Perfectly Secure Message Transmission Revisited

Yongge Wang and Yvo Desmedt

***Abstract*—Secure communications guaranteeing reliability and privacy (without unproven assumptions) in networks with active adversaries has been an important research issue. It has been studied for point to point networks by Dolev-Dwork-Waarts-Yung (JACM 1993), Desmedt-Wang (Eurocrypt 2002), and Srinathan-Narayanan-Rangan (Crypto 2004). Dolev-Dwork-Waarts-Yung gave necessary and sufficient conditions for secure communication in networks with the condition that (1) all the channels are two-way; or (2) all the channels are one-way from the sender to the receiver. In this paper, we study the general case with a network modeled by a directed graph. In this general case, there are communication channels from the sender to the receiver and there are feedback channels from the receiver to the sender. We give necessary and sufficient bounds on the number of channels that are required from sender to receiver given a number of "feedback" channels from receiver to sender. We give these bounds for the case reliability is perfect, as well as for the case it is not perfect.**

***Index Terms*—network security, privacy, reliability, network connectivity.**

I. Introduction

Secure communications guaranteeing reliability and privacy (without unproven assumptions) in networks with active adversaries has been an important research issue. Original work on secure distributed computation assumed a complete graph for secure and reliable communication. Dolev, Dwork, Waarts, and Yung [5] considered secure communication in networks that are not necessarily complete. The trade-off between network connectivity and secure communication has been studied extensively (see, e.g., [1], [2], [4], [5], [11], [19], [13]). For example, Dolev [4] and Dolev et al. [5] showed that, in the case of k Byzantine faults, reliable communication is achievable only if the system's network is $2k+1$ connected. They also showed that if all the paths are one way, then $3k+1$ connectivity is necessary and sufficient for reliable and private communications. However they did not prove any results for the general case when there are certain number of directed paths in one direction and another number of directed paths in the other direction. While undirected graphs correspond naturally to the case of pairwise two-way channels, directed graphs do not correspond to the case of all-one-way or all-two-way channels considered in [5], but to the mixed case where there are some paths in one direction and some paths in the other direction. In this paper, we will initiate the study in this direction by showing what can be done with a general directed graph. Note that this scenario is important in practice, in particular, when the network is not symmetric. For example, a channel from A to B is cheap and a channel from B to A is expensive but not impossible. Another example is that A has access to more resources than B does. Specifically, we will show the following necessary and sufficient result: If u is the number of feedback channels then perfectly secure message transmission from the sender to the receiver is possible if and only if there are $\max\{3k+1-2u, 2k+1\}$ forward channels.

Goldreich, Goldwasser, and Linial [10], Franklin and Yung [8], Franklin and Wright [7], and Wang and Desmedt [20] have studied secure communication and secure computation in *multi-recipient (multicast)* models. In a "multicast channel" (such as Ethernet), one participant can send the same message—simultaneously and privately—to a fixed subset of participants. Franklin and Yung [8] have given a necessary and sufficient condition for individuals to exchange private messages in multicast models in the presence of passive adversaries (passive gossipers). For the case of active Byzantine adversaries, many results have been presented by Franklin and Wright [7], and, Wang and Desmedt [20]. Note that Goldreich, Goldwasser, and Linial [10] have also studied fault-tolerant computation in the public multicast model (which can be thought of as the largest possible multirecipient channels) in the presence of active Byzantine adversaries. Specifically, Goldreich, et al. [10] have made an investigation of general fault-tolerant distributed computation in the full-information model. In the full information model no restrictions are made on the computational power of the faulty parties or the information available to them. (Namely, the faulty players may be infinitely powerful and there are no private channels connecting pairs of honest players). In particular, they present efficient two-party protocols for fault-tolerant computation of any bivariate function.

There are many examples of multicast channels (see, e.g. [7]), such as an Ethernet bus or a token ring. Another example is a shared cryptographic key. By publishing an encrypted message, a participant initiates a multicast to the subset of participants that is able to decrypt it.

We present our model in Section II. In Sections III and IV, we study secure message transmission over directed graphs. Section VI is devoted to reliable message transmission over hypergraphs, and Section VII is devoted to secure message transmission over neighbor networks.

II. Model

We will abstract away the concrete network structures and consider directed graphs. A directed graph is a graph $G(V,E)$ where all edges have directions. In our discussion, we will also assume that the network modelled by $G(V,E)$ is a synchronous network. In another word, there is a time notion that all involved parties could refer to. For a directed graph $G(V,E)$ and two nodes $A, B \in V$, throughout this paper, n denotes the number of vertex disjoint paths between the two nodes and k denotes the number of faults under the control of the adversary. We write $|S|$ to denote the number of elements in the set S. We write $x \in_R S$ to indicate that x is chosen with respect to the uniform distribution on S. Let $\mathbf{F}$ be a finite field, and let $a, b, c, M \in \mathbf{F}$. We define $\text{auth}(M; a, b) := aM + b$ (following [7], [9], [16], [17]) and $\text{auth}(M; a, b, c) := aM^2 + bM + c$ (following [20]). Note that each authentication key $key = (a, b)$ can be used to authenticate one message M without revealing any information about any fixed component of the authentication key and that each authentication key $key = (a, b, c)$ can be used to authenticate two messages M_1 and M_2 without revealing any information about any fixed component of the authentication key. Note that by authenticating a message, we reveal the linear combination of the authentication

An extended abstract of some results in this paper have appeared in [3].

Yongge Wang is with the Department of Software and Information Systems, University of North Carolina at Charlotte, Charlotte, 28223, USA, URL: http://www.sis.uncc.edu/ yonwang. Yvo Desmedt is BT Chair of Information Security, Department of Computer Science, University College London, UK, URL: http://www.cs.ucl.ac.uk/staff/Y.Desmedt/. Yvo Desmedt is partially funded by National Science Foundation CCR-0209092 and EPSRC EP/C538285/1. Part of this research was done while Yvo Desmedt was visiting Certicom, the Univ. of North Carolina Charlotte, and while at Florida State University.

keys though no information about any fixed component of the authentication key is revealed. We will also use a function $\langle \ldots \rangle$ which maps a variable size (we assume that this variable size is bounded by a pre-given bound) ordered subset of $\mathbf{F}$ to an image element in a field extension $\mathbf{F}^*$ of $\mathbf{F}$, and from any image element we can uniquely and efficiently recover the ordered subset.

Let k and n be two integers such that $0 \leq k < n \leq 3k+1$. A $(k+1)$-out-of-n secret sharing scheme is a probabilistic function S: $\mathbf{F} \rightarrow \mathbf{F}^n$ with the property that for any $M \in \mathbf{F}$ and $\mathrm{S}(M) = (v_1, \ldots, v_n)$, no information of M can be inferred from any k entries of $(v_1, \ldots, v_n)$, and M can be recovered from any $k+1$ entries of $(v_1, \ldots, v_n)$. The set of all so obtained possible $(v_1, \ldots, v_n)$ is called a code and its elements codewords [14]. We say that a $(k+1)$-out-of-n secret sharing scheme can detect k' errors if given any codeword $(v_1, \ldots, v_n)$ and any tuple $(u_1, \ldots, u_n)$ over $\mathbf{F}$ such that $0 < |\{i : u_i \neq v_i, 1 \leq i \leq n\}| \leq k'$ one can detect that $(u_1, \ldots, u_n)$ is not a codeword. If the code is Maximal Distance Separable, then the maximum value of errors that can be detected is $n-k-1$ as follows easily from [14]. We say that the $(k+1)$-out-of-n secret sharing scheme can correct k' errors if from any $\mathrm{S}(M) = (v_1, \ldots, v_n)$ and any tuple $(u_1, \ldots, u_n)$ over $\mathbf{F}$ with $|\{i : u_i \neq v_i, 1 \leq i \leq n\}| \leq k'$ one can recover the secret m. If the code is Maximal Distance Separable, then the maximum value of errors that allows the recovery of the vector $(v_1, \ldots, v_n)$ is $\lfloor (n-k-1)/2 \rfloor$ [14]. A $(k+1)$-out-of-n Maximal Distance Separable (MDS) secret sharing scheme is a $(k+1)$-out-of-n secret sharing scheme with the property that for any $k' \leq (n-k-1)/2$, one can correct k' errors and simultaneously detect $n-k-k'-1$ errors (as follows easily by generalizing [14, p. 10]). Maximal Distance Separable (MDS) secret sharing schemes can be constructed from any MDS codes, for example, from Reed-Solomon code [15], e.g., using Shamir secret sharing scheme [18], as basically observed in [15].

In a message transmission protocol, the sender A starts with a message M^A. At the end of the protocol, the receiver B outputs a message M^B. We assume that the message space $\mathcal{M}$ is a subset of a finite field $\mathbf{F}$. We consider two kinds of adversaries. A passive adversary (or gossiper adversary) is an adversary who can only observe the traffic through k internal nodes. An active adversary (or Byzantine adversary) is an adversary with unlimited computational power who can control k internal nodes. That is, an active adversary will not only listen to the traffics through the controlled nodes, but also control the message sent by those controlled nodes. Both kinds of adversaries are assumed to know the complete protocol specification, message space, and the complete structure of the graph. In this paper, we will not consider a dynamic adversary who could change the nodes it controls from round to round, instead we will only consider static adversaries. That is, at the start of the protocol, the adversary chooses the k faulty nodes. An alternative interpretation is that k nodes are static collaborating adversaries.

For any execution of the protocol, let adv be the adversary's view of the entire protocol. We write $adv(M, r)$ to denote the adversary's view when $M^A = M$ and when the sequence of coin flips used by the adversary is r.

Definition 2.1: (see Franklin and Wright [7])

1) Let $\delta < \frac{1}{2}$. A message transmission protocol is δ-reliable if, with probability at least $1-\delta$, B terminates with $M^B = M^A$. The probability is over the choices of M^A and the coin flips of all nodes.
2) A message transmission protocol is reliable if it is 0-reliable.
3) A message transmission protocol is ε-private if, for every two messages M_0, M_1, and for every r, $\sum_c |\Pr[adv(M_0, r) = c] - \Pr[adv(M_1, r) = c]| \leq 2\varepsilon$. The probabilities are taken over the coin flips of the honest parties, and the sum is over all possible values of the adversary's view.
4) A message transmission protocol is perfectly private if it is 0-private.
5) A message transmission protocol is (ε, δ)-secure if it is ε-private and δ-reliable.
6) An (ε, δ)-secure message transmission protocol is efficient if its round complexity and bit complexity are polynomial in the size of the network, $\log \frac{1}{\varepsilon}$ (if $\varepsilon > 0$) and $\log \frac{1}{\delta}$ (if $\delta > 0$).

For two nodes A and B in a directed graph such that there are $2k+1$ node disjoint paths from A to B, there is a straightforward reliable message transmission from A to B against a k-active adversary: A sends the message m to B via all the $2k+1$ paths, and B recovers the message m by a majority vote.

III. $(0, \delta)$-Secure message transmission in directed graphs

Our discussion in this section will be concentrated on directed graphs. Dolev, Dwork, Waarts, and Yung [5] addressed the problem of secure message transmissions in a point-to-point network. In particular, they showed that if all channels from A to B are one-way, then $(3k+1)$-connectivity is necessary and sufficient for (0,0)-secure message transmissions from A to B against a k-active adversary. They also showed that if all channels between A and B are two-way, then $(2k+1)$-connectivity is necessary and sufficient for (0,0)-secure message transmissions between A and B against a k-active adversary. In this section we assume that there are only $2k+1-u$ directed node disjoint paths from A to B, where $1 \leq u \leq k$. We show that u directed node disjoint paths from B to A are necessary and sufficient to achieve $(0, \delta)$-secure message transmissions from A to B against a k-active adversary.

Franklin and Wright [7] showed that even if all channels between A and B are two way, $2k+1$ channels between A and B are still necessary for $(1-\delta)$-reliable (assuming that $\delta < \frac{1}{2}\left(1 - \frac{1}{|\mathbf{F}|}\right)$) message transmission from A to B against a k-active adversary.

Theorem 3.1: (Franklin and Wright [7]) Let $G(V, E)$ be a directed graph, $A, B \in V$, and there are only $2k$ two-way node disjoint paths between A and B in G. Then δ-reliable message transmission from A to B against a k-active adversary is impossible for $\delta < \frac{1}{2}\left(1 - \frac{1}{|\mathbf{F}|}\right)$.

In the following, we first show that if there is no directed path from B to A, then $2k+1$ directed paths from A to B is necessary and sufficient for $(0, \delta)$-secure message transmission from A to B.

Theorem 3.2: Let $G(V, E)$ be a directed graph, $A, B \in V$, and $0 < \delta < \frac{1}{2}$. If there is no directed paths from B to A, then the necessary and sufficient condition for $(0, \delta)$-secure message transmission from A to B against a k-active adversary is that there are $2k+1$ directed node disjoint paths from A to B.

Proof. The necessity is proved in Theorem 3.1. Let $p_1, \ldots, p_{2k+1}$ be the $2k+1$ directed node disjoint paths from A to B. Let $M^A \in \mathbf{F}$ be the secret that A wants to send to B. A constructs $(k+1)$-out-of-$(2k+1)$ secret shares $(s_1^A, \ldots, s_{2k+1}^A)$ of M^A. The protocol proceeds from round 1 through round $2k+1$. In each round $1 \leq i \leq 2k+1$, we have the following steps:

Step 1 A chooses $\{(a_{i,j}^A, b_{i,j}^A) \in_R \mathbf{F}^2 : 1 \leq j \leq 2k+1\}$.

Step 2 A sends $(s_i^A, \mathrm{auth}(s_i^A; a_{i,1}^A, b_{i,1}^A), \ldots, \mathrm{auth}(s_i^A; a_{i,2k+1}^A, b_{i,2k+1}^A))$ to B via path p_i, and sends $(a_{i,j}^A, b_{i,j}^A)$ to B via path p_j for each $1 \leq j \leq 2k+1$.

Step 3 B receives $(s_i^B, c_{i,1}^B, \ldots, c_{i,2k+1}^B)$ via path p_i, and receives $(a_{i,j}^B, b_{i,j}^B)$ via path p_j for each $1 \leq j \leq 2k+1$.

Step 4 B computes $t = |\{j : c_{i,j}^B = \mathrm{auth}(s_i^B; a_{i,j}^B, b_{i,j}^B)\}|$. If $t \geq k+1$, then B decides that s_i^B is a valid share. Otherwise B discards s_i^B.

It is straightforward to see that the adversary will learn at most k shares of the $(k+1)$-out-of-$(2k+1)$ secret sharing scheme. Thus the protocol achieves perfect privacy. Now assume that the path p_i contains no faulty nodes, then B receives the correct share s_i on path p_i during the round i and decides that s_i^B is a valid share. In another word, B receives at least $k+1$ valid shares. For a faulty path p_i, s_i^B may be different from s_i^A. The protocol fails if for some faulty path p_i, $s_i^B \neq s_i^A$ but B decides that s_i^B is a valid share. In order for the adversary to fail the protocol, during round i, the adversary could deliver $(s_i^B, c_{i,1}^B, \ldots, c_{i,2k+1}^B)$ via the faulty path p_i for appropriately chosen (could be randomly chosen) $s_i^B, c_{i,1}^B, \ldots, c_{i,2k+1}^B$. At the same time, the adversary will guarantee that the values $(a_{i,j}^B, b_{i,j}^B)$ received by B on all faulty paths p_j would meet the condition $c_{i,j}^B = \text{auth}(s_i^B; a_{i,j}^B, b_{i,j}^B)$. Since the adversary has no control over non-faulty paths, only with a very small probability, $c_{i,j}^B = \text{auth}(s_i^B; a_{i,j}^B, b_{i,j}^B)$ for a non-faulty path p_j. In another word, B will decide that s_i^B is a valid share only if $c_{i,j}^B = \text{auth}(s_i^B; a_{i,j}^B, b_{i,j}^B)$ for a non-faulty path p_j, which occurs with a very small probability. In order to make the protocol failure probability δ smaller, one could chosen larger finite field $\mathbf{F}$.

From our above discussion, with high probability, B will recover the secret $M^B = M^A$. Thus the above protocol is a $(0,\delta)$-secure message transmission protocol from A to B against a k-active adversary. Q.E.D.

By Theorem 3.1, the necessary condition for $(0,\delta)$-secure message transmission from A to B against a k-active adversary is that there are at least $k+1$ node disjoint paths from A to B and there are at least $2k+1$ node disjoint paths in total from A to B and from B to A. In the following, we show that this condition is also sufficient. We first show that the condition is sufficient for $k=1$.

Theorem 3.3: Let $G(V,E)$ be a directed graph, $A, B \in V$. If there are two directed node disjoint paths p_0 and p_1 from A to B, and one directed path q (which is node disjoint from p_0 and p_1) from B to A, then for any $0 < \delta < \frac{1}{2}$, there is a $(0,\delta)$-secure message transmission protocol from A to B against a 1-active adversary.

Proof. In the following protocol, A $(0,\delta)$-securely transmits a message $M^A \in \mathbf{F}$ to B.

Step 1 A chooses $s_0^A \in_R \mathbf{F}$, $(a_0^A, b_0^A), (a_1^A, b_1^A) \in_R \mathbf{F}^2$, and let $s_1^A = M^A - s_0^A$. For each $i \in \{0,1\}$, A sends $(s_i^A, (a_i^A, b_i^A), \text{auth}(s_i^A; a_{1-i}^A, b_{1-i}^A))$ to B via path p_i.

Step 2 Assume that B receives $(s_i^B, (a_i^B, b_i^B), c_i^B)$ via path p_i. B checks whether $c_i^B = \text{auth}(s_i^B; a_{1-i}^B, b_{1-i}^B)$ for $i = 0, 1$. If both equations hold, then B knows that with high probability the adversary was either passive or not on the paths from A to B. B can recover the secret message, sends "OK" to A via the path q, and terminates the protocol. Otherwise, one of equations does not hold and B knows that the adversary was on one of the paths from A to B. In this case, B chooses $(a^B, b^B) \in_R \mathbf{F}^2$, and sends $((a^B, b^B), (s_0^B, (a_0^B, b_0^B), c_0^B), (s_1^B, (a_1^B, b_1^B), c_1^B))$ to A via the path q.

Step 3 If A receives "OK", then A terminates the protocol. Otherwise, from the information A received via path q, A decides which path from A to B is corrupted and recovers B's authentication key (a^A, b^A). A sends $(M^A, \text{auth}(M^A; a^A, b^A))$ to B via the uncorrupted path from A to B. Note that the adversary may control the path q and A may never receives any message on the path q. If this happens, A can assume that B has received the correct message (thus we are assuming that the network is a synchronous network).

Step 4 B recovers the message and checks that the authenticator is correct.

Similarly as in the proof of Theorem 3.2, it can be shown that the above protocol is $(0,\delta)$-secure against a 1-active adversary. Q.E.D.

Theorem 3.4: Let $G(V,E)$ be a directed graph, $A, B \in V$, and $k \geq u \geq 1$. If there are $2k+1-u$ directed node disjoint paths $p_1, \ldots, p_{2k+1-u}$ from A to B, and u directed node disjoint paths $q_1, \ldots, q_u$ ($q_1, \ldots, q_u$ are node disjoint from $p_1, \ldots, p_{2k+1-u}$) from B to A, then for any $0 < \delta < \frac{1}{2}$, there is an efficient $(0,\delta)$-secure message transmission protocol from A to B against a k-active adversary.

Before we give an efficient $(0,\delta)$-secure message transmission protocol from A to B, we first demonstrate the underlying idea by giving a non-efficient (exponential in k) $(0,\delta)$-secure message transmission protocol from A to B against a k-active adversary. Let $M^A \in \mathbf{F}$ be the secret that A wants to send to B, and $\mathcal{P}_1, \ldots, \mathcal{P}_t$ be an enumeration of size $k+1$ subsets of $\{p_1, \ldots, p_{2k+1-u}, q_1, \ldots, q_u\}$. Since there are at most k-corrupted paths, at least one of the path sets $\mathcal{P}_1, \ldots, \mathcal{P}_t$ contains all honest paths. If we know that some $\mathcal{P}_m$ ($m \leq t$) contains all non-faulty paths, we can let A and B to share a random pair $(\alpha_{i,m}, \beta_{i,m})$ for each path in p_i (or q_i) in $\mathcal{P}_m$. Then we compute $\alpha_m = \sum_i \alpha_{i,m}$, $\beta_m = \sum_i \beta_{i,m}$, and use (α_m, β_m) as the authentication key and α_m as the encryption key to communicate the message from A to B. Note that there is at least one directed path from A to B in $\mathcal{P}_m$.

Since we do not know which path set $\mathcal{P}_m$ contains all non-faulty paths, we have to try all $\mathcal{P}_m$. During our trial on the path set $\mathcal{P}_m$, if the adversary modifies any value during the transmission, the receiver B will notice the modification and will discard the received value through $\mathcal{P}_m$. After the entire trial, the receiver B will receive the message from the non-faulty $\mathcal{P}_m$ (or $\mathcal{P}_m$ is faulty, but the adversary was passive during the run of the protocol). Specifically, the protocol proceeds from round 1 through t. In each round $1 \leq m \leq t$, we have the following steps:

Step 1 For each $p_i \in \mathcal{P}_m$, A chooses $(a_{i,m}^A, b_{i,m}^A, k_{i,m}^A) \in_R \mathbf{F}^2$ and sends $(a_{i,m}^A, b_{i,m}^A, k_{i,m}^A)$ to B via p_i.

Step 2 For each $p_i \in \mathcal{P}_m$, B receives $(a_{i,m}^B, b_{i,m}^B, k_{i,m}^B)$ from A via p_i.

Step 3 For each $q_i \in \mathcal{P}_m$, B chooses $(c_{i,m}^B, d_{i,m}^B, l_{i,m}^B) \in_R \mathbf{F}^2$ and sends $(c_{i,m}^B, d_{i,m}^B, l_{i,m}^B)$ to A via q_i.

Step 4 For each $q_i \in \mathcal{P}_m$, A receives $(c_{i,m}^A, d_{i,m}^A, l_{i,m}^A)$ from B via q_i.

Step 5 A computes $C^A = \sum_{p_i \in \mathcal{P}_m} a_{i,m}^A + \sum_{q_i \in \mathcal{P}_m} c_{i,m}^A$, $D^A = \sum_{p_i \in \mathcal{P}_m} b_{i,m}^A + \sum_{q_i \in \mathcal{P}_m} d_{i,m}^A$, $K^A = \sum_{p_i \in \mathcal{P}_m} k_{i,m}^A + \sum_{q_i \in \mathcal{P}_m} l_{i,m}^A$, and sends $(M^A + K^A, \text{auth}(M^A + K^A; C^A, D^A))$ to B via all paths in p_i in $\mathcal{P}_m$.

Step 6 For each $p_i \in \mathcal{P}_m$, B receives $(e_{i,m}^B, f_{i,m}^B)$ from A via p_i.

Step 7 If $(e_{i,m}^B, f_{i,m}^B) = (e_{j,m}^B, f_{j,m}^B)$ for all $p_i, p_j \in \mathcal{P}_m$, then B goes to Step 8. Otherwise, B goes to round $m+1$.

Step 8 B computes $C^B = \sum_{p_i \in \mathcal{P}_m} a_{i,m}^B + \sum_{q_i \in \mathcal{P}_m} c_{i,m}^B$, $D^B = \sum_{p_i \in \mathcal{P}_m} b_{i,m}^B + \sum_{q_i \in \mathcal{P}_m} d_{i,m}^B$, and $K^B = \sum_{p_i \in \mathcal{P}_m} k_{i,m}^B + \sum_{q_i \in \mathcal{P}_m} l_{i,m}^B$.

Step 9 If $f_{i,m}^B = \text{auth}(e_{i,m}^B; C^B, D^B)$, then B computes the secret $M^B = e_{i,m}^B - K^B$ and terminates the protocol. Otherwise, B goes to round $m+1$.

Since there is at least one path set $\mathcal{P}_m$ such that $\mathcal{P}_m$ contains all non-faulty paths, B accepts a value by the end of the protocol with certainty. It remains to show that if B accepts a value, then with high probability, this value is the same as the value sent by A. Assume that at the end of the protocol, B accepts a value transmitted via the path set $\mathcal{P}_m$ ($m \leq t$). If $\mathcal{P}_m$ contains all non-faulty paths or if the adversary was passive during the protocol run, then obviously B

accepts the correct secret from A. Now assume that $\mathcal{P}_m$ contains at least one faulty path with active adversary. If the adversary was active from Step 1 to Step 4, then A and B shares different authentication key and encryption key. Since there are at most k faulty paths, the adversary learns zero information about the encryption key or any component of the authentication key. Thus the adversary can only let the verifications in Steps 7 and 9 pass with negligible probability. In the same way, if the adversary was active during Steps 5 and 6, the probability that verifications in Steps 7 and 9 pass is negligible.

Proof. (Proof of Theorem 3.4) We have just presented an exponential time $(0, \delta)$-secure message transmission protocol from A to B. In the following, we describe a polynomial time $(0, \delta)$-secure message transmission protocol from A to B. Let $M^A \in \mathbf{F}$ be the secret that A wants to send to B.

Our protocol proceeds from round 1 through $2k + 2 + u$. Different rounds are dedicated to different scenarios. In particular, we distinguish the following two cases:

1) There are $k + 1$ non-faulty paths from A to B.
2) There is at least one honest path from B to A.

In the protocol, A first constructs $(k+1)$-out-of-$(2k+1-u)$ secret shares $(s_1^A, \ldots, s_{2k+1-u}^A)$ of M^A.

For each round $1 \leq i \leq 2k + 1 - u$, A chooses a random authentication key pair $key_{i,j}$ for each path p_j and sends it to B via p_j. A then sends to B, via path p_i, the share s_i^A authenticated with all these authentication keys. If there are at least $k+1$ non-faulty paths from A to B, then at the end of round $2k+1-u$, B can recover at least $k+1$ correct shares and the $(k+1)$-out-of-$(2k+1-u)$ secret sharing scheme enables B to recover the secret M^B. Specifically, for each round $1 \leq i \leq 2k + 1 - u$, we have the following steps:

Step 1 A chooses $\{(a_{i,j}^A, b_{i,j}^A) \in_R \mathbf{F}^2 : 1 \leq j \leq 2k + 1 - u\}$.

Step 2 A sends $\{s_i^A, \text{auth}(s_i^A; a_{i,1}^A, b_{i,1}^A), \ldots, \text{auth}(s_i^A; a_{i,2k+1-u}^A, b_{i,2k+1-u}^A)\}$ to B via path p_i, and sends $(a_{i,j}^A, b_{i,j}^A)$ to B via path p_j for each $1 \leq j \leq 2k + 1 - u$.

Step 3 B receives $\{s_i^B, d_{i,1}^B, \ldots, d_{i,2k+1-u}^B\}$ via path p_i, and $(a_{i,j}^B, b_{i,j}^B)$ via path p_j for each $1 \leq j \leq 2k + 1 - u$.

Step 4 B computes $t = |\{j : d_{i,j}^B = \text{auth}(s_i^B; a_{i,j}^B, b_{i,j}^B)\}|$. If $t \geq k + 1$, then B decides that s_i^B is a valid share. Otherwise B decides that s_i^B is an invalid share.

At the end of round $2k + 1 - u$, if B has received $k + 1$ valid shares, then B recovers the secret M^B from these valid shares and terminates the protocol. If B cannot recover the secret M^B at the end of round $2k+1-u$, then there are less than $k+1$ honest paths from A to B. Thus there is at least one honest path from B to A (note that we have in total $2k + 1$ paths and at most k of them are faulty).

In the following, we describe the remaining rounds of the protocol. In this part of the protocol, with the help of feedback channels from B to A (note that at least one of these channels is uncorrupted), A and B can agree on a shared authentication key and encryption key to communicate the message M^A securely from A to B. In order to use the feedback channels from B to A, A first sends a random 6-tuple value to B via each forward path p_i from A to B. Specifically, the round $2k + 2 - u$ has the following steps:

Step 1 A chooses $\{(a_i^A, b_i^A, c_i^A) \in_R \mathbf{F}^6 : 1 \leq i \leq 2k+1-u\}$, and sends (a_i^A, b_i^A, c_i^A) to B via path p_i for each $i \leq 2k+1-u$. Note that we abuse our notations by letting $a_i^A, b_i^A, c_i^A \in \mathbf{F}^2$ for each i.

Step 2 For each $1 \leq i \leq 2k + 1 - u$, B receives (a_i^B, b_i^B, c_i^B) on path p_i from A (if no value is received on path p_i, B sets it to a default value).

Step 3 For each $1 \leq i \leq 2k + 1 - u$, B chooses $r_i^B \in_R \mathbf{F}$ and computes $\beta^B = \{(r_i^B, \text{auth}(r_i^B; a_i^B, b_i^B, c_i^B)) : 1 \leq i \leq 2k + 1 - u\}$. Note that $a_i^B, b_i^B, c_i^B \in \mathbf{F}^2$, but we can regard each a_i^B, b_i^B, c_i^B as an element of $\mathbf{F}$ by summing the components.

Now B needs to send a random 4-tuple (d_i^B, e_i^B) to A on each feedback channel q_i from B to A. After receiving these random 4-tuples from B, A needs to determine whether the received 4-tuple is the same as the original 4-tuple sent by B (since there is at least one honest feedback channel, at least one of the random 4-tuples received by A is not modified). In order to help A to get some hint on this, the techniques that we have used in round 1 to round $2k+1-u$ will be used again. That is, in order for B to send a random 4-tuple (d_i^B, e_i^B) to A on path q_i $(1 \leq i \leq u)$, B sends random authentication keys to A for each path q_j $(1 \leq j \leq u)$, and sends to A via q_i the random 4-tuple (d_i^B, e_i^B) authenticated with all these authentication keys. This is done by the rounds from $2k + 3 - u$ to $2k + 2$. Specifically, in each round $i+2k+2-u$ $(1 \leq i \leq u)$, we have the following steps:

Step 1 B chooses $(d_i^B, e_i^B) \in_R \mathbf{F}^4$ and $\{(v_{i,j}^B, w_{i,j}^B) \in_R \mathbf{F}^8 : 1 \leq j \leq u\}$.

Step 2 B sends (d_i^B, e_i^B), β^B, and $\{\text{auth}(\langle d_i^B, e_i^B\rangle; v_{i,j}^B, w_{i,j}^B) : 1 \leq j \leq u\}$ to A via path q_i, and $(v_{i,j}^B, w_{i,j}^B)$ to A via path q_j for each $1 \leq j \leq u$.

Step 3 A receives (or substitutes default values) (d_i^A, e_i^A), β_i^A, and $\{\alpha_{i,j}^A : 1 \leq j \leq u\}$ from B via path q_i, and $(v_{i,j}^A, w_{i,j}^A)$ from B via path q_j for each $1 \leq j \leq u$.

According to the values that A has received, A divides the paths set $\{q_1, \ldots, q_u\}$ into consistent subsets $\mathcal{Q}_1, \ldots, \mathcal{Q}_t$ such that for each $1 \leq l \leq t$, all paths in $\mathcal{Q}_l$ behave in a consistent way. In particular, there is at least one path set $\mathcal{Q}_l$ that behaved honestly during the rounds from $2k+3-u$ to $2k+2$ (though A cannot determine which path set was honest, A will try to use each of them in a separate way and let B to determine which path set is honest). The partition of the paths are done according to the following criteria. For any l, m, n with $1 \leq l \leq t$, $1 \leq m, n \leq u$, and $q_m, q_n \in \mathcal{Q}_l$, we have

1) $\beta_m^A = \beta_n^A$;
2) $\alpha_{m,n}^A = \text{auth}(\langle d_m^A, e_m^A\rangle; v_{m,n}^A, w_{m,n}^A)$;
3) $\alpha_{n,m}^A = \text{auth}(\langle d_n^A, e_n^A\rangle; v_{n,m}^A, w_{n,m}^A)$.

For each $\mathcal{Q}_l$, let $q_m \in \mathcal{Q}_l$ and $\beta_m^A = \{(r_{i,l}^A, \gamma_{i,l}^A) : 1 \leq i \leq 2k+1-u\}$. A computes the number

$$t_l = |\{i : \gamma_{i,l}^A = \text{auth}(r_{i,l}^A; a_i^A, b_i^A, c_i^A), 1 \leq i \leq 2k + 1 - u\}| + |\mathcal{Q}_l|$$

If $t_l \leq k$, then A decides that $\mathcal{Q}_l$ is an unacceptable set, otherwise, A decides that $\mathcal{Q}_l$ is an acceptable set. Let $\mathcal{Q}_l = \emptyset$ for $t < l \leq u$. It is straightforward to check that the following holds:

1) If q_i is an honest feedback channel and $q_i \in \mathcal{Q}_l$, then with high probability, the random 4-tuples that A received on the paths from $\mathcal{Q}_l$ are not modified.
2) If q_i is an honest feedback channel and $q_i \in \mathcal{Q}_l$, then A determines that $\mathcal{Q}_l$ is an acceptable set.

However, all acceptable path sets look the same to A and A cannot determine whether an acceptable path set contains all honest paths (or paths controlled by passive adversaries). A continues the protocol by assuming that each acceptable path set is honest. In another word, assuming that an acceptable path set $\mathcal{Q}_l$ is honest, from the values received by A via paths in $\mathcal{Q}_l$, A can determine which of the random 6-tuples (a_i^A, b_i^A, c_i^A) it sent to B during the round $2k + 2 - u$ have been received by B correctly. Using these "correctly-received-by-B" 6-tuples and the random 4-tuples received by A via paths in $\mathcal{Q}_l$, A can compute an authentication key and an encryption key to securely send the messages to B. If the assumption that $\mathcal{Q}_l$ is honest is valid,

then B should be able to compute the same authentication key and the same encryption key. Since at least one of these acceptable path sets is honest, B will be able to decrypt the secret message correctly. Specifically, for each round $2k+2+l$ ($1 \leq l \leq u$), we have the following steps:

Step 1 If $\mathcal{Q}_l = \emptyset$ or $\mathcal{Q}_l$ is an unacceptable set, then go to the next round.

Step 2 A computes $\mathcal{P}_l = \{p_i : \gamma_{i,l}^A = \text{auth}(r_{i,l}^A; a_i^A, b_i^A, c_i^A), 1 \leq i \leq 2k+1-u\}$, $C_l^A = \sum_{p_i \in \mathcal{P}_l} a_i^A + \sum_{q_i \in \mathcal{Q}_l} d_i^A$, $D_l^A = \sum_{p_i \in \mathcal{P}_l} b_i^A + \sum_{q_i \in \mathcal{Q}_l} e_i^A$, and K_l^A be the sum of the two components of C_l^A. Note that $C_l^A, D_l^A \in \mathbf{F}^2$, $K_l^A \in \mathbf{F}$, and $t_l = |\mathcal{P}_l| + |\mathcal{Q}_l| > k$ (thus the adversary learns no information regarding C_l^A, D_l^A).

Step 3 A sends $(\langle \mathcal{Q}_l, \mathcal{P}_l, M^A + K_l^A \rangle, \text{auth}(\langle \mathcal{Q}_l, \mathcal{P}_l, M^A + C_l^A \rangle; C_l^A, D_l^A))$ to B via all paths $p_i \in \mathcal{P}_l$. Without loss of generality, we assume that $\langle \mathcal{Q}_l, \mathcal{P}_l \rangle$ could be represented by an element of $\mathbf{F}$. Thus $\langle \mathcal{Q}_l, \mathcal{P}_l, M^A + C_l^A \rangle$ could be interpreted as an element of $\mathbf{F}^2$.

Step 4 B receives $(\xi_{i,l}^B, \lambda_{i,l}^B)$ from path p_i for $1 \leq i \leq 2k+1-u$.

Step 5 For each $1 \leq i \leq 2k+1-u$, B computes $\langle \mathcal{Q}_{i,l}^B, \mathcal{P}_{i,l}^B, \tau_{i,l}^B \rangle = \xi_{i,l}^B$ (that is, B decomposes $\xi_{i,l}^B$), $C_{i,l}^B = \sum_{p_j \in \mathcal{P}_{i,l}} a_j^B + \sum_{q_j \in \mathcal{Q}_{i,l}} d_j^B$, $D_{i,l}^B = \sum_{p_j \in \mathcal{P}_{i,l}} b_j^B + \sum_{q_j \in \mathcal{Q}_{i,l}} e_j^B$, and $K_{i,l}^B$ as the sum of the two components of $C_{i,l}^B$.

Step 6 For each $1 \leq i \leq 2k+1-u$, B checks whether $\lambda_{i,l}^B = \text{auth}(\xi_{i,l}^B; C_{i,l}^B, D_{i,l}^B)$. If the equation holds, then B computes the secret $M^B = \tau_{i,l}^B - K_{i,l}^B$.

If B has not got the secret at the end of round $2k+1-u$, then there exists an uncorrupted path q_j from B to A and a path set $\mathcal{Q}_l$ such that $q_j \in \mathcal{Q}_l$ and the information that A receives from paths in $\mathcal{Q}_l$ are reliable. Thus, at the end of round $2k+2+u$, B will output a secret M^B. It is easy to check that, with high probability, this secret is the same as M^A.

Since for an acceptable $\mathcal{Q}_l$, $t_l = |\mathcal{P}_l| + |\mathcal{Q}_l| > k$, the adversary learns no information about C_l^A or D_l^A or K_l^A. Thus it is clear that the protocol achieves perfect privacy. Thus it is a $(0, \delta)$-secure message transmission protocol from A to B against a k-active adversary. Q.E.D.

Corollary 3.5: Let $G(V, E)$ be a directed graph, $A, B \in V$, $k \geq u \geq 1$, $\delta < \frac{1}{2}\left(1 - \frac{1}{|\mathbf{F}|}\right)$, and there are $2k+1-u$ directed node disjoint paths $p_1, \ldots, p_{2k+1-u}$ from A to B. Then a necessary and sufficient condition for $(0, \delta)$-secure message transmission protocol from A to B against a k-active adversary is that there are u directed node disjoint paths $q_1, \ldots, q_u$ ($q_1, \ldots, q_u$ are node disjoint from $p_1, \ldots, p_{2k+1-u}$) from B to A.

Proof. This follows from Theorem 3.1 and Theorem 3.4. Q.E.D.

IV. $(0,0)$-Secure message transmission in directed graphs

In the previous section, we addressed probabilistic reliable message transmission in directed graphs. In this section, we consider perfectly reliable message transmission in directed graphs. We will show that if there are u directed node disjoint paths from B to A, then a necessary and sufficient condition for $(0,0)$-secure message transmission from A to B against a k-active adversary is that there are $\max\{3k+1-2u, 2k+1\}$ directed node disjoint paths from A to B.

Theorem 4.1: Let $G(V, E)$ be a directed graph, $A, B \in V$. Assume that there are u directed node disjoint paths from B to A. Then a necessary condition for $(0,0)$-secure message transmission from A to B against a k-active adversary is that there are $\max\{3k+1-2u, 2k+1\}$ directed node disjoint paths from A to B.

Proof. First we note a simple fact that if there are $\max\{3k+1-2u, 2k+1\}$ directed node disjoint paths from A to B, then A can always 0-reliably sends a message M to B by broadcasting the message M via all paths to B (B can recover the message reliably by a majority vote).

If $u = 0$, then by the results in [5], we need $3k+1$ directed node disjoint paths from A to B for $(0,0)$-secure message transmission against a k-active adversary. If $u \geq \lceil \frac{k}{2} \rceil$, then again by the results in [5], we need $2k+1$ directed node disjoint paths from A to B for 0-reliable (that is, perfectly reliable) message transmission from A to B against a k-active adversary. From now on, we assume that $0 < u < \lceil \frac{k}{2} \rceil$.

For a contradiction, we assume that there are only $3k-2u$ directed node disjoint paths from A to B, denoted as $p_1, \ldots, p_{3k-2u}$. Let $q_1, \ldots, q_u$ be the directed node disjoint paths from B to A.

Let Π be a $(0,0)$-secure message transmission protocol from A to B. In the following, we will construct a k-active adversary to defeat this protocol. The transcripts distribution $view_\Pi^A$ of A is drawn from a probability distribution that depends on the message M^A to be transmitted by A, the coin flips R^A of A, the coin flips R^B of B, the coin flips $R^{\mathcal{A}}$ of the adversary (without loss of generality, we assume that the value $R^{\mathcal{A}}$ will determine the choice of faulty paths controlled by the adversary), and the coin flips $R^{\mathcal{T}}$ of all other honest nodes. Without loss of generality, we can assume that the protocol proceeds in steps, where A is silent during even steps and B is silent during odd steps (see [5]).

The strategy of the adversary is as follows. First it uses $R^{\mathcal{A}}$ to choose a value b. If $b = 0$, then it uses $R^{\mathcal{A}}$ again to choose k directed paths $p_{a_1}, \ldots, p_{a_k}$ from A to B and controls the first node on each of these k paths. If $b = 1$, then it uses $R^{\mathcal{A}}$ again to choose $k-u$ directed paths $p_{a_1}, \ldots, p_{a_{k-u}}$ from A to B and controls the first node on each of these $k-u$ paths and the first node on each of the u paths from B to A. It also uses $R^{\mathcal{A}}$ to choose a message $\hat{M}^A \in \mathbf{F}$ according to the same probability distribution from which the actual message M^A was drawn. In the following we describe the protocol the adversary will follow.

- Case $b = 0$. The k paths $p_{a_1}, \ldots, p_{a_k}$ behave as a passive adversary. That is, it proceeds according to the protocol Π.
- Case $b = 1$. The $k-u$ paths $p_{a_1}, \ldots, p_{a_{k-u}}$ ignore what A sends in each step of the protocol and simulates what A would send to B when A sending $\hat{M}^A$ to B. The u paths from B ignore what B sends in each step of the protocol and simulate what B would send to A when $b = 0$.

In the following, we assume that the tuple $(M^A, R^A, R^B, R^{\mathcal{T}}, R^{\mathcal{A}})$ is fixed, $b = 0$, the protocol halts in l steps, and the view of A is $view_\Pi^A(M^A, R^A, R^B, R^{\mathcal{T}}, R^{\mathcal{A}})$. Let $\alpha_{i,j}^A$ be the values that A sends on path p_i in step j and $\vec{\alpha}_i^A = (\alpha_{i,1}^A, \ldots, \alpha_{i,l}^A)$. We can view $\vec{\alpha}_i^A$ as shares of the message M^A. Similarly, let $\alpha_{i,j}^B$ be the values that B receives on path p_i in step j and $\vec{\alpha}_i^B = (\alpha_{i,1}^B, \ldots, \alpha_{i,l}^B)$.

First, it is straightforward to show that for any k paths $p_{a_1}, \ldots, p_{a_k}$ from A to B, there is an $\hat{R}_1^{\mathcal{A}}$ such that $b = 0$, the adversary controls the paths $p_{a_1}, \ldots, p_{a_k}$, and

$$view_\Pi^A(M^A, R^A, R^B, R^{\mathcal{T}}, R^{\mathcal{A}}) = view_\Pi^A(M^A, R^A, R^B, R^{\mathcal{T}}, \hat{R}_1^{\mathcal{A}}) \quad (1)$$

Due to the fact that Π is a perfectly private message transmission protocol, from any k shares from $(\vec{\alpha}_1^A, \vec{\alpha}_2^A, \ldots, \vec{\alpha}_{3k-2u}^A)$ one cannot recover the secret message M^A. Thus $(\vec{\alpha}_1^A, \vec{\alpha}_2^A, \ldots, \vec{\alpha}_{3k-2u}^A)$ is at least a $(k+1)$-out-of-$(3k-2u)$ secret sharing scheme.

Secondly, for any $k-u$ paths $p_{a_1}, \ldots, p_{a_{k-u}}$ from A to B, there is an $\hat{R}_2^{\mathcal{A}}$ such that $b = 1$, $\hat{M}^A \neq M^A$, the adversary controls the

paths $p_{a_1}, \ldots, p_{a_{k-u}}, q_1, \ldots, q_u$, and

$$view_{\Pi}^{A}(M^A, R^A, R^B, R^{\mathcal{T}}, R^{\mathcal{A}}) = view_{\Pi}^{A}(M^A, R^A, R^B, R^{\mathcal{T}}, \hat{R}_2^{\mathcal{A}}) \quad (2)$$

Due to the fact that Π is a perfectly reliable message transmission protocol, any $k-u$ errors in the shares $(\vec{\alpha}_1^B, \vec{\alpha}_2^B, \ldots, \vec{\alpha}_{3k-2u}^B)$ can be corrected by B to recover the secret message M^A.

In summary, $(\vec{\alpha}_1^A, \vec{\alpha}_2^A, \ldots, \vec{\alpha}_{3k-2u}^A)$ is at least a $(k+1)$-out-of-$(3k-2u)$ secret sharing scheme that can correct $k-u$ errors. By the results in [14], we know that the maximum number of errors that a $(k+1)$-out-of-$(3k-2u)$ secret sharing scheme could correct is

$$\left\lfloor \frac{(3k-2u)-k-1}{2} \right\rfloor = \left\lfloor \frac{2k-2u-1}{2} \right\rfloor = k-u-1.$$

This is a contradiction, which concludes the proof. Q.E.D.

The following theorem gives a sufficient condition for $(0,0)$-secure message transmissions.

Theorem 4.2: Let $G(V,E)$ be a directed graph, $A, B \in V$, and $k \geq 2$. If there are $n = \max\{3k+1-2u, 2k+1\}$ directed node disjoint paths $p_1, \ldots, p_n$ from A to B and u directed path $q_1, \ldots, q_u$ from B to A ($q_1, \ldots, q_u$ are node disjoint from $p_1, \ldots, p_n$) then there is an efficient $(0,0)$-secure message transmission protocol from A to B against a k-active adversary.

Proof. If $u = 0$, then $n = 3k+1$ and the results in [5] show that there is a $(0,0)$-secure message transmission protocol from A to B against a k-active adversary. If $k = 0$, then $n \geq 1$. Thus A can send the secret message from any path from A to B and B can reliably and privately recover the message. That is, there is a $(0,0)$-secure message transmission protocol from A to B against a k-active adversary.

Assume that $u > 0$, $k > 0$, and the theorem holds for $u-1$ and $k-1$. We show that the Theorem holds for u and k by induction. In the following we describe a protocol π in which A $(0,0)$-securely transmits $M^A \in_R \mathbf{F}$ to B. There is a detailed case analysis at the end of the protocol. Thus the reader is referred to read the paragraphs at the end of the protocol at the same time when the reader reads the protocol details. In the following protocol, we often say that A reliably sends a value x to B. This means that A broadcasts x to B via all paths from A to B and B recovers this value x using a majority vote.

Step 1 A chooses $R_0^A, R_1^A \in_R \mathbf{F}$ such that $R_0^A + R_1^A = M^A$.

Step 2 A constructs $(k+1)$-out-of-n MDS secret shares $(s_{1,0}^A, ..., s_{n,0}^A)$ of R_0^A. For each $i \leq n$, A sends $s_{i,0}^A$ to B via the path p_i.

Step 3 For each $i \leq n$, B receives (or sets default) $s_{i,0}^B$ on path p_i. B distinguishes the following two cases:

1) There is no error in the received shares. B recovers R_0^B from the received shares, sets FLAG $= 0$, and sends "OK" back to A via all paths q_j for $j \leq u$.
2) There are errors in the received shares. B sets FLAG $= 1$ and sends $(s_{1,0}^B, ..., s_{n,0}^B)$ back to A via all paths q_j for $j \leq u$.

Step 4 For each $j \leq u$, A receives "OK" or $(\bar{s}_{1,j}^A, ..., \bar{s}_{n,j}^A)$ from q_j (if anything else is received, A sets default value). A distinguishes the following two cases:

1) $(\bar{s}_{1,j}^A, ..., \bar{s}_{n,j}^A) = (s_{1,0}^A, ..., s_{n,0}^A)$ for all $j \leq u$ or A received "OK" from all u paths q_j. A reliably sends "OK" to B. A goes to Step 6.
2) All other cases. Let $i_0 \leq n, j_0 \leq u$ be any integers such that $s_{i_0,0}^A \neq \bar{s}_{i_0,j_0}^A$. A reliably sends "path p_{i_0} or q_{j_0} is faulty". A goes to the $(0,0)$-secure message transmission protocol against a $(k-1)$-active adversary on the paths $\{p_i : i \neq i_0\} \cup \{q_j : j \neq j_0\}$ to transmit M^A to B (here we use induction).

Step 5 B distinguishes the following two cases:

1) B reliably receives "OK". B further distinguishes the following two cases:
   1.a) FLAG $= 0$. B goes to Step 6.
   1.b) FLAG $= 1$. B knows that there are at most $k-u$ errors in the shares $(s_{1,0}^B, ..., s_{n,0}^B)$. Thus B recovers R_0^B from these shares by correcting at most $k-u$ errors. B goes to Step 6.
2) B reliably receives "path p_{i_0} or q_{j_0} is faulty". In this case, B goes to the $(0,0)$-secure message transmission protocol against a $(k-1)$-active adversary on the paths $\{p_i : i \neq i_0\} \cup \{q_j : j \neq j_0\}$ and receives the message M^B.

Step 6 A constructs $(k+1)$-out-of-n MDS secret shares $(s_{1,1}^A, ..., s_{n,1}^A)$ of R_1^A. For each $i \leq n$, A sends $s_{i,1}^A$ to B via the path p_i.

Step 7 For each $i \leq n$, B receives (or sets default) $s_{i,1}^B$ on path p_i. B distinguishes the following two cases:

1) There is no error in the received shares. B recovers R_1^B from the received shares, and sends "OK" back to A via all paths q_j for $j \leq u$. B computes the secret message $M^B = R_0^B + R_1^B$ and terminates the protocol.
2) There are errors in the received shares. B further distinguishes the following two cases:
   2.a) FLAG $= 0$. B sends $(s_{1,1}^B, ..., s_{n,1}^B)$ back to A via all paths q_j for $j \leq u$.
   2.b) FLAG $= 1$. B knows that there are at most $k-u$ errors in the shares $(s_{1,1}^B, ..., s_{n,1}^B)$. Thus B recovers R_1^B from these shares by correcting at most $k-u$ errors. B sends "OK" back to A via all paths q_j $(j \leq u)$, computes the secret message $M^B = R_0^B + R_1^B$, and terminates the protocol.

Step 8 For each $j \leq u$, A receives "OK" or $(\bar{s}_{1,j}^A, ..., \bar{s}_{n,j}^A)$ from q_j (if anything else is received, A sets default value). A distinguishes the following two cases:

1) $(\bar{s}_{1,j}^A, ..., \bar{s}_{n,j}^A) = (s_{1,0}^A, ..., s_{n,0}^A)$ for all $j \leq u$ or A received "OK" from all u paths q_j. A reliably sends "OK" to B. A terminates the protocol.
2) All other cases. Let $i_0 \leq n, j_0 \leq u$ be any integers such that $s_{i_0,0}^A \neq \bar{s}_{i_0,j_0}^A$. A reliably sends "path p_{i_0} or q_{j_0} is faulty". A goes to the $(0,0)$-secure message transmission protocol against a $(k-1)$-active adversary on the paths $\{p_i : i \neq i_0\} \cup \{q_j : j \neq j_0\}$ to transmit M^A to B (here we use induction).

Step 9 B distinguishes the following two cases:

1) B reliably receives "OK". In this case, B knows that there are at most $k-u$ errors in the shares $(s_{1,1}^B, \ldots, s_{n,1}^B)$. B recovers R_1^B from these shares by correcting at most $k-u$ errors, computes the secret message $M^B = R_0^B + R_1^B$, and terminates the protocol.
2) B reliably receives "path p_{i_0} or q_{j_0} is faulty". In this case, B goes to the $(0,0)$-secure message transmission protocol against a $(k-1)$-active adversary on the paths $\{p_i : i \neq i_0\} \cup \{q_j : j \neq j_0\}$ and receives the message M^B.

Since $n = \max\{3k+1-2u, 2k+1\}$ and $n - k \geq k+1$, the $(k+1)$-out-of-n MDS secret sharing scheme in Steps 2 and 3 can

be used to detect at most k errors without correcting them. If there is no error in the received shares, B recovers R_0^B correctly in Step 3. If B detects that there are errors in the received shares, B sends back the received shares to A via all the B to A paths in Step 3. If A receives "OK" from all B to A paths in Step 4, then we can distinguish the following two cases:

1) B sends "OK" to A in Step 3. In this case, B has recovered R_0^B correctly in Step 3 and A can send R_1^A to B now. Thus A goes to Step 6.
2) B does not send "OK" to A in Step 3. In this case, all the u paths from B to A are corrupted. Thus there are at most $k-u$ corrupted paths from A to B. Since $n-k-1 = \max\{2k-2u, k\} \geq 2(k-u)$, B can recover the value R_0^B by correcting at most $k-u$ errors if B can get this additional information that there are at most $k-u$ errors in the received shares. Thus A can send "OK" to B and B will learn this information. A can then go to Step 6 to send R_1^A to B.

In Step 4, if A receives from all B to A paths exactly the same shares that it has sent in Step 2. Then we can distinguish the following two cases:

1) B sends "OK" to A in Step 3. Then B has recovered R_0^B and A can go to Step 6 to send R_1^A to B.
2) B detects errors and sends the received shares to A in Step 3. In this case, all the u paths from B to A are corrupted and there are at most $k-u$ corrupted paths from A to B. The similar argument as in the previous case shows that B can recover the value R_0^B by correcting at most $k-u$ errors if B can get this additional information and A can go to Step 6 to send R_1^A to B.

If B receives "OK" in Step 5, then either B has already recovered R_0^B in Step 3 or B has detected errors in Step 5. If it is the latter case, our above analysis shows that there are at most $k-u$ errors in the shares that B has received in Step 3. Thus B can recover R_0^B by correcting at most $k-u$ errors and safely goes to Step 6. If B receives "path p_{i_0} or q_{j_0} is faulty" in Step 5, then B goes to the induction step (even if B has recovered R_0^B in Step 3, B still needs to go to the induction step since A does not know this fact). Note that if we delete the two paths p_{i_0} and q_{j_0}, then we have at most $k-1$ unknown faulty paths, $n-1$ paths from A to B, and $u-1$ paths from B to A. Since

$$\begin{aligned}
&\max\{3(k-1)+1-2(u-1), 2(k-1)+1\} \\
&= \max\{3k-2u, 2k-1\} \\
&\leq \max\{3k-2u, 2k\} \\
&= n-1,
\end{aligned}$$

there is (by induction) a $(0,0)$-secure message transmission protocol from A to B against a $(k-1)$-active adversary on the paths $\{p_i : i \neq i_0\} \cup \{q_j : j \neq j_0\}$, and B can receive the correct message M^B.

Now assume that B has recovered R_0^B and we continue to Step 6. If B detects no error in Step 7, then B can recover R_1^B (thus the secret M^B) safely and terminate the protocol. If B detects errors in Step 7, then we can distinguish the following two cases:

1) FLAG $= 0$. In this case, B has not asked for help in Step 3 and B can ask for help by sending the received shares back to A.
2) FLAG $= 1$. In this case, B has already asked for help in Step 3 and the adversary may have recovered the value of R_0^B. Thus B cannot ask for help any more. However, A only initiates Step 6 if Case 1 of Step 4 happens. Combining with the fact that FLAG $= 1$, we know that all the paths from B to A are corrupted. This means that there are at most $k-u$ errors in the shares that B receives, and B can safely recover R_1^B by correcting at most $k-u$ errors.

Now assume that B asks for help in Step 7 and we come to Steps 8 and 9. The similar arguments as for Steps 4 and 5 show that one of the following two cases happens:

1) A sends "OK" to B, B recovers R_1^B(therefore M^B also).
2) A and B identify two paths so that at least one of them is corrupted, and go to the induction step.

We therefore proved that the protocol π is $(0,0)$-secure against a k-active adversary. Q.E.D.

V. Allowing overlaps between feedback channels and forward channels

In the previous section, we proved a necessary and a sufficient condition for $(0,0)$-secure message transmission from A to B. Our sufficient result requires that the paths from A to B be node disjoint from the paths from B to A. In this section, we show some results that allows overlaps between these paths.

Theorem 5.1: Let $G(V,E)$ be a directed graph, $A, B \in V$ and $k \geq 0$. If there are $n = 2k+1$ directed node disjoint paths $p_1, \ldots, p_n$ from A to B and $k+1$ directed node disjoint paths $q_1, \ldots, q_{k+1}$ from B to A (where $q_1, \ldots, q_{k+1}$ are not necessarily node disjoint from $p_1, \ldots, p_n$) then there is an efficient $(0,0)$-secure message transmission protocol from A to B against a k-active adversary.

Proof. We note that the proof of Theorem 5.1 could not be used here since if we remove (in the induction step) two paths p_i and q_j such that one of them is corrupted, we are not guaranteed that the k-active adversary becomes a $(k-1)$-active adversary (q_j may share a node with some other directed paths from A to B and that node could be corrupted).

First we describe the proof informally. In the protocol, A tries to transmit the secret message to B assuming that one of the directed paths from B to A is not corrupted. This is done by running $k+1$ concurrent sub-protocols in phase one, in each sub-protocol B uses one of the directed paths from B to A to send some feedback information to A. Since there are $k+1$ directed node disjoint paths from B to A and there are at most k corrupted paths, B will be guaranteed to receive the correct secret.

A and B execute the following protocol on the path set $\{p_i : 1 \leq i \leq n\} \cup \{q\}$ for each directed path q from B to A. First A chooses $R_0 \in_R \mathbf{F}$ and sends shares of R_0 to B via the paths $p_1, \ldots, p_n$ using a $(k+1)$-out-of-n MDS secret sharing scheme. If there is no error in the received shares, B recovers R_0. Otherwise B needs help from A and B sends the received shares back to A via the B to A path q. The problem is that: B may receive help even if B has never asked for. However B can detect this. Therefore B always works with A on such a protocol and recovers the correct R_0. Then A sends $R_1 = M^A - R_0$ using a $(k+1)$-out-of-n MDS secret sharing scheme. If there is no error in the received shares of R_1, B has found the secret and can terminate the protocol. If B cannot correct these errors, B needs to continue the protocol. In this situation, B distinguishes the following two cases:

1) B has not asked for help in the transmission of R_0. B can ask for help now and B will then recover the secret M^A.
2) B has asked for help in the transmission of R_0. In this case B cannot ask for help (otherwise the adversary may learn both the values of R_0 and R_1 and thus may recover the secret). The sub-protocol needs to be restarted (that is, A constructs different R_0 and R_1 for M^A and sends them to B again). Each time when A and B restart this sub-protocol, A sends the shares of R_0 and R_1 only via these "non-corrupted" paths from A to B. The "non-corrupted" paths are computed from the feedbacks

that A has received from the path q. If q is not corrupted, then the computation is reliable. However, if q is corrupted, then the computation is unreliable. Since there is at least one non-corrupted path q_{i_0} from B to A, B recovers the secret from the sub-protocol running on the path set $\{p_1, \ldots, p_n\} \cup \{q_{i_0}\}$.

If B asks for help in the transmission of R_0, then both A and B "identify" the corrupted paths from A and B according to the information that B sends to A via the path q. If k' dishonest paths from A to B have been (correctly or incorrectly) identified at the restart of the sub-protocol, A uses a $(k+1)$-out-of-$(2k+1-k')$ MDS secret sharing scheme. This MDS secret sharing scheme will only be used for error detection (or message recovery in the case that no error occurs), thus it can be used to detect $2k+1-k'-k-1 = k-k'$ errors. Due to the fact that this MDS secret sharing scheme cannot detect k errors we need to organize ourselves that B will never use incorrectly identified paths from A to B since otherwise B could compute the incorrect "secret". This is easy to be addressed by having B detect whether the path q from B to A is dishonest or not. This is done by having A reliably send to B what A received via the path q. Since a $(k+1)$-out-of-$(2k+1)$ MDS secret sharing scheme can detect k errors, both A and B identify at least $k' \geq k-u+1$ dishonest paths from A to B in the first run of the sub-protocol. During each following run of the sub-protocol, B will either recover the secret message (when no error occurs) or detect at least one corrupted path from A to B (A could also detect the corrupted path from A to B according to the information A received on the path q). Thus the sub-protocol will be restarted at most k times.

Now we present the entire protocol formally.

Step 1 For each directed path q from B to A, A and B run the sub-protocol between Step 2 and Step 10 (the sub-protocols for the $k+1$ paths could be run in parallel).

Step 2 A sets $\mathrm{AB\_CHANNEL}^A = \{p_1, \ldots, p_n\}$ and $j^A = 0$. B sets $\mathrm{AB\_CHANNEL}^B = \{p_1, \ldots, p_n\}$ and $j^B = 0$.

Step 3 Let $n_j = |\mathrm{AB\_CHANNEL}^A|$. A chooses $R_0 \in_R \mathbf{F}$, and constructs $(k+1)$-out-of-n_j MDS secret shares $\{s_i^A : p_i \in \mathrm{AB\_CHANNEL}^A\}$ of R_0. For each $p_i \in \mathrm{AB\_CHANNEL}^A$, A sends s_i^A to B via the path p_i.

Step 4 For each $p_i \in \mathrm{AB\_CHANNEL}^B$, B receives s_i^B from A via the path p_i. B distinguishes the following two cases:

1) B can recover R_0. B recovers R_0 only if there is no error in the received shares. B sends "ok" to A via the path q.
2) B cannot recover R_0. B sends $\{s_i^B : p_i \in \mathrm{AB\_CHANNEL}^B\}$ to A via the path q.

Step 5 A distinguishes the following two cases:

1) A receives "ok" via the path q. A reliably sends "ok" to B.
2) A receives $\{\bar{s}_i^B : p_i \in \mathrm{AB\_CHANNEL}^A\}$ (or sets default values if the received values are not in valid format). A sets $\mathrm{BAD}^A = \{p_i : \bar{s}_i^B \neq s_i^A\}$ and reliably sends $\{\bar{s}_i^B : p_i \in \mathrm{AB\_CHANNEL}^A\}$ and BAD^A to B. A sets $\mathrm{AB\_CHANNEL}^A = \mathrm{AB\_CHANNEL}^A \setminus \mathrm{BAD}^A$,

Step 6 B distinguishes the following two cases:

1) B reliably receives "ok" from A. If B sent "ok" to A in the Step 4, then goes to Step 7. Otherwise, B terminates this sub-protocol.
2) B reliably receives $\{\bar{s}_i^B : p_i \in \mathrm{AB\_CHANNEL}^B\}$ and BAD^B from A. If $\bar{s}_i^B = s_i^B$ for all $p_i \in \mathrm{AB\_CHANNEL}^B$, then B sets $\mathrm{AB\_CHANNEL}^B = \mathrm{AB\_CHANNEL}^B \setminus \mathrm{BAD}^B$, recovers R_0 from $\{s_i^B : p_i \in \mathrm{AB\_CHANNEL}^B\}$, and goes to Step 7. Otherwise, B terminates this sub-protocol.

Step 7 Let $n_j = |\mathrm{AB\_CHANNEL}^A|$. A constructs $(k+1)$-out-of-n_j MDS secret shares $\{s_i^A : p_i \in \mathrm{AB\_CHANNEL}^A\}$ of $R_1 = M^A - R_0$. For each $p_i \in \mathrm{AB\_CHANNEL}^A$, A sends s_i^A to B via the path p_i.

Step 8 For each $p_i \in \mathrm{AB\_CHANNEL}^B$, B receives s_i^B from A via the path p_i. B distinguishes the following two cases:

1) B can recover R_1. B recovers R_1 only if there is no error in the received shares. B sends "ok" to A via the path q.
2) B cannot recover R_1. For this situation we need to distinguish two cases:
   2.a) B sent "ok" to A in Step 4. That is, B has not asked for help to recover R_0. Then B can ask for help now. B sends $\{s_i^B : p_i \in \mathrm{AB\_CHANNEL}^B\}$ to A via the path q.
   2.b) B sent the received shares to A in Step 4. That is, B has asked for help to recover R_0. Then B cannot ask for help now. B sends "continue to the next round" to A via the path q.

Step 9 A distinguishes the following three cases:

1) A receives "ok" via the path q. A reliably sends "ok" to B.
2) A receives "continue to the next round" via the path q. A sets $j^A = j^A + 1$, reliably sends "continue to the next round" to B, and goes to Step 3.
3) A receives $\{\bar{s}_i^B : p_i \in \mathrm{AB\_CHANNEL}^A\}$ (or sets default values if the received values are in invalid format). A sets $\mathrm{BAD}^A = \{p_i : \bar{s}_i^B \neq s_i^A\}$, $\mathrm{AB\_CHANNEL}^A = \mathrm{AB\_CHANNEL}^A \setminus \mathrm{BAD}^A$, and reliably sends $\{\bar{s}_i^B : p_i \in \mathrm{AB\_CHANNEL}^A\}$ and BAD^A to B.

Step 10 B distinguishes the following three cases:

1) B reliably receives "ok" from A. If B sent "ok" to A in the Step 8, then B has recovered the secret. B terminates the entire protocol. Otherwise, B terminates this sub-protocol.
2) B reliably receives "continue to the next round". If B sent "continue to the next round" to A in the Step 8, then B sets $j^B = j^B + 1$ and goes to Step 3. Otherwise, B terminates this sub-protocol.
3) B reliably receives $\{\bar{s}_i^B : p_i \in \mathrm{AB\_CHANNEL}^B\}$ and BAD^B from A. If $\bar{s}_i^B = s_i^B$ for all $p_i \in \mathrm{AB\_CHANNEL}^B$, then B sets $\mathrm{AB\_CHANNEL}^B = \mathrm{AB\_CHANNEL}^B \setminus \mathrm{BAD}^B$, recovers R_1 from $\{s_i^B : p_i \in \mathrm{AB\_CHANNEL}^B\}$, computes the secret $M^B = R_0 + R_1$, and terminates the entire protocol. Otherwise, B terminates this sub-protocol.

It is straightforward to show that at the beginning of each run of the sub-protocol between Step 2 and Step 10, Both A and B have the same sets of AB_CHANNEL, that is, $\mathrm{AB\_CHANNEL}^A = \mathrm{AB\_CHANNEL}^B$ at Step 2. From the analysis before the above protocol, it is straightforward that the above protocol is a $(0,0)$-secure message transmission protocol against a k-active adversary. Q.E.D.

VI. Secure message transmissions in hypergraphs

Applications of hypergraphs in secure communications have been studied by Franklin and Yung in [8]. A hypergraph H is a pair (V, E)

where V is the node set and E is the hyperedge set. Each hyperedge $e \in E$ is a pair (A, A^*) where $A \in V$ and A^* is a subset of V. In a hypergraph, we assume that any message sent by a node A will be received identically by all nodes in A^*, whether or not A is faulty, and all parties outside of A^* learn nothing about the content of the message.

Let $A, B \in V$ be two nodes of the hypergraph $H(V, E)$. We say that there is a "*direct link*" from node A to node B if there exists a hyperedge (A, A^*) such that $B \in A^*$. We say that there is an "*undirected link*" from A to B if there is a directed link from A to B or a directed link from B to A. If there is a directed (undirected) link from A_i to A_{i+1} for every i, $0 \leq i < k$, then we say that there is a "*directed path*" ("*undirected path*") from A_0 to A_k. A and B are "*strongly k-connected*" ("*weakly k-connected*") in the hypergraph $H(V, E)$ if for all $S \subset V \setminus \{A, B\}$, $|S| < k$, there remains a directed (undirected) path from A to B after the removal of S and all hyperedges (X, X^*) such that $S \cap (X^* \cup \{X\}) \neq \emptyset$. Franklin and Yung [8] showed that reliable and private communication from A to B is possible against a *k-passive* adversary if and only if A and B are strongly 1-connected and weakly $(k+1)$-connected. It should be noted that B and A are strongly k-connected does not necessarily mean that A and B are strongly k-connected.

Following Franklin and Yung [8], and, Franklin and Wright [7], we consider multicast as our only communication primitive in this section. A message that is multicast by any node A in a hypergraph is received by all nodes A^* with privacy (that is, nodes not in A^* learn nothing about what was sent) and authentication (that is, nodes in A^* are guaranteed to receive the value that was multicast and to know which node multicast it). We assume that all nodes in the hypergraph know the complete protocol specification and the complete structure of the hypergraph.

Definition 6.1: Let $H(V, E)$ be a hypergraph, $A, B \in V$ be distinct nodes of H, and $k \geq 0$. A, B are *k-separable* in H if there is a node set $W \subset V$ with at most k nodes such that any directed path from A to B goes through at least one node in W. We say that W *separates* A, B.

Remark. Note that there is no straightforward relationship between strong connectivity and separability in hypergraphs.

Theorem 6.2: Let $H(V, E)$ be a hypergraph, $A, B \in V$ be distinct nodes of H, and $k \geq 0$. The nodes A and B are not $2k$-separable if and only if there are $2k + 1$ directed node disjoint paths from A to B in H.

Proof. This follows directly from the maximum-flow minimum-cut theorem in classical graph theory. For details, see, e.g., [6]. Q.E.D.

Theorem 6.3: Let $H(V, E)$ be a hypergraph, $A, B \in V$ be distinct nodes of H, and $k \geq 0$. A necessary and sufficient condition for reliable message transmission from A to B against a k-active adversary is that A and B are not $2k$-separable in H.

Proof. First assume that A and B cannot be separated by a $2k$-node set. By Theorem 6.2, there are $2k + 1$ directed node disjoint paths from A to B in H. Thus reliable message transmission from A to B is possible.

Next assume that A and B can be separated by a $2k$-node set W in H. We shall show that reliable message transmission is impossible. Suppose that π is a message transmission protocol from A to B and let $W = W_0 \cup W_1$ be a $2k$-node separation of A and B with W_0 and W_1 each having at most k nodes. Let m_0 be the message that A transmits. The adversary will attempt to maintain a simulation of the possible behavior of A by executing π for message $m_1 \neq m_0$. The strategy of the adversary is to flip a coin and then, depending on the outcome, decide which set of W_0 or W_1 to control. Let W_b be the chosen set. In each execution step of the transmission protocol, the adversary causes each node in W_b to follow the protocol π as if the protocol were transmitting the message m_1. This simulation succeeds with nonzero probability. Since B does not know whether $b = 0$ or $b = 1$, at the end of the protocol B cannot decide whether A has transmitted m_0 or m_1 if the adversary succeeds. Thus with nonzero probability, the reliability is not achieved. Q.E.D.

Theorem 6.3 gives a sufficient and necessary condition for achieving reliable message transmission against a k-active adversary over hypergraphs. In the following example, we show that this condition is not sufficient for achieving privacy against a k-active adversary (indeed, even not for a k-passive adversary).

Example 1: Let $H(V, E_h)$ be the hypergraph in Figure 1 where $V = \{A, B, v_1, v_2, v, u_1, u_2\}$ and $E_h = \{(A, \{v_1, v_2\}), (v_1, \{v, B\}), (v_2, \{v, B\}), (A, \{u_1, u_2\}), (u_1, \{v, B\}), (u_2, \{v, B\})\}$. Then the nodes A and B are not 2-separable in H. Theorem 6.3 shows that reliable message transmission from A to B is possible against a 1-active adversary. However, the hypergraph H is not weakly 2-connected (the removal of the node v and the removal of the corresponding hyperedges will disconnect A and B). Thus, the result by Franklin and Yung [8] shows that private message transmission from A to B is not possible against a 1-passive adversary.

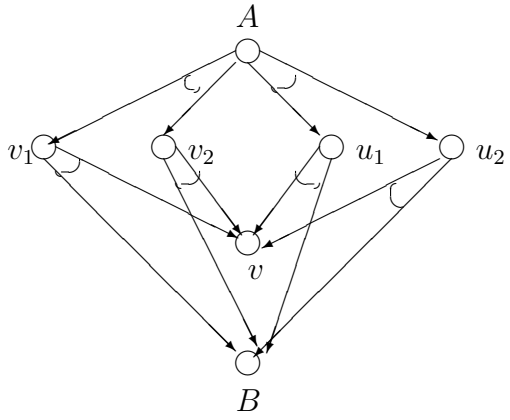

Fig. 1. The hypergraph $H(V, E_h)$ in Example 1

Theorem 6.4: Let $\delta > 0$ and A and B be two nodes in a hypergraph $H(V, E)$ satisfying the following conditions:

1) A and B are not $2k$-separable in H.
2) B and A are not $2k$-separable in H.
3) A and B are strongly k-connected in H.

Then there is a $(0, \delta)$-secure message transmission protocol from A to B against a k-active adversary.

Proof. Assume that the conditions of the theorem is satisfied. For each k-node subset set S of $V \setminus \{A, B\}$, let p_S be a directed path from A to B which witnesses that A and B are strongly k-connected by removing the nodes in S and corresponding hyperedges in H. Let $\mathcal{S} = \{S : S \subset V \setminus \{A, B\}, |S| = k\}$ and $\mathcal{P} = \{p_S : S \in \mathcal{S}\}$. Then A transmits the message M^A to B using the following protocol.

Step 1 For each $S \in \mathcal{S}$, A chooses a random pair $(a_S, b_S) \in_R \mathbf{F}^2$, and transmits this pair to B via the path p_S.

Step 2 For each $S \in \mathcal{S}$, B receives a pair (a_S^B, b_S^B) from A via the path p_S.

Step 3 For each $S \in \mathcal{S}$, B chooses a random $r_S \in_R \mathbf{F}$ and computes $s_S = \text{auth}(r_S; a_S^B, b_S^B)$.

Step 4 B reliably transmits $s = \langle \langle r_S, s_S \rangle : S \in \mathcal{S} \rangle$ to A.

Step 5 A reliably receives the value $s = \langle \langle r_S, s_S \rangle : S \in \mathcal{S} \rangle$ from B.

Step 6 A computes the key index set $K_{\text{index}} = \{i_S : s_S = \text{auth}(r_S; a_S^A, b_S^A)\}$ and the shared secret $K^A = \sum_{i_S \in K_{\text{index}}} a_S^A$.

Step 7 A reliably transmits $\langle K_{\text{index}}, M^A + K^A \rangle$ to B, where M^A is the secret message.

Step 8 B reliably receives the value $\langle K_{\text{index}}, c \rangle$ from A. B computes the secret $K^B = \sum_{i_S \in K_{\text{index}}} a_S^B$, and decrypts the message $M^B = c - K^B$.

It is possible that $a_S^A \neq a_S^B$ but $\text{auth}(r_S; a_S^A, b_S^A) = \text{auth}(r_S; a_S^B, b_S^B)$ for some $S \in \mathcal{S}$. However this probability is negligible. Thus the above protocol is reliable with high probability. Since A and B are strongly k-connected in H, there is a pair (a_S, b_S) such that (a_S, b_S) reliably reaches B and the adversary cannot infer any information of a_S from its view. Thus the above protocol is $(0, \delta)$-secure against a k-active adversary if one chooses sufficiently large **F**. Q.E.D.

The results in Sections III and IV show that the condition in Theorem 6.4 is not necessary. For example, for a graph $G(V, E)$ with $2k$ directed node disjoint paths from A to B, and 1 directed node disjoint path from B to A, A and B are $2k$ separable in $G(V, E)$. But according to Theorem 3.4, there is a $(0, \delta)$-secure message transmission protocol from A to B against a k-active adversary.

VII. Secure message transmission over neighbor networks

A. Definitions

A special case of the hypergraph is the *neighbor networks*. A neighbor network is a graph $G(V, E)$. In a neighbor network, a node $A \in V$ is called a neighbor of another node $B \in V$ if there is an edge $(A, B) \in E$. In a neighbor network, we assume that any message sent by a node A will be received identically by all its neighbors, whether or not A is faulty, and all parties except for A's neighbors learn nothing about the content of the message.

For a neighbor network $G(V, E)$ and two nodes A, B in it, Franklin and Wright [7], and, Wang and Desmedt [20] showed that if there are n multicast lines (that is, n paths with disjoint neighborhoods) between A and B and there are at most k malicious (Byzantine style) processors, then the condition $n > k$ is necessary and sufficient for achieving efficient probabilistically reliable and perfect private communication.

For each neighbor network $G(V, E)$, there is a hypergraph $H_G(V, E_h)$ which is equivalent to $G(V, E)$ in functionality. $H_G(V, E_h)$ is defined by letting E_h be the set of hyperedges (A, A^*) where $A \in V$ and A^* is the set of neighbors of A.

Let A and B be two nodes in a neighbor network $G(V, E)$. We have the following definitions:

1) A and B are *k-connected* in $G(V, E)$ if there are k node disjoint paths between A and B in $G(V, E)$.
2) A and B are *weakly k-hyper-connected* in $G(V, E)$ if A and B are weakly k-connected in $H_G(V, E_h)$.
3) A and B are *k-neighbor-connected* in $G(V, E)$ if for any set $V_1 \subseteq V \setminus \{A, B\}$ with $|V_1| < k$, the removal of $neighbor(V_1)$ and all incident edges from $G(V, E)$ does not disconnect A and B, where

$$neighbor(V_1) = V_1 \cup \{A \in V : \exists B \in V_1 \text{ such that } (B, A) \in E\} \setminus \{A, B\}.$$

4) A and B are *weakly (n, k)-connected* if there are n node disjoint paths $p_1, \ldots, p_n$ between A and B and, for any node set $T \subseteq (V \setminus \{A, B\})$ with $|T| \leq k$, there exists $1 \leq i \leq n$ such that all nodes on p_i have no neighbor in T.

It is easy to check that the following relationships hold.

weak $(n, k-1)$-connectivity $(n \geq k) \Rightarrow$ k-neighbor-connectivity $\Rightarrow$ weak k-hyper-connectivity $\Rightarrow$ k-connectivity

In the following examples, we show that these implications are strict.

Example 2: Let $G(V, E)$ be the graph in Figure 2 where $V = \{A, B, C, D\}$ and $E = \{(A, C), (C, B), (A, D), (D, B), (C, D)\}$. Then it is straightforward to check that $G(V, E)$ is 2-connected but not weakly 2-hyper-connected.

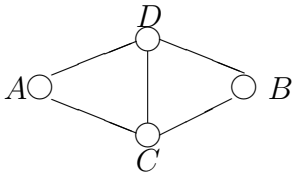

Fig. 2. The graph $G(V, E)$ in Example 2

Example 3: Let $G(V, E)$ be the graph in Figure 3 where $V = \{A, B, C, D, F\}$ and $E = \{(A, C), (A, D), (C, B), (D, B), (C, F), (F, D)\}$. Then it is straightforward to check that A and B are weakly 2-hyper-connected but not 2-neighbor-connected.

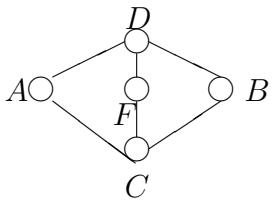

Fig. 3. The graph $G(V, E)$ in Example 3

Example 4: Let $G(V, E)$ be the graph in Figure 4 where $V = \{A, B, C, D, E, F, G, H\}$ and $E = \{(A, C), (C, D), (D, E) (E, B), (A, F), (F, G), (G, H) (H, B), (C, H), (E, F)\}$. Then it is straightforward to check that A and B are 2-neighbor-connected but not weakly $(2, 1)$-connected.

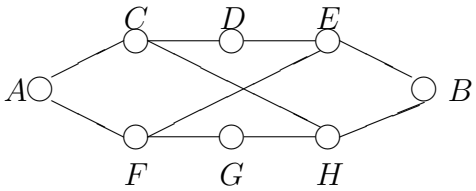

Fig. 4. The graph $G(V, E)$ in Example 4

Example 2 shows that k-connectivity does not necessarily imply weak k-hyper-connectivity. Example 3 shows that weak k-hyper-connectivity does not necessarily imply k-neighbor-connectivity. Example 4 shows that k-neighbor connectivity does not necessarily imply weak $(n, k-1)$-connectivity for some $n \geq k$.

B. $(0, \delta)$-Secure message transmission over neighbor networks

Wang and Desmedt [20] have given a sufficient condition for achieving $(0, \delta)$-security message transmission against a k-active adversary over neighbor networks. In this section, we show that their condition is not necessary.

Theorem 7.1: (Wang and Desmedt [20]) If A and B are *weakly (n, k)-connected* for some $k < n$, then there is an efficient $(0, \delta)$-secure message transmission between A and B.

The condition in Theorem 7.1 is not necessary. For example, the neighbor network G in Example 3 is not 2-neighbor-connected, thus

not weakly $(2,1)$-connected. In the following we present a $(0,\delta)$-secure message transmission protocol against a 1-active adversary from A to B for the neighbor network of Example 3. In the protocol, we will often say that a node X sends a value x to its neighbor node Y. This could be achieved in a neighbor network by the following procedures:

1) Y broadcasts a value y to all its neighbors.
2) X broadcasts $x+y$ to all of its neighbors.
3) Y recovers the message x using the value y.

In this procedure, Y will receive the value x secretly only if X and Y have no common neighbors (or they have no collaborating faulty neighbors).

Message transmission protocol for neighbor network G in Example 3.

Step 1 A chooses two random pairs $(r_1^A, r_2^A) \in_R \mathbf{F}^2$ and $(r_3^A, r_4^A) \in_R \mathbf{F}^2$. A sends (r_1^A, r_2^A) to C and (r_3^A, r_4^A) to D.

Step 2 B chooses two random pairs $(r_1^B, r_2^B) \in_R \mathbf{F}^2$ and $(r_3^B, r_4^B) \in_R \mathbf{F}^2$. B sends (r_1^B, r_2^B) to C and (r_3^B, r_4^B) to D.

Step 3 C chooses a random pair $(a_1, b_1) \in_R \mathbf{F}^2$. C sends $(a_1 + r_1^A, b_1 + r_2^A)$ to A and $(a_1 + r_1^B, b_1 + r_2^B)$ to B.

Step 4 D chooses a random pair $(a_2, b_2) \in_R \mathbf{F}^2$. D sends $(a_2 + r_3^A, b_2 + r_4^A)$ to A and $(a_2 + r_3^B, b_2 + r_4^B)$ to B.

Step 5 From the messages received from C and D, A computes (a_1^A, b_1^A) and (a_2^A, b_2^A).

Step 6 From the messages received from C and D, B computes (a_1^B, b_1^B) and (a_2^B, b_2^B).

Step 7 B chooses a random $r \in_R \mathbf{F}$, computes $s_1 = \text{auth}(r; a_1^B, b_1^B)$ and $s_2 = \text{auth}(r; a_2^B, b_2^B)$. Using the probabilistically reliable message transmission protocol of Franklin and Wright [7], B transmits $\langle r, s_1, s_2 \rangle$ to A.

Step 8 Let $\langle r^A, s_1^A, s_2^A \rangle$ be the message received by A in the last step, A computes the key index set $K_{\text{index}} = \{i : s_i^A = \text{auth}(r^A; a_i^A, b_i^A)\}$. A also computes the shared secret $K^A = \sum_{i \in K_{\text{index}}} a_i^A$.

Step 9 Using the probabilistically reliable message transmission protocol of Franklin and Wright [7], A transmits $\langle K_{\text{index}}, M^A + K^A \rangle$ to B, where M^A is the secret message.

Step 10 Let $\langle K_{\text{index}}^B, c^B \rangle$ be the message that B received in the last step. B computes the shared secret $K^B = \sum_{i \in K_{\text{index}}^B} a_i^B$, and decrypts the message $M^B = c^B - K^B$.

It is straightforward to check that the above protocol is an efficient $(0,\delta)$-secure message transmission protocol from A to B against a 1-active adversary.

Example 1 shows that for a general hypergraph, the existence of a reliable message transmission protocol does not imply the existence of a private message transmission protocol. We show that this is true for probabilistic reliability and perfect privacy in neighbor networks also.

Example 5: Let $G(V,E)$ be the neighbor network in Figure 5 where $V = \{A,B,C,D,E,F,G\}$ and $E = \{(A,C),(C,D),(D,B),(A,E),(E,F),(F,B),(G,C),(G,D),(G,E),(G,F)\}$. Then there is a probabilistic reliable message transmission protocol from A to B against a 1-active adversary in G. But there is no private message transmission from A to B against a 1-passive (or 1-active) adversary in G.

Proof. It is straightforward to check that $G(V,E)$ is not weakly 2-hyper-connected. Indeed, in the hypergraph $H_G(V,E_h)$ of $G(V,E)$, the removal of node G and the removal of the corresponding hyperedges will disconnect A and B completely. Thus Franklin and Yung's result in [8] shows that there is no private message transmission protocol against a 1-passive (or 1-active) adversary from A to B. It is also straightforward to check that Franklin and Wright's [7] reliable message transmission protocol against a 1-active adversary works for the two paths (A,C,D,B) and (A,E,F,B). Q.E.D.

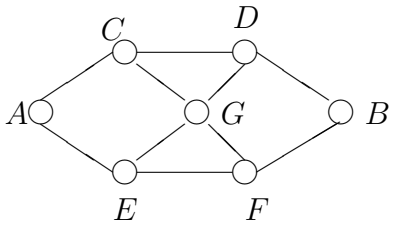

Fig. 5. The graph $G(V,E)$ in Example 5

Though weak k-hyper-connectivity is a necessary condition for achieving probabilistically reliable and perfectly private message transmission against a $(k-1)$-active adversary, we do not know whether this condition is sufficient. We conjecture that there is no probabilistically reliable and perfectly private message transmission protocol against a 1-active adversary for the weakly 2-hyper-connected neighbor network $G(V,E)$ in Figure 6, where $V = \{A, B, C, D, E, F, G, H\}$ and $E = \{(A,C), (C,D), (D,E), (E,B), (A,F), (F,G), (G,H), (H,B), (D,G)\}$. Note that in order to prove or refute our conjecture, it is sufficient to show whether there is a probabilistically reliable message transmission protocol against a 1-active adversary for the neighbor network. For this specific neighbor network, the trick in our previous protocol could be used to convert any probabilistically reliable message transmission protocol to a probabilistically reliable and perfectly private message transmission protocol against a 1-active adversary.

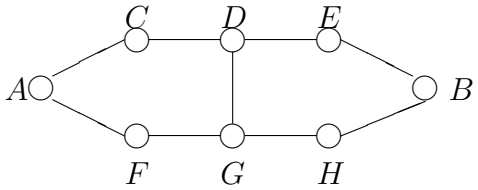

Fig. 6. The graph $G(V,E)$

ACKNOWLEDGEMENT

The authors would like to thank the anonymous referees for comments on improving the presentation of this paper.

REFERENCES

[1] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computing. In: *Proc. ACM STOC, '88*, pages 1–10, ACM Press, 1988.

[2] D. Chaum, C. Crepeau, and I. Damgard. Multiparty unconditional secure protocols. In: *Proc. ACM STOC '88*, pages 11–19, ACM Press, 1988.

[3] Y. Desmedt and Y. Wang. Perfectly secure message transmission revisited. In: *Proc. Eurocrypt '02*, pages 502–517, Lecture Notes in Computer Science 2332, Springer-Verlag, 2002.

[4] D. Dolev. The Byzantine generals strike again. *J. of Algorithms*, **3**:14–30, 1982.

[5] D. Dolev, C. Dwork, O. Waarts, and M. Yung. Perfectly secure message transmission. *J. of the ACM*, **40**(1):17–47, 1993.

[6] L.R. Ford and D. R. Fulkerson. *Flows in Networks*. Princeton University Press, Princeton, NJ, 1962.

[7] M. Franklin and R. Wright. Secure communication in minimal connectivity models. *Journal of Cryptology*, **13**(1):9–30, 2000.
[8] M. Franklin and M. Yung. Secure hypergraphs: privacy from partial broadcast. In: *Proc. ACM STOC '95*, pages 36–44, ACM Press, 1995.
[9] E. Gilbert, F. MacWilliams, and N. Sloane. Codes which detect deception. *The BELL System Technical Journal*, **53**(3):405–424, 1974.
[10] O. Goldreich, S. Goldwasser, and N. Linial. Fault-tolerant computation in the full information model. *SIAM J. Comput.* **27**(2):506–544, 1998.
[11] V. Hadzilacos. *Issues of Fault Tolerance in Concurrent Computations*. PhD thesis, Harvard University, Cambridge, MA, 1984.
[12] W. Jackson and K. Martin. Combinatorial models for perfect secret sharing schemes. *Journal of Comb. Mathematics and Comb. Computing* **28**:249–265, 1998.
[13] M. Kumar, P. Goundan, K. Srinathan, and C. Rangan. On perfectly secure communication over arbitrary networks. In: *Proc. 21st ACM PODC*, pages 193–202, 2002.
[14] F. J. MacWilliams and N. J. A. Sloane. *The theory of error-correcting codes*. North-Holland Publishing Company, 1978.
[15] R. J. McEliece and D. V. Sarwate. On sharing secrets and Reed-Solomon codes. *Comm. ACM*, **24**(9):583–584, September 1981.
[16] T. Rabin. Robust sharing of secrets when the dealer is honest or faulty. *J. of the ACM*, **41**(6):1089–1109, 1994.
[17] T. Rabin and M. Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority. In: *Proc. ACM STOC '89*, pages 73–85, ACM Press, 1989.
[18] A. Shamir. How to share a secret. *Commun. ACM*, **22**:612–613, November 1979.
[19] K. Srinathan, A. Narayanan, and C. Rangan. Optimal Perfectly Secure Message Transmission. In: Proc. Crypto '04, pages 545–561, 2004.
[20] Y. Wang and Y. Desmedt. Secure communication in multicast channels: the answer to Franklin and Wright's question. *J. of Cryptology*, **14**(2):121–135, 2001.

Yongge Wang Dr. Yongge Wang received his PhD degree from University of Heidelberg of Germany. Since then, Dr. Wang has worked in the industry for a few years until he joined UNC Charlotte in 2002. In particular, Dr. Wang has worked in Certicom as a cryptographic mathematician specializing in efficient cryptographic techniques for wireless communications. Dr. Wang has been actively participated in and contributed to the standards bodies such as IETF, W3C XML Security protocols, IEEE 1363 standardization groups for cryptographic techniques, and ANSI T11 groups for SAN network security standards. Dr. Wang is the inventor of Remote Password Authentication protocols SRP5 which is an IEEE 1363.2 standard. Dr. Wang has also worked with Cisco researchers and American Gas Association researchers to design security protocols for the SCADA industry.

Yvo Desmedt Dr. Yvo Desmed received his Ph.D. (Summa cum Laude) from the University of Leuven, Belgium (1984). He is presently the BT Chair of Information Security at University College London, UK. He is also a courtesy professor at Florida State University. His interests include cryptography, network security and computer security. He was (co-)program chair of ICITS 2007, CANS 2005, PKC 2003, the 2002 ACM Workshop on Scientific Aspects of Cyber Terrorism and Crypto '94. He is editor-in-chief of the IEE Proceedings of Information Security, editor of the Journal of Computer Security, of Information Processing Letters and of Advances in Mathematics of Communications. He has given invited lectures at several conferences and workshop in 5 different continents. He has authored over 150 refereed papers.