Group Secret Key Generation Algorithms

Chunxuan Ye and Alex Reznik
InterDigital Communications Corporation
King of Prussia, PA 19406
Email: {Chunxuan.Ye, Alex.Reznik}@interdigital.com

***Abstract*— We consider a pair-wise independent network where every pair of terminals in the network observes a common pair-wise source that is independent of all the sources accessible to the other pairs. We propose a method for secret key agreement in such a network that is based on well-established point-to-point techniques and repeated application of the one-time pad. Three specific problems are investigated. 1) Each terminal's observations are correlated only with the observations of a central terminal. All these terminals wish to generate a common secret key. 2) In a pair-wise independent network, two designated terminals wish to generate a secret key with the help of other terminals. 3) All the terminals in a pair-wise independent network wish to generate a common secret key. A separate protocol for each of these problems is proposed. Furthermore, we show that the protocols for the first two problems are optimal and the protocol for the third problem is efficient, in terms of the resulting secret key rates.**

I. Introduction

The problem of secret key generation by two terminals, based on their respective observations of a common source followed by public transmissions between them, was first studied by Maurer [4], and Ahlswede and Csiszár [1]. Various extensions of this problem have been investigated since then (see, e.g., [2], [5], [8], [9], [10]).

Csiszár and Narayan [3] generalize the secret key generation problem to multiple terminals. They consider a model with an arbitrary number of terminals, each with distinct observations of a common source. A group of terminals wish to generate a secret key with the help of other terminals. In generating such a key, these terminals are allowed to communicate with each other through a noiseless public channel.

In this paper, we consider a pair-wise independent network where every pair of terminals in the network observes a common source that is independent of all the sources accessible to the other pairs. This model, as a special case of the model in [3], is motivated by wireless communications [10], [11]. In a wireless communication environment, each pair of wireless terminals typically possesses some means of estimating their mutual channel. The resulting estimates are highly statistically similar, provided that the terminals communicate on the same carrier frequency. Moreover, any third terminal's observations are essentially uncorrelated with the observations of the first two terminals, provided that the third terminal is located at least half a wavelength away from those two.

The main contribution of this paper is the following. We propose a method for secret key agreement in the pair-wise independent network that is based on well-established point-to-point techniques [9], [10] and repeated application of the one-time pad. Specifically, we propose protocols for three cases of the pair-wise independent model and prove that the secret key rates achieved by our protocols are optimal in the first two cases. Therefore, the capacity problem in such situations is now solved. Furthermore, the efficiency of our protocol for the last case is shown through examples. The innate connections between the pair-wise independent network and graphs can be observed through these protocols.

II. Preliminaries

Suppose $m \geq 2$ terminals respectively observe n independent and identically distributed repetitions of the random variables $(X_1, X_2, ..., X_m)$, denoted by $(X_1^n, X_2^n, \cdots, X_m^n)$ with $X_i^n = (X_{i,1}, \cdots, X_{i,n})$. A group $A \subseteq \{1, \cdots, m\}$ of terminals wish to generate a common secret key, with the help of the remaining terminals. To do so, these m terminals can communicate with each other through a noiseless public channel. The generated group secret key K should be nearly statistically independent of the public transmissions. The entropy rate of the secret key, viz., $H(K)/n$, is called a secret key rate. The largest achievable secret key rate is called the secret key capacity, denoted by $C_{SK}(A)$. It is shown in [3] that

$$C_{SK}(A) = H(X_1, ..., X_m) - \min_{(R_1, ..., R_m) \in \mathcal{R}(A)} \sum_{i=1}^{m} R_i,$$

where

$$\begin{aligned} \mathcal{R}(A) \quad = \quad & \{(R_1, \cdots, R_m) : \sum_{i \in B} R_i \geq H(X_B | X_{B^c}), \\ & B \subset \{1, \cdots, m\}, A \not\subset B\}, \end{aligned}$$

with $X_B = \{X_j : \ j \in B\}$ and $B^c = \{1, \cdots, m\} \backslash B$.

Let $(B_1, \cdots, B_k)$ be a k-partition of $\{1, \cdots, m\}$, such that each element B_l, $1 \leq l \leq k$, intersects with the set $A \subseteq \{1, \cdots, m\}$. Denote by $\mathcal{B}_k(A)$ the set of all such k-partitions. Then an upper bound on the secret key capacity is [3]

$$C_{SK}(A) \leq \min_{2 \leq k \leq |A|} \frac{1}{k-1} I_k(A), \tag{1}$$

where

$$I_k(A) = \min_{(B_1, \cdots, B_k) \in \mathcal{B}_k(A)} \sum_{l=1}^{k} H(X_{B_l}) - H(X_1, \cdots, X_m).$$

III. A pair-wise independent network

In this paper, we focus on a pair-wise independent network, which is a special case of the network described in Section II. Suppose that the observation X_i by terminal i has $m-1$ components $(Y_{i,1},\cdots,Y_{i,i-1},Y_{i,i+1},\cdots,Y_{i,m})$. Each component $Y_{i,j}$ denotes the observation of the source that is accessible only to terminals i and j. Furthermore, it is assumed that

$$I(Y_{i,j},Y_{j,i};\{Y_{k,l}:(k,l)\neq(i,j),(j,i)\})=0. \qquad (2)$$

This implies that each source accessible to a pair of terminals is independent of all other sources–hence, the network is called pair-wise independent.

If a group of terminals in the pair-wise independent network generate a common secret key, then an upper bound on the secret key capacity is given in the following lemma.

Lemma 1: In the pair-wise independent network,

$$C_{SK}(A)\le \min_{2\le k\le|A|}\frac{1}{k-1}I'_k(A), \qquad (3)$$

where

$$I'_k(A)=\min_{(B_1,\cdots,B_k)\in\mathcal{B}_k(A)}\sum_{\substack{i,j:i\in B_l;\\ j\in B_r;l<r}}I(Y_{i,j};Y_{j,i}).$$

Proof: Let $(B_1,\cdots,B_k)$ be an arbitrary k-partition belonging to $\mathcal{B}_k(A)$. It follows from the independence condition (2) that

$$H(X_1,\cdots,X_m)=\sum_{1\le i<j\le m}H(Y_{i,j},Y_{j,i}),$$

and for $1\le l\le k$,

$$H(X_{B_l})=\sum_{i,j:i<j;i,j\in B_l}H(Y_{i,j},Y_{j,i})+\sum_{i,j:i\in B_l;j\notin B_l}H(Y_{i,j}).$$

Then

$$\begin{aligned}&\sum_{l=1}^{k}H(X_{B_l})-H(X_1,\cdots,X_m)\\ =&\sum_{\substack{i,j:i\in B_l;\\ j\in B_r;l<r}}[H(Y_{i,j})+H(Y_{j,i})-H(Y_{i,j},Y_{j,i})]\\ =&\sum_{\substack{i,j:i\in B_l;\\ j\in B_r;l<r}}I(Y_{i,j};Y_{j,i}).\end{aligned}$$

Therefore, the upper bound (3) follows from (1) and the above equality. ■

The decomposition observed in the proof suggests that a graph based approach can be used to study the pair-wise independent network. It is our conjecture that the upper bound (3) is always tight for the pair-wise independent network; we demonstrate that this conjecture holds in at least two special cases.

IV. The Broadcast Case

In this section, we consider the broadcast case of a pair-wise independent network in which the observations of each terminal in $\{2,\cdots,m\}$ are correlated only with the observations of terminal 1 (called the central terminal). In other words, the observation X_i by terminal $i\neq 1$ is equal to $Y_{i,1}$, and $Y_{i,j}$ is a constant for $j\neq 1$.

By restricting $\mathcal{B}_k(A)$ to the set of 2-partitions

$$(\{1,3,\cdots,m\},\{2\}),\cdots,(\{1,2,\cdots,m-1\},\{m\})$$

in (3), we obtain an upper bound on the secret key capacity for the broadcast case

$$C_{SK}(\{1\cdots,m\})\le\min_{2\le i\le m}I(Y_{1,i};Y_{i,1}). \qquad (4)$$

Next, we propose a protocol for the secret key establishment among all m terminals.

Terminals $2,\cdots,m$ begin by separately establishing secret keys with the central terminal using the standard techniques [4], [1]. This results in $m-1$ pair-wise secret keys $K_{1,i}$, $2\le i\le m$, where $K_{1,i}$ denotes the secret key shared by terminals 1 and i. Without loss of generality, these keys are stored using a binary alphabet. Let $|K_{1,i}|$ denote the length of the secret key $K_{1,i}$. According to [4], [1], for any $\epsilon>0$, each secret key $K_{1,i}$, as a function of $(Y^n_{1,i},Y^n_{i,1})$, satisfies the secrecy condition

$$I(K_{1,i};V_{1,i})\le\epsilon, \qquad (5)$$

and the uniformity condition

$$H(K_{1,i})\ge|K_{1,i}|-\epsilon, \qquad (6)$$

where $V_{1,i}$ denotes the public transmissions between terminal i and the central terminal to generate the pair-wise secret key $K_{1,i}$. It follows from the independence condition (2) that

$$I(K_{1,i};\{K_{1,j}:j\neq i\})\le\epsilon. \qquad (7)$$

The entropy rate of $K_{1,i}$ is given by [4], [1]

$$\frac{1}{n}H(K_{1,i})\ge I(Y_{1,i};Y_{i,1})-\epsilon. \qquad (8)$$

Let K_{1,i^*}, $2\le i^*\le m$, be the shortest key among the $m-1$ generated keys, i.e., $|K_{1,i^*}|=\min_{2\le i\le m}|K_{1,i}|$. This implies that

$$I(Y_{1,i^*};Y_{i^*,1})=\min_{2\le i\le m}I(Y_{1,i};Y_{i,1}). \qquad (9)$$

The central terminal sends $\bar{K}_{1,i}\oplus K_{1,i^*}$ to terminal i, where $\bar{K}_{1,i}$ denotes the first $|K_{1,i^*}|$ bits of $K_{1,i}$. At this point, all m terminals have K_{1,i^*}, which is set as the group secret key. The independence between K_{1,i^*} and all the public transmissions is shown in the following proposition.

Proposition 1: For any $\delta>0$, the secret key K_{1,i^*} generated above satisfies

$$I(K_{1,i^*};\{V_{1,i},K_{1,i^*}\oplus\bar{K}_{1,i}:2\le i\le m\})\le\delta. \qquad (10)$$

Proof: In the interest of simple notation, we denote $K_{1,i^*} \oplus \bar{K}_{1,i}$ by $\bar{V}_{1,i}$. Then the left side of (10) is written as

$$\begin{aligned} & I(K_{1,i^*}; V_{1,2}, \cdots, V_{1,m}, \bar{V}_{1,2}, \cdots, \bar{V}_{1,m}) \\ \leq \; & I(K_{1,i^*}; \bar{V}_{1,2}, \cdots, \bar{V}_{1,m}) \\ & + I(K_{1,i^*}, \bar{V}_{1,2}, \cdots, \bar{V}_{1,m}; V_{1,2}, \cdots, V_{1,m}). \end{aligned} \tag{11}$$

The former term in (11) is upper bounded by

$$\begin{aligned} & I(K_{1,i^*}; \bar{V}_{1,2}, \cdots, \bar{V}_{1,m}) \\ & \leq \sum_{\substack{i=2 \\ i \neq i^*}}^{m} \left[H(\bar{K}_{1,i} \oplus K_{1,i^*}) - H(\bar{K}_{1,i} | K_{1,i^*}, \bar{K}_{1,2}, \cdots, \bar{K}_{1,i-1}) \right] \\ & \leq 2(m-2)\epsilon, \end{aligned}$$

where the latter inequality follows from (6) and (7). The latter term in (11) is upper bounded by

$$\begin{aligned} & I(K_{1,i^*}, \bar{V}_{1,2}, \cdots, \bar{V}_{1,m}; V_{1,2}, \cdots, V_{1,m}) \\ = \; & I(K_{1,2}, \cdots, K_{1,m}; V_{1,2}, \cdots, V_{1,m}) \\ = \; & \sum_{i=2}^{m} I(K_{1,i}; V_{1,i}) \leq (m-1)\epsilon, \end{aligned}$$

where the inequality follows from (5). This completes the proof. ■

It follows from (8) and (9) that the generated secret key K_{1,i^*} has a rate close to the upper bound (4). Hence, the protocol is optimal. Furthermore, it is not difficult to show that the protocol is also optimal for the broadcast case with rate constraints (cf., [2]) on the public transmissions.

V. The Sub-group Key Case

We now consider a sub-group key generation problem. Suppose that, in a pair-wise independent network, terminals 1 and m wish to generate a secret key with the help of other $m-2$ terminals. In other words, the sub-group $A = \{1, m\}$ of terminals wish to generate a secret key.

We begin this section with a short overview of some definitions and algorithms related to graphs. Then we propose a protocol for the sub-group key generation problem. This protocol is based on existing graph algorithms. Further, we show that the resulting secret key has a rate close to the capacity.

Let $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ be a weighted directed graph. Let $s \in \mathcal{N}$ be a source node and $t \in \mathcal{N}$ be a destination node in $\mathcal{G}$. An $s-t$ *cut* of the graph $\mathcal{G}$ is a partition of the nodes $\mathcal{N}$ into two sets $\mathcal{N}_1$ and $\mathcal{N}_2$ such that $s \in \mathcal{N}_1$ and $t \in \mathcal{N}_2$. Any edge crossing from $\mathcal{N}_1$ to $\mathcal{N}_2$ is said to be a *cut edge*. The *weight* of an $s-t$ cut is the sum of the weights of its edges. An $s-t$ cut is *minimal* if the weight of the $s-t$ cut is not larger than the weight of any other $s-t$ cut.

A *network flow* is an assignment of flow to the edges of a weighted directed graph such that the amount of flow along the edge does not exceed its weight. The maximal $s-t$ flow problem is to find a maximal feasible flow from the source node s to the destination node t. The labeling algorithm [7] is known to solve the maximal $s-t$ flow problem.

By the max-flow min-cut theorem [6], the maximal $s-t$ flow is equal to the weight of the minimal $s-t$ cut.

We now return to the sub-group secret key generation problem. It follows from Lemma 1 that the secret key capacity, which can be achieved by terminals 1 and m with the help of other terminals, is upper bounded by

$$C_{SK}(\{1,m\}) \leq \min_{(B_1,B_2) \in \mathcal{B}_2(\{1,m\})} \sum_{i,j: i \in B_1; j \in B_2} I(Y_{i,j}; Y_{j,i}), \tag{12}$$

where $\mathcal{B}_2(\{1,m\})$ is the set of all 2-partitions of the set $\{1, \cdots, m\}$ such that either atom of a 2-partition intersects with $\{1, m\}$.

The upper bound (12) can be represented via graphs. Consider a weighted directed graph G_1 with m nodes, each node corresponding to a terminal. The edge from node i to j has weight $I(Y_{i,j}; Y_{j,i})$. Let node 1 be the source node and node m be the destination node. Then the upper bound (12) is equivalent to the minimal $s-t$ cut of G_1.

Next, we propose a protocol for the secret key establishment between terminals 1 and m.

All the terminals begin by establishing pair-wise secret keys using the standard techniques [4], [1]. This results in $\binom{m}{2}$ pair-wise secret keys. Let $K_{i,j}$ $(= K_{j,i})$ denote the secret key shared by terminals i and j. Each secret key $K_{i,j}$, as a function of $(Y_{i,j}^n, Y_{j,i}^n)$, satisfies certain secrecy condition and uniformity condition as in (5), (6). Further, for any $\epsilon > 0$,

$$I(K_{i,j}; \{K_{k,l} : (k,l) \neq (i,j), (j,i)\}) \leq \epsilon, \tag{13}$$

and the entropy rate of $K_{i,j}$ is given by [4], [1]

$$\frac{1}{n} H(K_{i,j}) \geq I(Y_{i,j}; Y_{j,i}) - \epsilon. \tag{14}$$

Based on the pair-wise secret key $K_{i,j}$, terminal i can cipher $|K_{i,j}|$ random bits with $K_{i,j}$ through the one-time pad before transmitting these random bits to terminal j (and vice versa). This implies the existence of a secure channel between nodes i and j with capacity $\frac{1}{n}|K_{i,j}|$.

Consider a weighted directed graph G_2 with m nodes, each node corresponding to a terminal. The weight of an edge (i,j) in the graph is equal to the capacity of the secure channel connecting terminals i and j, i.e., $\frac{1}{n}|K_{i,j}|$. Using the labeling algorithm [7], one can find the maximal $s-t$ flow F in this graph. Accordingly, terminal 1 can securely send random bits through the network to terminal m at rate F. Let these random bits be the secret key of terminals 1 and m. By arguments similar to those used in the proof of Proposition 1, it is easy to show that this secret key is nearly statistically independent of the public transmissions.

Proposition 2: Let V denote all the public transmissions needed in the protocol above. For any $\delta > 0$, the secret key K generated above satisfies $I(K; V) \leq \delta$.

According to the max-flow min-cut theorem [6], the rate F of the generated secret key is equal to the minimal $s-t$

cut of G_2. It follows from (14) that the minimal $s-t$ cut of G_2 is close to the minimal $s-t$ cut of G_1. Hence, the achieved secret key rate is close to the upper bound (12), and the protocol is optimal.

VI. The Group Key Case

In this section, we examine the problem of all the terminals in a pair-wise independent network generating a common secret key. We start by a short overview of more definitions and algorithms related to graphs. Then we propose a protocol for the group secret key generation problem. This protocol is based on existing graph algorithms. Finally, we demonstrate the efficiency of this protocol through several examples.

Let $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ be a weighted undirected graph. The graph $\mathcal{G}$ is said to be *connected* if for every two distinct nodes $i, j \in \mathcal{N}$, there exists a path from node i to node j. Otherwise, the graph is said to be *unconnected*. Define a *multi-cut* of $\mathcal{G}$ to be a partition of the nodes $\mathcal{N}$ into several sets $\mathcal{N}_1, \cdots, \mathcal{N}_L$, $2 \leq L \leq m$, with m being the number of nodes in $\mathcal{G}$. Any edge $(i, j) \in \mathcal{E}$ with end-nodes i, j belonging to different sets is said to be a *multi-cut edge*. The *weight* of a multi-cut is the weight sum of its edges. The *normalized weight* of a multi-cut is the weight of the multi-cut divided by $L-1$, where L is the number of sets in the partition of $\mathcal{G}$ generating the multi-cut.

Given a connected undirected graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, let $\mathcal{E}_1$ be a subset of $\mathcal{E}$ such that $\mathcal{T} = (\mathcal{N}, \mathcal{E}_1)$ is a tree. Such a tree is called a *spanning tree*. A *maximum spanning tree* from a weighted graph is defined as a spanning tree such that the weight sum of its edges is as large as possible. The problem of finding a maximum spanning tree can be solved by several greedy algorithms. Two examples are Kruskal's algorithm and Prim's algorithm (cf., e.g., [6]).

The upper bound (3) on the secret key capacity for the group secret key case, i.e., $A = \{1, \cdots, m\}$, can be represented via graphs. Consider a weighted undirected graph G_3 with m nodes, each node corresponding to a terminal. The weight of an edge (i, j) in the graph is equal to $I(Y_{i,j}; Y_{j,i})$. Note that each multi-cut of the graph G_3 is equivalent to a partition in (3), and the set of all multi-cuts of the graph G_3 is precisely equivalent to the set of partitions $\{(B_1, \cdots, B_k) \in \mathcal{B}_k(\{1, \cdots, m\}) : 2 \leq k \leq m\}$ in (3). Moreover, the normalized weight of a multi-cut is precisely $\frac{1}{k-1} I'_k(\{1, \cdots, m\})$. Consequently, we have the following corollary.

Corollary 1: The secret key capacity for the group secret key case is upper bounded by the minimal normalized weight of the multi-cuts of G_3. In particular, this upper bound implies the following two upper bounds:

i). the minimal weight of the cuts of G_3, where a cut is a multi-cut generated by a partition into 2 sets;

ii). the weight sum of all edges in G_3 divided by $m-1$.

Next, we propose a protocol for the group secret key generation problem. All the terminals begin by establishing pair-wise secret keys using the standard techniques [4], [1]. Let $K_{i,j}$ $(= K_{j,i})$ denote the secret key shared by terminals i and j. These secret keys satisfy the certain secrecy condition, uniformity condition, and (13), (14).

Consider a weighted undirected graph G_4 with m nodes, each corresponding to a terminal. The weight of an edge (i, j) in the graph is equal to the length[1] of the corresponding pair-wise secret key $K_{i,j}$, i.e., $|K_{i,j}|$.

Our group key generation algorithm is related to Lemma 2 below. This lemma discusses the generation of a single secret bit among m nodes, based on a single bit from each of the $m-1$ pair-wise secret keys whose corresponding edges constitute a spanning tree.

Lemma 2: Consider an arbitrary tree connecting m nodes. If every pair of neighbor nodes on the tree shares a single pair-wise secret bit, then a single secret bit can be generated among all m nodes.

Proof: A simple algorithm on generating a single secret bit among all m nodes is illustrated below.

Single Bit Algorithm:

Step 1. Randomly pick up an edge (i^*, j^*) from the spanning tree. Nodes i^* and j^* share a secret bit B_{i^*,j^*}.

Step 2. If node i knows B_{i^*,j^*}, but its neighbor node j does not, then node i sends $B_{i^*,j^*} \oplus B_{i,j}$ to node j, where $B_{i,j}$ is the secret bit shared by nodes i and j. Upon receiving this message, node j is able to decode B_{i^*,j^*}. Repeat this step until the above condition does not hold. □

This algorithm stops when all the nodes are able to decode B_{i^*,j^*}. It is trivial to show the independence between B_{i^*,j^*} and the public transmissions. Hence, B_{i^*,j^*} is a secret bit. ■

Our group secret key generation algorithm is given below.

Group Key Generation Algorithm:

Let G be the weighted undirected graph G_4 defined above.

Step 1: Determine a maximum spanning tree from G, using any known algorithm (e.g., Kruskal's or Prim's). If there is more than one maximum spanning tree, randomly select one.

Step 2: Apply the single bit algorithm to generate a single secret bit among all nodes, based on a single bit from every pair-wise secret key on the determined maximum spanning tree. Note that these used bits will be of no use in the remaining group key generation process.

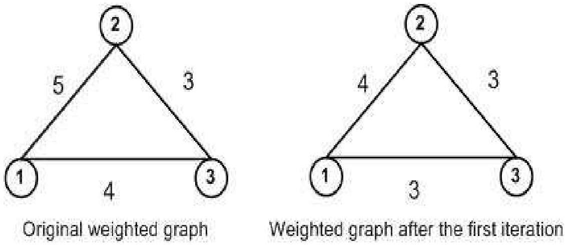
Step 3: Update the graph by reducing the edge weight by 1 for the edges on the determined spanning tree. Remove an edge when its weight becomes zero.

Step 4: If the remaining graph G is unconnected, then set the group secret key as the collection of all generated secret bits. Otherwise, return to Step 1. □

Since each iteration of the group key generation algorithm leads to a single secret bit, the length of the resulting secret key is equal to the number of iterations of the algorithm that can be run until the graph becomes unconnected. The purpose of searching a maximum spanning tree (rather than picking up an arbitrary spanning tree) in Step 1 is to maximize the number of iterations of the algorithm by means of "balancing" edge weights in the weight reduction procedure.

By arguments similar to those used in the proof of Proposition 1, it is easy to show that the secret key resulting from

[1] For the purpose of simple notations, we shall use the length, rather than the rate, of a secret key as an edge weight. This should not lead to any confusion.



Original weighted graph

Weighted graph after the first iteration

Fig. 1. Example network with 3 nodes

the above algorithm is nearly statistically independent of the public transmissions.

Proposition 3: Let V denote all the public transmissions needed in the protocol above. For any $\delta > 0$, the secret key K generated above satisfies $I(K;V) \leq \delta$.

We illustrate the operations of the group key generation algorithm through the following example.

Example 1: Consider a network with 3 nodes. Nodes 1 and 2 share a secret key of 5 bits; nodes 1 and 3 share a secret key of 4 bits; and nodes 2 and 3 share a secret key of 3 bits. This network is drawn in the left part of Fig. 1.

Let the pair-wise secret keys be $K_{1,2} = (K_{1,2}^1, \cdots, K_{1,2}^5)$, $K_{1,3} = (K_{1,3}^1, \cdots, K_{1,3}^4)$, and $K_{2,3} = (K_{2,3}^1, \cdots, K_{2,3}^3)$, where $K_{i,j}^k$ denotes the k^{th} bit of the secret key shared by nodes i and j.

The spanning tree $((1,2),\ (1,3))$ is the maximum spanning tree from the graph in the left part of Fig. 1, as it has a larger weight (= 9) than other spanning trees. Hence, by the single bit algorithm, node 1 transmits $K_{1,2}^1 \oplus K_{1,3}^1$ and sets $K_{1,2}^1$ (or $K_{1,3}^1$) as the secret bit. Update the graph by reducing the weights of the edges $(1,2)$, $(1,3)$ by 1. This results in the graph given in the right part of Fig. 1.

By repeating the above process, the determined maximum spanning trees and the corresponding public transmissions in the next five iterations are

$$((1,2),\ (1,3)), \quad ((1,2),\ (2,3)), \quad ((1,2),\ (2,3)),$$
$$((1,3),\ (2,3)), \quad ((1,2),\ (1,3)),$$

and

$$K_{1,2}^2 \oplus K_{1,3}^2, \quad K_{1,2}^3 \oplus K_{2,3}^1, \quad K_{1,2}^4 \oplus K_{2,3}^2,$$
$$K_{1,3}^3 \oplus K_{2,3}^3, \quad K_{1,2}^5 \oplus K_{1,3}^4,$$

respectively. The algorithm stops after these iterations, as the remaining graph is unconnected. The group secret key is set as $(K_{1,2}^1, K_{1,2}^2, K_{1,2}^3, K_{1,2}^4, K_{1,3}^3, K_{1,2}^5)$. By restricting k to $|A| = 3$ and setting $Y_{i,j} = Y_{j,i} = K_{i,j}$ in (3), we find that the length of any group secret key in this example cannot be larger than 6 bits. Hence, the algorithm is optimal.

For a network with 3 nodes, determining a maximum spanning tree in the group key generation algorithm is equivalent to determining a node such that the weight sum of two edges connecting with this node is the largest.

Example 2: Consider a network with m nodes and all $\binom{m}{2}$ edges having the same even weight $w = 2u$, for a certain positive integer u. A secret key of length mu bits can be generated by using the group key generation algorithm. On the other hand, by restricting k to $|A| = m$ and setting $Y_{i,j} = Y_{j,i} = K_{i,j}$ in (3), we find that the length of any group secret key in this example cannot be larger than $\frac{w\binom{m}{2}}{m-1} = mu$ bits. Hence, the algorithm is optimal.

Although the group key generation algorithm is shown to be optimal in the examples above, its potential non-optimality is demonstrated by the following example.

Example 3: Consider a network with 4 nodes. Each node is connected with every other node by an edge of weight 1. It is clear that $((1,2),\ (1,3),\ (1,4))$ is a maximum spanning tree of the graph, which means that 1 secret bit can be generated from it. However, the updated graph then becomes unconnected, resulting in a secret key of 1 bit.

Nevertheless, the upper bound in (3) can be achieved by simply making a *better selection from the possible maximal spanning trees*. One such tree is $((1,2),\ (2,3),\ (3,4))$. After the weight reduction, the new graph is still connected, having the spanning tree $((1,3),\ (1,4),\ (2,4))$. Hence, 2 secret bits, which is optimal, can be established in this manner.

This example suggests the importance of deliberately selecting a maximum spanning tree in Step 1 of the algorithm. What a good selection scheme might look like, and whether it would guarantee the optimality of this algorithm, remains open.

ACKNOWLEDGMENT

The authors would like to thank Yogendra Shah and Inhyok Cha for introducing the sub-group key generation problem and for helpful discussions on this topic.

REFERENCES

[1] R. Ahlswede and I. Csiszár, “Common randomness in information theory and cryptography, Part I: Secret sharing,” *IEEE Trans. Inform. Theory*, vol. 39, pp. 1121–1132, July 1993.

[2] I. Csiszár and P. Narayan, “Common randomness and secret key generation with a helper,” *IEEE Trans. Inform. Theory*, vol. 46, pp. 344–266, Mar. 2000.

[3] I. Csiszár and P. Narayan, “Secrecy capacities for multiple terminals,” *IEEE Trans. Inform. Theory*, vol. 50, pp. 3047–3061, Dec. 2004.

[4] U. Maurer, “Secret key agreement by public discussion from common information,” *IEEE Trans. Inform. Theory*, vol. 39, pp. 733–742, May 1993.

[5] U. Maurer and S. Wolf, “Secret key agreement over a non-authenticated channel — parts I-III,” *IEEE Trans. Inform. Theory*, vol. 49, pp. 822–851, Apr. 2003.

[6] A. Schrijver, *Combinatorial Optimization — Polyhedra and Efficiency*, New York: Springer, 2003.

[7] G. Strang, *Introduction to Applied Mathematics*, Massachusetts: Wellesley, 1986.

[8] C. Ye and P. Narayan, “The private key capacity region for three terminals,” *Proceedings Int. Symp. on Inform. Theory*, p. 44, 2004.

[9] C. Ye and P. Narayan, “Secret key and private key constructions for simple multiterminal source models,” *Proceedings Int. Symp. on Inform. Theory*, pp. 2133–2137, 2005.

[10] C. Ye, A. Reznik and Y. Shah, “Extracting secrecy from jointly Gaussian random variables,” *Proceedings Int. Symp. on Inform. Theory*, pp. 2593–2597, 2006.

[11] C. Ye, A. Reznik, G. Sternberg and Y. Shah, “Secret key generation from multipath fading channels,” *submitted to IEEE Int. Conf. on Communications*, 2007.