Key Agreement for Wireless Networks in the Presence of Active Adversaries

Hongyi Yao[1], Tracey Ho[1] and Cristina Nita-Rotaru[2]
[1]California Institute of Technology, [2]Purdue University

Abstract—In this work we consider key agreement in a distributed wireless network in which adversarial nodes may exist in the network. Two practical scenarios are studied, i.e., path-key establishment and key pool bootstrapping. For both scenarios, previous work mainly focuses on preventing adversarial eavesdropping attacks. When the adversarial nodes can both eavesdrop and change the packets in the network, we show that an optimal scheme (in terms of error correction capability) can be obtained by using techniques from the literature of network error correction codes. In particular, for the path-key establishment problem, which has a particular topological structure, we propose a lightweight scheme with lower error probability and computational complexity than existing network error correction codes. For the key pool bootstrapping scenario, we show a connection with multisource network error correction coding, and show that network coding is necessary for optimal resilience against adversarial attacks.

I. INTRODUCTION

In this paper we consider key agreement protocols for wireless networks in which several nodes are compromised by an adversary. We assume active attacks, in which the compromised nodes are able to transmit fake packets to disrupt the key agreement process.

The contribution of this paper is showing that information theoretic network error correction codes [2][6] can be applied to achieve optimal performance for two practical key agreement scenarios. Compared with cryptographic authentication, schemes based on information-theoretic error correction codes do not need pre-key set up, achieve low computational overhead, and are secure even when the adversary has unlimited computational capability.

Two scenarios are considered. In the first scenario, path-key establishment [3], we consider a multihop wireless network. In such network, two nodes, A and B,

want to set up multi-hop secure communications, without an *a priori* shared secret key. To establish the secure channel, nodes A and B need to communicate over the multi-hop wireless network to establish a secret key. We assume a pre-distribution scheme [1] exists, such that each pair of nodes share a secret key with some probability. In this scenario, each node which shares secret keys with both A and B in fact sets up a logical two-hop path between A and B. Thus, the problem of key agreement between A and B can be reduced to the problem of secure communication in a two-hop parallel path network. When some of the wireless nodes can be compromised by an adversary, the work in [6] shows the maximum rate at which the source can transmit secretly and securely when using network error correction codes. We propose a new code which achieves the same rate, but, being designed for a specific topology, achieves better performance, i.e. lower computational complexity and probability of error. The new scheme is an extension of [4] which only considers error correction but not the secrecy of the messages.

In the second scenario, key pool bootstrapping, there is a set of keys (key pool), $\mathcal{K}$, generated by a key center. Each network node, say V, needs a subset of keys from $\mathcal{K}$. Rather than each node having to communicate directly with the key center, nodes in the network set up keys in a recursive manner. In particular, each node V retrieves the desired keys from its neighbors who are closer to the key center and have already retrieved their keys in previous stages. When several neighbors of V are compromised by an adversary, we show that multisource network error correction codes [2] are able to achieve the maximum secure rate. An interesting observation is that network coding is necessary for optimal resilience against such active adversarial attacks.

II. PATH-KEY ESTABLISHMENT

A. Problem Formulation

Let $\mathbb{F}_q$ be a finite field with size q. We assume that a key pre-distribution scheme [1] exists, such that each
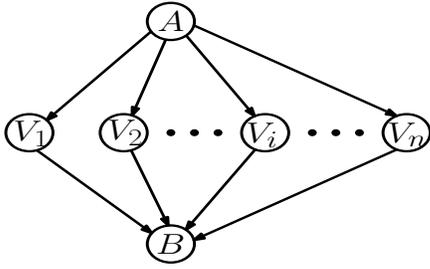
Fig. 1. Logical two-hop model for path-key establishment.

Fig. 2. Illustrating example for the scheme in [4].

pair of nodes shares a secret key with some probability.

For a pair of nodes A and B who want to set up secure communications but do not share any secret key in the pre-distribution scheme, a path-key establishment protocol allows them to establish a secret key in a multi-hop fashion. Specifically, it allows A to transmit a secret key $K \in \mathbb{F}_q^k$ to node B with the help of the logical one-hop keys established with the key pre-distribution scheme. Let $\mathcal{V} = \{V_1, V_2, ..., V_n\}$ be the set of nodes which share secret keys with both A and B. Thus, each node in $\mathcal{V}$ has a secure secure one-hop link to both A and B. As shown in Figure 1, the nodes in $\mathcal{V}$ set up n logical two-hop parallel paths between A and B. Let C symbols in $\mathbb{F}_q$ be transmitted over each path.

We assume z nodes in $\mathcal{V}$ are compromised by an adversary, who can both eavesdrop on the packet and change the packet contents arbitrarily so as to disrupt the key establishment protocol. Paths through the non-compromised nodes are secure.

B. Background

Our scheme builds on two previous results. The network error correction code [6] gives the maximum rate at which the source can transmit secretly and securely; our construction achieves the same maximum rate but with a lower computational complexity and a higher error correction probability. Our new scheme is an extension of that in [4] which only considers error correction but not the secrecy of the messages.

Directly applying the result in [6], which provides polynomial time encoding and decoding algorithms for all achievable rates, gives the following:

Theorem 1. *For the path-key establishment problem above,*

- *If $z \leq \lfloor (n-1)/2 \rfloor$, there exists exists polynomial time scheme such that A can transmit K to B secretly and correctly. The computational complexity is $\mathcal{O}(C \log(n) M(n))$, where $M(n)$ is the matrix multiplication complexity for dimensions $n \times n$. In particular, the adversary can get zero information*
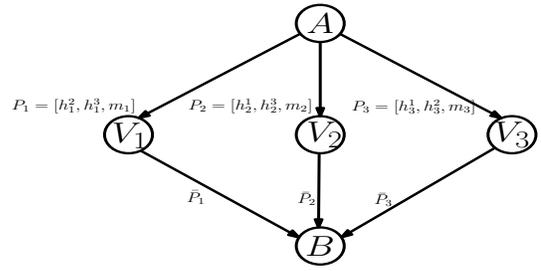
about K, and the probability that B cannot decode K is $\mathcal{O}(\exp(n)C/q)$.

- *Else if $z > \lfloor (n-1)/2 \rfloor$, for any k and C, with a probability 1 node A cannot transmit K to B for any scheme.*

We overview the scheme in [4] with an example where there are three nodes and one malicious node, $n = 3$ and $z = 1$, as shown in Figure 2. Node A constructs three packets P_1, P_2 and P_3, to deliver along V_1, V_2 and V_3 respectively. In particular, P_i is constructed as $[h_i^j, h_i^k, m_i]$, where m is the message part, $j \neq k$, $j \neq i$ (and $k \neq i$) and h_i^j (and h_i^k) is a hash check for the message part in packet P_j (and P_k).

Upon receiving the three packets $\{\bar{P}_1, \bar{P}_2, \bar{P}_3\}$, node B decodes as follows. For the message part of each received packet, say $\bar{m}_i$, node B checks how many hash checks in other packets agree with m_i. If there exists at least one hash check that agrees with $\bar{m}_i$, node B accepts $\bar{m}_i$ as the correct message m_i; otherwise, $\bar{m}_i$ is taken to be erroneous.

The scheme does not consider the secrecy of the message. For instance, assume node V_2 is compromised. Then both the message part m_2 and the hash checks h_2^1, h_2^3 would leak information about the secret key.

C. Our Construction for Secrecy and Error Correction

In this subsection, the complete construction is provided for the path-key establishment problem defined in II-A. In the following, we use z_m to denote $\lfloor (n-1)/2 \rfloor$.

Encoder at node A.

- Let $K \in \mathbb{F}_q^k$ denote the key that is desired to be transmitted from A to B.
- Node A independently and uniformly generates z_m random packets $\mathcal{R} = \{R_1, R_2, ..., R_{z_m}\}$ over $\mathbb{F}_q^k$.
- Node A independently and uniformly generates n random packets $\mathcal{D} = \{D_1, D_2, ..., D_n\}$ over $\mathbb{F}_q^{z_m}$.
- As shown in Figure 3, node A constructs n packets $\mathcal{P} = \{P_1, P_2, ..., P_n\}$, each of which is delivered through one of $\mathcal{V} = \{V_1, V_2, ..., V_n\}$.
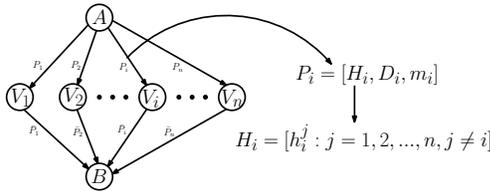
Fig. 3. Coding construction at node A.

- Packet P_i is constructed as $P_i = [H_i, D_i, m_i]$, as shown in Figure 3.
  - Message $m_i \in \mathbb{F}_q^k$ is a linear combination of K and the packets in $\mathcal{R}$, i.e., $m_i = \sum_{\ell=1}^{z_m}(c_i)^\ell R_\ell + (c_i)^{\ell+1}K$. We assume $c_1, c_2, ..., c_n$ are distinct and publicly known.
  - As shown in Figure 3, hash H_i is constructed as $H_i = [h_i^j : j = 1, 2, ..., n, j \neq i]$. Hash $h_i^j \in \mathbb{F}_q^2$ is defined as $h_i^j = [\alpha_i^j, v_i^j]$, where α_i^j is independently and uniformly chosen from $\mathbb{F}_q$ and v_i^j is the linear product between $[D_j, m_j]$ and vector $[(\alpha_i^j)^1, (\alpha_i^j)^2, ..., (\alpha_i^j)^{z_m+k}]$.

Each intermediate node V_i in $\mathcal{V}$ directly forwards P_i to node B.

Decoder at node B.

- From node V_i, node B receives packet $\bar{P}_i = [\bar{H}_i, \bar{D}_i, \bar{m}_i]$. • For message $\bar{m}_i$, node B checks whether hash $\bar{h}_j^i = [\bar{\alpha}_j^i, \bar{v}_j^i]$ agrees with $\bar{m}_i$, for each $j \neq i$.
  - Message $\bar{m}_i$ is said to agree with $\bar{h}_j^i$ if and only if $\bar{v}_j^i$ is equal to the inner product of $[\bar{D}_j, \bar{m}_j]$ and vector $[(\bar{\alpha}_j^i)^1, (\bar{\alpha}_j^i)^2, ..., (\bar{\alpha}_j^i)^{z_m+k}]$.
- Message $\bar{m}_i$ is accepted as a valid message if and only if there are at least z_m hashes that agree with it. • If there are more than z_m+1 valid messages, node B has z_m+1 random linear combinations of K and $\mathcal{R}$. If they are all linear independent, node B can decode K by Gaussian Elimination.

In the next subsection, we prove that when the number of compromised nodes z is no larger than z_m, with high probability node B can decode K.

D. Theoretical Analysis

For the codes constructed above, we have the following theorem.

Theorem 2. *When $z \leq z_m$, the adversary gets zero information about the key K, and node B can decode K with a probability at least $1 - \mathcal{O}(n^2C/q)$. The encoding and decoding complexity is $\mathcal{O}(C \cdot M(n))$, where $M(n)$ is the complexity of matrix multiplication for dimensions $n \times n$.*

Proof: We first prove the secrecy. Without loss of generality, we assume $z = z_m$, nodes $V_1, ..., V_{z_m}$ are compromised and therefore packets $P_1, P_2, ..., P_{z_m}$ are eavesdropped. For any $j > z_m$, if $\alpha_1^j, \alpha_2^j, ..., \alpha_{z_m}^j$ are not distinct, the information in $h_1^j, h_2^j, ..., h_{z_m}^j$ is redundant. Thus, we can assume $\alpha_1^j, \alpha_2^j, ..., \alpha_{z_m}^j$ are distinct, which only strengthen the eavesdropping capability of the adversary.

To prove the secrecy, it is sufficient to prove that for any $K \in \mathbb{F}_q^k$, there exists exactly one possible value of $(\mathcal{D}, \mathcal{R})$ resulting in the adversarial observation $\{P_1, P_2, ..., P_{z_m}\}$. For any $K \in \mathbb{F}_q^k$, to result in $\{m_1, m_2, ..., m_{z_m}\}$, the z_m vectors in $\mathcal{R}$ needs to satisfy z_m linear vector equations. Since $c_1, c_2, ..., c_{z_m}$ are distinct, due to the property of Vandermonde matrix, such z_m linear vector equations are independent. Thus, there is exactly one possible sample of $\mathcal{R}$.

For each $j > z_m$, to agree with $\{h_1^j, h_2^j, ..., h_{z_m}^j\}$, the z_m elements in D_j needs to satisfy z_m linear equations. Since $\alpha_1^j, \alpha_2^j, ..., \alpha_{z_m}^j$ are distinct, due to the property of Vandermonde matrix, such z_m linear equations are independent. Thus, there is exactly one possible sample of D_j.

Thus, we complete the proof of secrecy. In the following we prove that node B can decode key K with probability $1 - \mathcal{O}(nC/q)$. Since $n \geq 2z_m + 1$, for any valid received packet $\bar{P}_i = [\bar{H}_i, \bar{D}_i, \bar{m}_i] = P_i$, there are at least z_m hashes from other received packets that agree with $\bar{m}_i$. For any corrupted packet $\bar{P}_i = [\bar{H}_i, \bar{D}_i, \bar{m}_i]$, if $[\bar{D}_i, \bar{m}_i] \neq [D_i, m_i]$, in the following we compute the probability that the hash of a valid packet agrees with $\bar{m}_i$. Assume $\bar{P}_j = P_j$ is valid, since the adversary has no information about α_j^i, the probability that it can forge $[\bar{D}_i, \bar{m}_i]$ to agree with h_j^i is no more than $\mathcal{O}(C/q)$. Using union bound [5], the probability is no more than $\mathcal{O}(nC/q)$ that $[\bar{D}_i, \bar{m}_i]$ agrees with the hash of a valid packet.

Also using the union bound [5], the probability is no more than $\mathcal{O}(n^2C/q)$ that node B will accept a corrupted message $\bar{m}_i$. Thus, with the at least $z_m + 1$ valid messages, node B can decode K by solving z_m+1 linear vector equations. Since any z_m+1 of $c_1, c_2, ..., c_n$ are distinct, due to the property of Vandermonde matrix, such $z_m + 1$ linear vector equations are independent. Thus, node B can decode K correctly.

For node A, the computational complexity of constructing each P_i is $\mathcal{O}(nC)$. Thus the total complexity of A is $\mathcal{O}(n^2C)$. For node B, the computational complexity of checking each $\bar{P}_i$ is $\mathcal{O}(nC)$. Thus the total complexity of checking is $\mathcal{O}(n^2C)$. Since, the complexity of linear decoding K from the valid messages is $\mathcal{O}(M(n)C)$, the total complexity of B is $\mathcal{O}(M(n)C)$.

III. KEY POOL BOOTSTRAPPING

In this section, we apply the theoretical multi-source network error correction codes [2] to the practical key pool bootstrapping problem [1].

A. Problem Formulation

Let $\mathcal{S} = \{K_1, K_2, ...\}$ be a set of keys generated by a key center, each of which is a vector in $\mathbb{F}_q^k$. Each wireless node V desires a subset $\mathcal{S}_V = \{K_{v_1}, K_{v_2}, ..., K_{v_m}\}$ of keys from $\mathcal{K}$. Rather than every node communicating directly with the key center, nodes in the network set up keys in a recursive manner. For instance, node V obtains the desired keys from neighbours who are closer to the key center and have already obtained their keys in previous stages.

Let $\mathcal{V} = \{V_1, ..., V_n\}$ be the set of the neighbours of V, each of which has a subset of $\mathcal{S}_V$. For instance, in Figure 5, node V wants to recover keys $\{K_1, K_2, K_3\}$ from its neighbours $\{V_1, ..., V_9\}$, each of which has a subset of $\{K_1, K_2, K_3\}$.

We assume the links between V and its neighbours are secure, and a communication constraint whereby each neighbour node is able to transmit a vector in $\mathbb{F}_q^k$ to V.

We assume a subset of z neighbours are compromised by an omniscient adversary, who can transmit arbitrarily corrupted packets from the compromised nodes, and knows every packet transmitted by the other neighbours.

B. Connection between Multi-source Network Error Correction Coding and Key Pool Bootstrapping

In the following, we show a connection between the multi-source network error correction problem solved in [2] and the key pool bootstrapping problem defined above.

Consider a key-pool bootstrapping instance, in which node V wants to recover the keys in $\mathcal{S}_V = \{K_{v_1}, K_{v_2}, ..., K_{v_m}\}$. Nodes in $\mathcal{V} = \{V_1, V_2, ..., V_n\}$ are neighbours of V and each maintains a subset of $\mathcal{S}_V$. We construct the equivalent two-hop multi-source network shown in Figure 4. In the network, node R is the receiver, there are m sources $\{S_1, S_2, ..., S_m\}$ and S_i has message K_{v_i}. Nodes in $\{U_1, U_2, ..., U_n\}$ form the intermediate layer. Each of the intermediate nodes has a link to the receiver R, which can transmit a packet in $\mathbb{F}_q^k$. For any i and j, if V_i in the key pool bootstrapping instance has key K_{v_j}, node U_i in Figure 4 has a reliable link with infinite capacity from source S_j.

If there is a error correction coding solution for the network in Figure 4, the nodes in $\mathcal{V}$ can emulate the solution to bootstrap $\mathcal{S}_V$ at V. On the other hand, if there
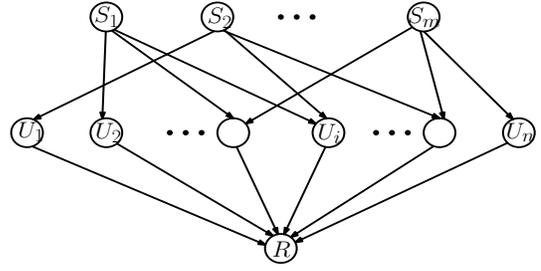

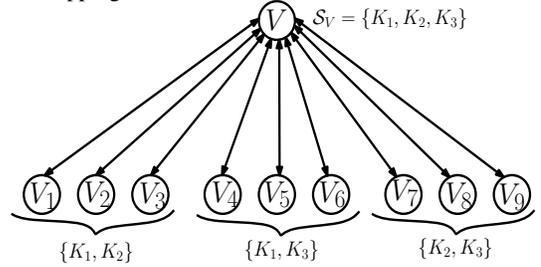Fig. 4. An equivalent multi-source multicast network model for key pool bootstrapping.

Fig. 5. The illustrating example key-pool bootstrapping.

is a solution for the key-pool bootstrapping instance, the sources in Figure 4 can forwarded their messages to the corresponding intermediate nodes allowing them to emulate the key-pool bootstrapping solution.

From [2], we have the following theorem for the key-pool bootstrapping problem.

Theorem 3. *Node V can decode $\mathcal{S}_V = \{K_{v_1}, K_{v_2}, ..., K_{v_m}\}$ in the presence of z compromised nodes if and only if for each non-empty subset $\mathcal{L}$ of $\{1, 2, ..., m\}$,*

$$|\mathcal{L}| \leq |\mathcal{V}_L| - 2z,$$

where $\mathcal{V}_L \subseteq \mathcal{V}$ contains each node in $\mathcal{V}$ that has at least one component in $\{K_{v_j}, j \in \mathcal{L}\}$.

C. Multi-source Network Error Correction Codes for Key Pool Bootstrapping

We consider the illustrating example shown in Figure 5. Node V wants to recover $\{K_1, K_2, K_3\}$ from its neighbours $\{V_1, V_2, ..., V_9\}$, each of which maintains a subset of $\{K_1, K_2, K_3\}$. In particular, nodes in group $\mathcal{G}_1 = \{V_1, V_2, V_3\}$ have $\{K_1, K_2\}$, nodes in group $\mathcal{G}_2 = \{V_4, V_5, V_6\}$ have $\{K_1, K_3\}$ and nodes in group $\mathcal{G}_3 = \{V_7, V_8, V_9\}$ have $\{K_2, K_3\}$. In the following, we show that network codes can achieve strictly better error correction performance than conventional error correction codes where there is no coding across different keys K_i.

Conventional error correction codes
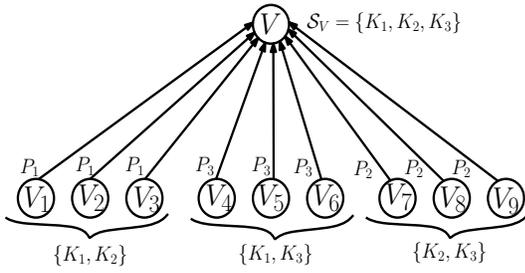
In a conventional error correction code, separate capacity

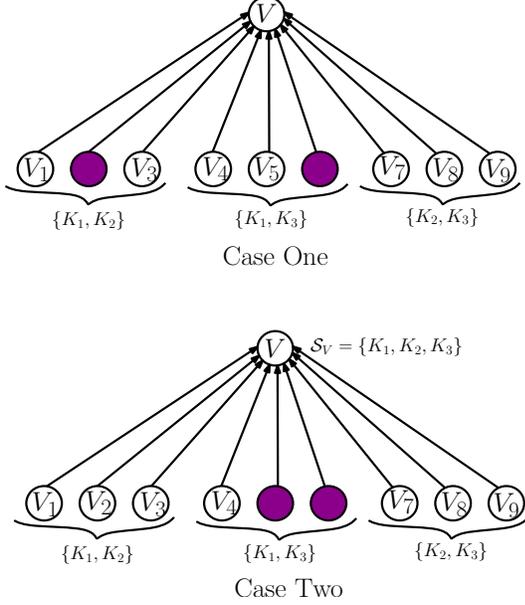Fig. 6. Conventional error correction codes.

Case One

Case Two

Fig. 7. There are two possible cases when two nodes are compromised by the adversary.

is allocated for transmission of each key. As shown in Figure 6, nodes in groups $\mathcal{G}_1$, $\mathcal{G}_2$ and $\mathcal{G}_3$ use majority coding to transmit keys K_1, K_2 and K_3, respectively. Thus, if there is only one compromised node, node V can recover $\{K_1, K_2, K_3\}$ correctly.

With two compromised nodes, there are two cases as shown in Figure 7. In case one, the two compromised nodes are in different groups. Therefore node V can still recover $\{K_1, K_2, K_3\}$ by majority decoding. However, in case two, the two compromised nodes are in the same group $\mathcal{G}_2$, and thus can forge the same erroneous key value K_2'. In such case, node V cannot decode K_2 correctly.

Error correction by network error correction codes
Using network error correction codes, each node in group $\mathcal{G}_1$ independently forms a random linear combination of K_1 and K_2, and then transmits such combined vector to V. Nodes in $\mathcal{G}_2$ and $\mathcal{G}_3$ similarly encode their respective keys.

Suppose there are two compromised nodes, as shown in Figure 7. Node B first checks how many groups have

consistent packets, *i.e.*, decoding from each subset of two nodes results in the same decoded key values. Since there are no more than two adversaries, there is at least one consistent group. If there are only one consistent group, the compromised nodes must be distributed as in case one. Assume $\mathcal{G}_3 = \{V_7, V_8, V_9\}$ is the Consistent group, then node V can decode K_2 and K_3 from $\mathcal{G}_3$. Using K_2, node V can compute K_1 from any packet transmitted from group $\mathcal{G}_1$. Two of the packet from $\mathcal{G}_1$ must be the same and valid. Thus node V can decode K_1 by majority decoding.

Assume there are two consistent groups. Since the inconsistent group must contain a compromised node, each of the consistent groups has at least two valid packets. Thus, the keys decoded from the two consistent groups are correct, and can be used to correctly recover $\{K_1, K_2, K_3\}$.

If there are three consistent groups, then the two invalid packets must be in the same group (case two). As illustrated in Figure 7, suppose the packets transmitted by V_5 and V_6 are invalid. Since the packet transmitted by the non-compromised node V_4 is a linear combination of the correct keys K_1 and K_3, the keys decoded from $\mathcal{G}_2$ must satisfy $K_1' \neq K_1$ and $K_3' \neq K_3$. Thus, the keys decoded from $\mathcal{G}_2$ are inconsistent with those from both $\mathcal{G}_1$ and $\mathcal{G}_3$. However, the keys decoded from $\mathcal{G}_1$ and $\mathcal{G}_3$ are consistent with each other. Thus, node V can tell that K_1' and K_3' from $\mathcal{G}_2$ are invalid.

By Theorem 3, correcting two compromised nodes is the maximum error correction performance for this example.

REFERENCES

[1] H. Chan, A. Perrig, and D. Song. Key distribution techniques for sensor networks. Technical report, Carnegie Mellon University. Avaible at: http://www.cs.berkeley.edu/ dawn-song/papers/randomkey.pdf.
[2] T. K. Dikaliotis, T. Ho, S. Jaggi, S. Vyetrenko, H. Yao, M. Effros, J. Kliewer, and E. Erez. Multiple-access network information-flow and correction codes. *IEEE Trans on Information Theory*, 2011.
[3] D. Huang and D. Medhi. A byzantine resilient multi-path key establishment scheme and its robustness analysis for sensor networks. In *Proc. of 5th IEEE International Workshop on Algorithms for Wireless, Mobile, Ad Hoc and Sensor Networks*, 2005.
[4] S. Jaggi, M. Langberg, T. Ho, and M. Effros. Correction of adversarial errors in networks. In *Proc. of ISIT*, 2005.
[5] M. Mitzenmacher and E. Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005.
[6] H. Yao, D. Silva, S. Jaggi, and M. Lanberger. Network codes resilient to jamming and eavesdropping. In *Proc. of NetCod*, 2010.