{SD} with e erasures and d errors, given that $2d+e = m-v-1$. Toward this end, we re-write equation (5) using a matrix representation:

$$\begin{bmatrix} M_1 \\ M_2 \\ \vdots \\ M_m \end{bmatrix} = \begin{bmatrix} 1 & 1^1 & 1^2 & \ldots & 1^v \\ 1 & 2^1 & 2^2 & \ldots & 2^v \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & m^1 & m^2 & \ldots & m^v \end{bmatrix} \cdot \begin{bmatrix} K_{SD} \\ A_1 \\ \vdots \\ A_v \end{bmatrix}$$

It is easy to verify that the $m \times (v+1)$ coefficient matrix (denoted by G) on the right hand side is a Vandermonde matrix, whose any $v+1$ rows are full rank because

$$\begin{vmatrix} 1 & i_1^1 & i_1^2 & \ldots & i_1^v \\ 1 & i_2^1 & i_2^2 & \ldots & i_2^v \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & i_{v+1}^1 & i_{v+1}^2 & \ldots & i_{v+1}^v \end{vmatrix} = \prod_{j \neq l} (i_j - i_l) \neq 0$$

where $i_1, \ldots, i_v + 1$ are the index of any $v+1$ rows of matrix G. Thus, any non-zero vector $\vec{x}$ in $GF_p^{(v+1)}$ of size $1 \times (v+1)$ can be orthogonal to at most v rows of matrix G. We have

$$\forall \vec{x} \neq \vec{0}, \ \mathrm{Hamming}(G\vec{x}, \vec{0}) \geq m - v \tag{6}$$

where $\vec{0}$ is a zero vector and $\mathrm{Hamming}(\cdot)$ is the Hamming distance function. This implies that matrix G is a generating matrix for a $(m, v+1, s)$ linear error control code in GF_p with a minimum Hamming distance of at least $m - v$. According to error control coding theory, given that $2d + e + 1 \leq m - v$, any d modifications and e erasures of the keying messages can be corrected at the destination node using a sphere decoding algorithm which finds the closest feasible message vector to the received one [17]. We summarize the optimal key establishment algorithm as follows:

Algorithm 1: *Optimal Key Establishment Algorithm*

1) Source node generates a random key K_{SD} and v random integers $A_1, \ldots, A_v$.
2) Initialize $i = 1$.
3) Source node generates $\mathbb{M}_i = K_{SD} + A_1 i + \ldots + A_v i^v$ and sends it to destination node.
4) If $i < m$, let $i = i + 1$ and go to step 3.
5) Destination node employs sphere decoding to derive K_{SD} upon receiving the keying messages.

This complete the proof of Theorem 1. ■

Remark 1: When the length of keying messages is less than that of the secret key (i.e. length$(M_i) < k$), it can be proven that a Byzantine resilience $(v, e, d)_m$ can be achieved if and only if $v + e + 2d \leq m - \lceil \frac{k}{\text{length}(M_i)} \rceil$. This is a more general result than Theorem 1 that applies to all key establishment algorithms. Its proof is omitted due to space limitation.

Theorem 1 shows that for a given $m > 1$, the set of achievable Byzantine resilience vectors $(v, e, d)_m$ form a 3-dimensional tetrahedron $2d+e+v \leq m-1$ as shown in Fig 2, while previous key establishment schemes only achieve certain

2-dimensional sub-planes in the tetrahedron: the polynomial based approach in [14] achieves $\{v+e+1 \leq m, d=0\}$, the Reed-Solomon code based approach in [15] achieves $\{v = 0, e+2d+1 \leq m\}$, and the XOR based approach in [4] only achieves a single line $\{v \leq m-1, e=0, d=0\}$. Our framework for key establishment includes all previous results as lower-dimensional special cases.

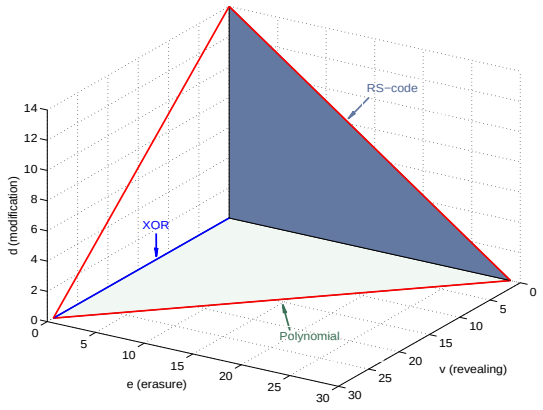

Fig. 2. For $m=30$, this figure plots the 3-D optimal Byzantine resilience region (i.e. the tetrahedron defined by $2d+e+v \leq m-1$) and 2-D sub-planes achieved by previous schemes.

IV. Applications in Wireless Ad Hoc Networks

The Byzantine resilience vector and its optimal bound proved in Theorem 1 provide a fundamental benchmark, from which many important security performance metrics can be derived directly, for given statistical attack patterns. In this section, we show two examples to illustrate how to apply our results to the design and analysis of key establishment schemes in wireless ad hoc networks. The first example focuses on maximizing the resilience to malicious paths, while the second example optimizes the secure key establishment probability.

A. Maximum Malicious Path Resilience

In ad hoc networks, a message forwarding path is malicious if it contains at least one compromised or fabricated node. As the simplest case, if there is exactly one attack on each malicious path and the number of each type of attack is equal (i.e. $v=e=d$), then from $2d+e+v \leq m-1$, it is easy to verify that an end-to-end key can be established securely if less than three quarters of paths are malicious (i.e. $v+e+d \leq \lfloor \frac{3(m-1)}{4} \rfloor$). However, the assumption of equal numbers of attacks is impractical, because malicious nodes can collude to perform multiple attacks on a single path and cause maximum damage to the secret key establishment. For such smart Byzantine attacks, we derive the maximum malicious path resilience as follows.

Corollary 1: Under smart Byzantine attacks, an end-to-end secret key can be established securely if and only if less than one third of paths (i.e. $\lfloor \frac{m-1}{3} \rfloor$) are malicious.

Proof: Suppose that x out of m paths are malicious. We need to find the maximum allowable x such that a secret key can be established under smart Byzantine attacks. Since each malicious path contains at least one attack, we obtain $x \leq v+e+d$. Clearly, a erasure attack and a modification attack can not exist on a single path, while each of them can co-exist with a revealing attack. This implies two more feasibility constraints $x \geq v$ and $x \geq d+e$. Thus, we can formulate a max-min optimization problem, where a network operator tries to maximize path resilience x under the feasibility constraints and colluding malicious nodes minimize path resilience x over all possible Byzantine attacks, given that the vector (v,e,d) satisfies $2d+v+e=m-1$. Formally, we have

$$\begin{aligned} \max_{x} \min_{v,e,d} \quad & x \\ \text{subject to} \quad & x \geq e+d \\ & x \geq v \\ & x \leq v+e+d \\ & v+2d+e=m-1 \\ & x,v,e,d \geq 0 \end{aligned} \tag{7}$$

Solving this linear optimization problem, we derive $x_{\text{opt}} = \lfloor \frac{m-1}{3} \rfloor$, which completes the proof. ■

B. Maximum Secure Key Establishment Probability

For known attack statistics, we compute the secure key establishment probability for any given scheme and then maximize the probability over the set of achievable Byzantine resilience vectors. In practice, when keying messages are forwarded on non-deterministic or random communication paths, the average probability that a keying message has been successfully attacked during transmission can be derived from historical statistics of Byzantine attacks. Assume that the probability is i.i.d. for each keying message and is given by P_v, P_e, and P_d for the revealing, erasure, and modification attacks respectively. Thus, a keying message remains attack-free with probability $P_0 = 1-P_v-P_e-P_d$. For a key establishment scheme using m keying messages, the probability that exactly v revealing attacks, e erasure attacks, and d modification attacks have been successful is

$$\begin{aligned} & P_m(v,e,d) \\ &= \binom{n}{v}\binom{n-v}{e}\binom{n-v-e}{d} P_v^v P_e^e P_d^d P_0^{n-v-e-d} \\ &= \frac{(n!)P_v^v P_e^e P_d^d P_0^{n-v-e-d}}{(v!)(e!)(d!)(n-v-e-d)!} \end{aligned} \tag{8}$$

According to Theorem 1, a secret key can be established securely if and only if $2d+e+v \leq m-1$. For given Byzantine attack statistics, we can derive the maximum secure key establishment probability as the solution to the following optimization problem over the set of achievable Byzantine resilience vectors:

$$\max_{v,e,d} \quad \sum_{i \leq v, j \leq e, l \leq d} P_m(i,j,l) \tag{9}$$

$$\text{s.t.} \quad 2d+e+v \leq m-1, \quad v,d,e \geq 0 \tag{10}$$

The optimizer of problem (9) defines the optimal key establishment scheme with the best security performance.

In general, problem (9) can be solved by an exhaustive search over all Pareto optimal vectors satisfying the bound $2d + e + v = m - 1$. However, if $P_v, P_e, P_d \leq \frac{P_0}{m}$ holds (i.e. attack probabilities are small), then we can prove that $P_m(v, e, d)$ is a monotonically decreasing function over v, e, and d, respectively. In this case, the optimization problem (9) can be easily solved by a greedy algorithm that incrementally improves v, e , and d by a unit step-size according to a discrete gradient $[P_m(v+1, e, d), P_m(v, e+1, d), P_m(v, e, d+1)]$. The greedy algorithm is guaranteed to converge to the optimal Byzantine resilience vector that satisfies $2d+e+v = m-1$ and achieves the maximum secure key establishment probability. Details of the algorithm are omitted.

V. Low-Complexity Key Establishment Scheme Using XOR

Theorem 1 gives the optimal Byzantine resilience that is achievable by the key establishment scheme in Algorithm 1. However, the $(v + 1)m$ multiplications of large integers in GF_p with $p > 2^k$ for constructing keying messages and the $2^{(v+1)m}$ Hamming distance computation for recovering the secret key with a sphere decoder renders Algorithm 1 impractical in wireless ad hoc networks. In this section, we derive a class of low-complexity key establishment algorithms that only requires bitwise XOR operations and simple table lookups. The new algorithm, generalized from linear binary error control codes, is able to achieve a nearly-optimal performance. We first describe the proposed algorithm and then provide a security and complexity analysis.

A. Syndrome Decoding for Linear Binary Codes

A linear binary code $\mathcal{C}$ is a linear subspace of the field of binary vectors. If $\mathcal{C}$ is an (m, t, s)-code, then it encodes vectors of length t into codewords of length m, whose minimum Hamming distance is s. Let G of size $m \times t$ be a generating matrix for this linear code. Codewords are obtained by linear combinations of the rows of G, i.e. if $\vec{x}$ is a vector of length t, then $\vec{y} = G\vec{x}$ has length m and is the codeword for $\vec{x}$.

To correct both error and modification in a received codeword, the following syndrome decoding procedure for binary linear codes can be employed: Let H be a parity check matrix for code $\mathcal{C}$. We first replace the erased coordinates by all zeros (denoted by $\vec{y^0}$) and all ones (denoted by $\vec{y^1}$) and compute two different syndromes (i.e. $\vec{r^0} = H^T\vec{y^0}$ and $\vec{r^1} = H^T\vec{y^1}$) respectively. By looking up $\vec{r^0}$ and $\vec{r^1}$ in the syndrome table to obtain two different error vectors $\vec{t^0}$ and $\vec{t^1}$, the one that contains fewer number of errors on non-erased coordinates gives us the correct syndrome that should be chosen. More precisely, if $\vec{r^0}$ (or $\vec{r^1}$ instead) gives fewer errors, then the original codeword can be recovered by inserting zeros (or ones) on the erased coordinates and then subtracting the error vector $\vec{t^0}$ (or $\vec{t^1}$) i.e. $\vec{y} = \vec{y^0} - \vec{t^0}$ (or $\vec{y} = \vec{y^1} - \vec{t^1}$). In classical coding theory, it has been proven that an (m, t, s)-code is able to correct any e erasures and d modifications at the same time, given that $2d + e \leq s - 1$ [17].

The following example contains a generating matrix and a parity check matrix for an $(8, 2, 5)$ linear binary code

$$G = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}^T,$$

$$H = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}^T.$$

For an input vector $\vec{x} = [1\ 1]^T$, the corresponding codeword is given by $\vec{y} = G\vec{x} = [1\ 1\ 1\ 0\ 0\ 1\ 1\ 1]^T$. Now, suppose that the first two bits of $\vec{y}$ are erased and the third bit is flipped, i.e. the received vector becomes $\vec{\hat{y}} = [*\ \ *\ \ 0\ 0\ 0\ 1\ 1\ 1]^T$. We replace erased coordinates in $\vec{\hat{y}}$ with ones and zeros respectively and compute two syndromes $\vec{r^0} = [0\ 0\ 0\ 0\ 0\ 1]^T$ and $\vec{r^1} = [0\ 1\ 0\ 0\ 0\ 1]^T$. By looking up the syndrome table for this $(8, 2, 5)$-code, we get $\vec{t^0} = [0\ 0\ 0\ 1\ 1\ 0\ 0\ 0]$ and $\vec{t^1} = [0\ 0\ 1\ 0\ 0\ 0\ 0\ 0]$. Since $\vec{t^0}$ contains two errors on non-erased coordinates, while $\vec{t^1}$ contains only one error, we choose all ones on the erased bits in $\vec{\hat{y}}$ and subtract $\vec{t^1}$ from it. This gives us the correct codeword $\vec{y}$. In the next section, we generalize this syndrome decoding method and derive an algorithm for secure key establishment. The proposed algorithm not only corrects modifications and erasures, but also achieves unconditional secrecy for end-to-end keys.

B. Low-Complexity Key Establishment

Let $X_1, \ldots, X_t$ be a set of length-k pseudo-random vectors constructed at the source node. We stack these t random vectors into a matrix $[X_1, \ldots, X_t]$ and encode each row of the matrix using a $(m + 1, t, s)$ error control code, which has a generating matrix

$$G = \begin{bmatrix} g_{01} & g_{02} & \cdots & g_{0t} \\ g_{11} & g_{12} & \cdots & g_{1t} \\ \vdots & \vdots & \ddots & \vdots \\ g_{m1} & g_{m2} & \cdots & g_{mt} \end{bmatrix}_{(m+1)\times t} \tag{11}$$

The codewords form a matrix $G \cdot [X_1, \ldots, X_t]^T$ of size $(m + 1) \times (k)$. Now, we choose a secret key as the first row of the codeword matrix and keying message $\mathbb{M}_i$ as the $(i+1)$'th row, for $i = 1, \ldots, m$, i.e.

$$[K_{SD}, M_1, \ldots, M_m]^T = G \cdot [X_1, \ldots, X_t]^T \tag{12}$$

Since linear binary codes are used, all operations required are simply binary XORs. Let $\oplus$ be the bitwise XOR operator for vectors. The algorithm for generating keying messages at the source node is summarized as follows:

Algorithm 2: *Generating Keying Messages*

1) The source node constructs t length-k pseudo-random vectors $\mathbb{X}_1, \ldots, \mathbb{X}_t$.

2) The end-to-end secret key is derived as

$$K_{SD} = (g_{01}X_1) \oplus (g_{02}X_2) \oplus \ldots \oplus (g_{0t}X_t)\,.$$

3) Initialize $i = 1$.

4) Source node generates keying message $\mathbb{M}_i$, sends it to the destination node, and erases $\mathbb{M}_i$ locally:

$$\mathbb{M}_i = (g_{i1}X_1) \oplus (g_{i2}X_2) \oplus \ldots \oplus (g_{it}X_t)\,.$$

5) If $i < n$, let $i = i + 1$ and go to step 4.

6) Source node erases $X_1, \ldots, X_t$ from his memory.

Without loss of generality, we assume that the last e keying messages are unavailable to the destination node due to erasure attacks and the remaining $m - e$ keying messages contain d faulty ones due to modification attacks. Let H be a parity check matrix of size $(m + 1) \times (m + 1 - t)$ for the generating matrix in (11). We use the following algorithm for deriving the secret key at the destination node:

Algorithm 3: *Deriving Secret Key*

1) Destination node receives at least $m-e$ keying messages $\hat{M}_1, \ldots, \hat{M}_{m-e}$.

2) Define a mask vector A according to the indices of received keying messages: $A_1 = 0$ and

$$A_{i+1} = \begin{cases} 1, & \text{if } \hat{M}_i \text{ is received} \\ 0, & \text{otherwise} \end{cases} \quad \forall i = 1, \ldots, m.$$

3) Destination node computes a submatrix $\tilde{H}$, consisting of the $m - e$ non-erased rows of H:

$$\tilde{H}_i = H_{i+1}, \text{ for } i = 1, \ldots, m - e.$$

4) Destination node computes a syndrome perturbation vector $\tilde{r}$ as the XOR of the $e + 1$ erased rows of H:

$$\tilde{r} = H_1 \oplus H_{m-e+2} \ldots \oplus H_{m+1}.$$

5) Destination node derives $R^0 = \tilde{H}^T \cdot \left[\hat{\mathbb{M}}_1, \ldots, \hat{\mathbb{M}}_{m-e}\right]^T$.

6) Initialize $i = 1$. Let $ADDR$ be the base address of the syndrome table stored at the destination node.

7) Retrieve $\vec{t^0}$ from address $ADDR + R_i^0$.

8) Retrieve $\vec{t^1}$ from address $ADDR + (R_i^0 \oplus \tilde{r})$.

9) The i'th bit of K_{SD} is given by

$$K_{SD,i} = \begin{cases} \vec{t_1^0}, & \text{if } popcnt(\vec{t^0} \bigwedge A) < popcnt(\vec{t^1} \bigwedge A) \\ 1 \oplus \vec{t_1^1}, & \text{otherwise} \end{cases}$$

10) If $i < k$, let $i = i + 1$ and go to step 5.

In Step 5 above, each row of $\left[\hat{M}_1, \ldots, \hat{M}_{m-e}\right]$ is a valid codeword generated by (11) with $e + 1$ erasures and d modifications. According to the syndrome decoding procedure described in Section V.A, if we assume that the erased keying messages are all zero vectors, we can compute a syndrome matrix $R^0 = \tilde{H}^T \cdot \left[\hat{M}_1, \ldots, \hat{M}_{m-e}\right]^T$, where each column of R^0 is a syndrome vector. On the other hand, if we assume that the erased keying messages are all one vectors, it is easy to show that the syndrome for the i'th row of $\left[\hat{M}_1, \ldots, \hat{M}_{m-e}\right]$ becomes $\tilde{r} \oplus R_i^0$, with $\tilde{r}$ as a perturbation vector defined in Step 4. Thus, by looking up the syndrome table and comparing the corresponding error vectors, we can recover the first bit of the secret key, and thereafter bit by bit. In Step 9. $popcnt$ is a population count instruction which counts the number of "1" bits in a word and cmp means comparison of two words [18]. To facilitate table lookups, we use computed syndromes as the indexes into the syndrome table.

C. Security Analysis

Theorem 2: For a linear binary error control code $(m + 1, t, s)$ with dual code $(m+1, m+1-t, s')$, the proposed low-complexity key establishment algorithm (i.e. Algorithms 2 and 3) achieves a Byzantine fault tolerance vector $(v, e, d)_m$ for $v = s'-2$ and $2d+e = s-2$. In particular, when both codes are maximum distance separable (MDS), the proposed algorithm achieves an optimal Byzantine resilience of $2d+e+v = m-3$.

Proof: We first prove that the proposed algorithm can recover the secret key under e erasure and d modification attacks, and then show attacks have absolutely no information about the secret key with v revealing attacks.

Since each row of the codeword matrix $[K_{SD}, M_1, \ldots, M_n]$ is a valid codeword for the $(m + 1, t, s)$ error control code, classical coding theory shows that up to $\lfloor \frac{s-1}{2} \rfloor$ errors can be corrected by syndrome decoding. In Algorithm 3, we choose the $e + 1$ erased keying messages to be all zeros or all ones. Because the error control code is binary, one of the two choices introduces no more than $\lfloor \frac{e+1}{2} \rfloor$ new errors, and thus leads to no more than $d + \lfloor \frac{e+1}{2} \rfloor$ errors in total. These errors can be corrected by the syndrome decoding in Algorithm 3, if the following is satisfied:

$$d + \lfloor \frac{e+1}{2} \rfloor = \lfloor \frac{2d+e+1}{2} \rfloor \leq \lfloor \frac{s-1}{2} \rfloor \tag{13}$$

This establishes $2d + e \leq s - 2$ as a sufficient condition for recovering secret key K_{SD}.

To show that secret key K_{SD} remains completely unknown to attackers, without loss of generality, we assume that keying messages $M_1, \ldots, M_v$ are revealed to attackers. According to (12), attackers have $v + 1$ equations in the following matrix representation

$$\begin{bmatrix} K_{SD}^T \\ M_1^T \\ \vdots \\ M_v^T \end{bmatrix} = \begin{bmatrix} g_{01} & g_{02} & \cdots & g_{0t} \\ g_{11} & g_{12} & \cdots & g_{1t} \\ \vdots & \vdots & \ddots & \vdots \\ g_{v1} & g_{v2} & \cdots & g_{mt} \end{bmatrix} \cdot \begin{bmatrix} X_1^T \\ X_2^T \\ \vdots \\ X_t^T \end{bmatrix} \tag{14}$$

Because the dual error control code $(m + 1, m + 1 - t, s')$ has distance s', classical coding theory shows that any $s' - 1$ rows of the G matrix are linearly independent. Further, s' is upper bounded by $s' \leq t + 1$. When $v \leq s' - 2$ as claimed in the statement of Theorem 2, we also have $v + 1 \leq t$. This

implies that the first matrix on the right hand side of (14) is full row-rank.

Thus, when $M_1, \ldots, M_v$ are fixed in (14), for each possible choice of secret key K_{SD}, equation (14) defines a system of $v+1$ linear equations with t unknowns, i.e. $\mathbb{X}_1, \ldots, \mathbb{X}_t$. There exists 2^{t-v-1} possible $\mathbb{X}_1, \ldots, \mathbb{X}_t$ vectors such that (14) is satisfied. More precisely, since vectors $\mathbb{X}_1, \ldots, \mathbb{X}_t$ are generated randomly by a uniform distribution, we have

$$\begin{aligned}
&\text{Prob}\left\{K_{SD} = \hat{K} \middle| [M_1, \ldots, M_v] = \hat{M}\right\} \\
&= \frac{\text{Prob}\left\{K_{SD} = \hat{K}, [M_1, \ldots, M_v] = \hat{M}\right\}}{\sum_K \text{Prob}\left\{K_{SD} = K, [M_1, \ldots, M_v] = \hat{M}\right\}} \\
&= \frac{\text{Prob}\left\{[X_1, \ldots, X_t] \in \mathcal{X}_{\hat{K},\hat{M}}\right\}}{\sum_K \text{Prob}\left\{[X_1, \ldots, X_t] \in \mathcal{X}_{K,\hat{M}}\right\}} \\
&= \frac{1}{2^k} \qquad (15)
\end{aligned}$$

where $\mathcal{X}_{\hat{K},\hat{M}}$ is the set of all $X_1, \ldots, X_t$ satisfying (14) for $K_{SD} = \hat{K}$ and $[M_1, \ldots, M_v] = \hat{M}$. Equation (15) used the fact that $\left|\mathcal{X}_{\hat{K},\hat{M}}\right| = 2^{t-v}$ for all $\hat{K}$ and $\hat{M}$ and that $\mathbb{X}_1, \ldots, \mathbb{X}_t$ are uniformly distributed. From (15), we conclude that given keying messages $\mathbb{M}_1, \ldots, \mathbb{M}_v$, unconditional secrecy as defined in (1) is achieved if $v \leq s' - 2$.

In addition, according to classical coding theory, for binary error control codes, we have $s+s' = m+1$ when both the primal and the dual codes are maximum distance separable.Thus, we derive $v + 2d + e = s + s' - 4 = m - 3$, which is the desired result. ■

D. Complexity Analysis

We analyze the complexity of the proposed key establishment algorithm in terms of computation overhead and storage space. For computation overhead, since we are restricted to linear binary codes in this paper, all operations are performed in Gf_2. We observe that the algorithm consists of four basic operations: binary XOR, table lookup, pseudo-random vectors, and assembly instructions (i.e. $popcnt$ and cmp). For storage space, a syndrome table, generating and parity check matrices, and auxiliary vectors have to be stored at each node.

Table II summarizes the complexity of our proposed key establishment algorithm. As a numerical example, for a network using $m = 30$ keying messages and an AES encryption with key size $k = 128$, the complexity is on the order of 200K operations and 4M bits of storage for generating keying messages and recovering a secret key. Our proposed algorithm, which is able to guard against all three attacks in the Byzantine attack model, is much less complex than previous key establishment schemes [14][15].

Remark 2: The complexity analysis summarized in Table.II is derived for the worst-case. A practical implementation of the proposed key establishment algorithm may have much lower complexity by performing the algorithm in parallel. For example, multiple entries in the syndrome table can be accessed at once, such that the complexity for table lookups can be greatly reduced. Similarly, a logic circuit that consists of multiple bitwise XOR logic gates can be used to perform the XOR of vectors.

Complexity Metrics		Generating	Recovering
Computation	Bitwise XOR	$o(km^2)$	$o(km^2)$
	Random Vector	$o(m)$	-
	Table Lookup	-	$o(k)$
	$popcnt$ and cmp	-	$o(k)$
Total Computation		$o(km^2)$	$o(km^2)$
Storage (bits)	Syndrome Table	-	$o(m2^{m-t})$
	Coding Matrices	$o(m^2)$	$o(m^2)$
	Auxiliary Vectors	$o(km)$	$o(km)$
Total Storage		$o(km+m^2)$	$o(km+m2^{m-t})$

TABLE II
SUMMARY OF THE COMPLEXITY ANALYSIS FOR THE PROPOSED KEY ESTABLISHMENT ALGORITHM IN TERMS OF COMPUTATION OVERHEAD AND STORAGE SPACE.

VI. NUMERICAL EXAMPLES

Consider a wireless ad hoc network with $N = 1000$ nodes, uniformly distributed in a square area of size $L = 100$. We assume that nodes in the neighborhood of Communication range $R = 15$ share pre-installed keys with probability p. These pre-installed link keys are used to secure keying messages during transmission. To discover m separate message forwarding paths for each node pair, we implement the Zone Routing Protocol (ZRP) [16] with a zone radius of 2 hops. In all simulations, compromised nodes are randomly selected from the N nodes such that the locations of compromised nodes are also uniformly distributed in the area.

We define the probability of secure key establishment as the average probability that two nodes can successfully establish an end-to-end secret key, and at the same time, the secret key remains completely unknown to attackers even if they collude. For $p = 0.5$ and the optimal key establishment algorithm (i.e. Algorithm 1), Fig. 3 plots the probability of secure key establishment for the use of $m = 1, 5, 10, 20, 30, 40$ keying messages, under Byzantine attacks with equal probabilities. It can be observed that the optimal key establishment algorithm with $m \geq 20$ can safeguard secret keys with a successful probability of over 80% for as many as 80 (i.e. 8%) malicious nodes, and its security performance benefits from the increase of keying messages as more path diversity is exploited. In another simulation with $m = 30$, Fig. 4 shows that the probability of secure key establishment remains almost the same for different pre-installed key-sharing probabilities $p = 0.5, 0.6, 0.7, 0.8$. This observation implies that no matter what key pre-distribution algorithm is used, the Byzantine resilience achieved by the proposed optimal key establishment algorithm can be guaranteed. Thus, complicated key pre-distribution algorithms that are intended to provide high pre-installed key-sharing probability may not be necessary, since

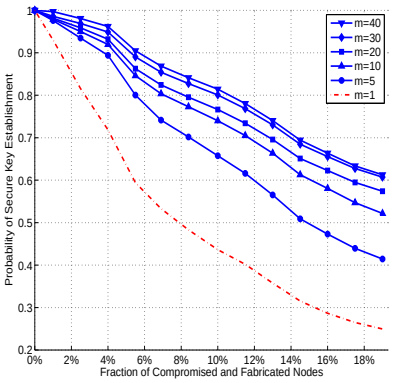

Fig. 3. Probability of secure path key establishment v.s. number of compromised nodes for $m = 1, 5, 10, 20, 30, 40$ keying messages.

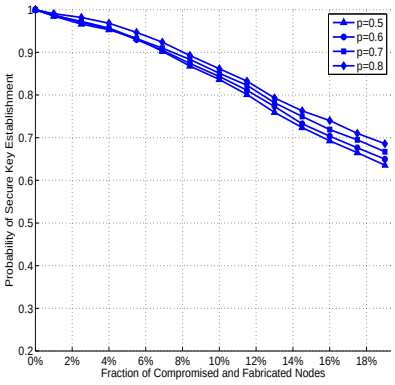

Fig. 4. Probability of secure path key establishment v.s. number of compromised nodes for pre-installed key-sharing probability $p = 0.5, 0.6, 0.7, 0.8$.

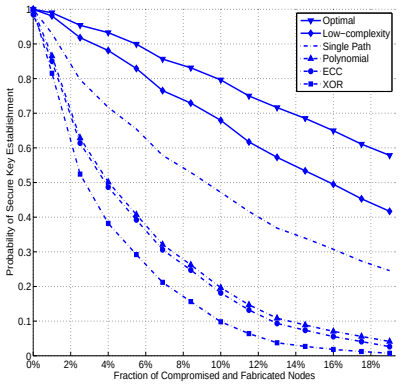

Fig. 5. Compare the security of different key establishment schemes under Byzantine attacks.

full key-connectivity can be achieved by our on-demand key establishment algorithm.

For the same network model with $p = 0.5$ and $m = 30$, we compare the security of the optimal key establishment algorithm in Section III, the low-complexity key establishment algorithm in section V, key establishment using single path, and the three previous multi-path key establishment schemes. Our two proposed key establishment algorithms both achieve a significant Byzantine resilience improvement over previous schemes, while the lower-complexity algorithm in Section V has a performance that is close to the optimal one, and is more suitable for practical implementations. This comparison highlights the importance of defending against multiple attacks simultaneously: The overall resilience of a security algorithm is determined by the worst individual-attack resilience (i.e. $\min(v, e, d)$) against Byzantine attacks. It also demonstrates the excellent security-complexity properties of the proposed method.

VII. Conclusion

This paper proposes a unifying framework for analyzing the Byzantine-resilience of any key establishment scheme, quantified by a new security metric we call a Byzantine resilience vector. A universal bound on achievable Byzantine resilience vectors is derived in closed-form and can be attained by our proposed optimal key establishment algorithm. For practical implementations, we also develop a low-complexity key establishment algorithm that achieves nearly-optimal Byzantine resilience. Our analysis and simulation show that the capability of defending against multiple attack classes simultaneously is critical for the security of wireless ad hoc networks.

References

[1] B.C. Neuman and T. Tso, "Kerberos: An Authentication Service for Computer Networks," *IEEE Comm. Magazine*, vol. 32, no. 9, pp. 33-38, 1994.

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

[3] L. Eschenauer and D. Gligor, "A key-management scheme for distributed sensor networks", *in Proc. ACM Computer and Communications Security Conference*, November 2002.

[4] H. Chan, A. Perrig and D. Song, "Random Key Pre-distribution Schemes for Sensor Networks", *in Proc. IEEE Symposium on Research in Security and Privacy*, May 2003.

[5] W. Du , J. Deng , Y. S. Han , and P. K. Varshney, "A pairwise key pre-distribution scheme for wireless sensor networks", *in Proc. ACM Computer and Communications Security Conference*, October, 2003.

[6] W. Du, J. Deng, Y.S. Han, S. Chen, and P. Varshney, "A Key Management Scheme for Wireless Sensor Networks Using Deployment Knowledge", *in Proc. of IEEE INFOCOM*, March 2004.

[7] D. Liu, P. Ning, and W. Du, "Group-Based Key Pre-Distribution in Wireless Sensor Networks", *in Proc. ACM Workshop on Wireless Security*, September 2005.

[8] L. Zhou, J. Ni, and C.V. Ravishankar, "Efficient Key Establishment for Group-Based Wireless Sensor Deployments:, *in Proc. ACM Workshop on Wireless Security*, September 2005.

[9] D. Liu, P. Ning, and R. Li, "Establishing Pairwise Keys in Distributed Sensor Networks", *ACM Transactions on Information Systems Security*, vol. 8, no. 1, pp. 41-77, 2005.

[10] E. Shi and A. Perrig, "Designing Secure Sensor Networks", *Wireless Communication Magazine*, vol. 11, no. 6, pp. 38-43, 2004.

[11] W. Du, R. Wang, and P. Ning, "An Efficient Scheme for Authenticating Public Keys in Sensor Networks", *in Proc. ACM MobiHoc*, May 2005.

[12] D. Xu, J. Huang, J. Dwoskin, M. Chiang, and R. Lee, "Re-examining Probablistic Versus Dtermininistic Key Management," *in Proc. IEEE Symposium on Informaiton Theory*, June 2007.

[13] S. Zhu, S. Xu, S. Setia, and S. Jajodia. "Establishing pair-wise keys for secure communication in ad hoc networks: A probabilistic approach", *in Proc. IEEE International Conference on Network Protocols*, November 2003.

[14] A. Shamir, "How to Share a Secret", *Communications of the ACM*, 1979.

[15] D. Huang and D. Medhi, "A Byzantine resilient multi-path key establishment scheme and its robustness analysis for sensor networks", *in Proc. IEEE International Parallel and Distributed Processing Symposium*, April 2005.

[16] Z.J. Haas and M.R. Pearlman, "The Performance of Query Control Schemes for the Zone Routing Protocol", *IEEE Transactions on Networking*, vol. 9, no. 4, pp. 427 - 438, 2001.

[17] S. Lin and D.J. Costello, "Error Control Coding: Fundamentals and Applications", *Prentice Hall*, 1983.

[18] Intel Architecture Software Developers Manual, "Volume 2: Instruction Set Reference", 1999.