

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 in which every pair of terminals 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 over a graphical representation of the network. 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) Two designated terminals wish to generate a secret key with the help of other terminals. 3) All terminals wish to generate a common secret key. In each of these cases, we show that our two-step approach can yield an optimal protocol, in terms of the resulting secret key rates. Furthermore, such a protocol is provided for the first two problems, while an efficient protocol is given for the third problem.

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 wishes 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 in which every pair of terminals observes a common source that is independent of all other sources accessible to the other pairs. This model, as a special case of the model in [3], is motivated by the practical aspects of the wireless communication channel (see, e.g. [10]).

Our main contributions are as follows. We propose a method for secret key agreement in the pair-wise independent network; the method is based on well-established point-to-point techniques [1], [4], [9], [10] and repeated application of the one-time pad over a graphical representation of the network. We demonstrate the existence of optimal graph-based protocols for the three cases considered. In the first two cases, we provide the optimal protocols; in the third case, we show the efficiency of our protocol through examples. The innate connection between the pair-wise independent network and graphs can be observed through these protocols. For the

three special cases considered here, our results also prove a conjecture of Csiszár and Narayan [3] about the tightness of a certain upper bound on network secrecy capacity.

II. PRELIMINARIES

Suppose $m \geq 2$ terminals respectively observe n independent and identically distributed repetitions of the random variables (X_1, X_2, \dots, X_m) , denoted by $(X_1^n, X_2^n, \dots, X_m^n)$ with $X_i^n = (X_{i,1}, \dots, X_{i,n})$. A group $A \subseteq \{1, \dots, m\}$ of terminals wishes 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

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

where

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

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

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

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

where

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

It is conjectured in [3] that this bound is tight, although the proof is only given for some special cases.

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 $(Y_{i,1}, \dots, Y_{i,i-1}, Y_{i,i+1}, \dots, Y_{i,m})$. 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

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

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 generates 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,

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

where

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

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

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

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

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

Then

$$\begin{aligned} & \sum_{l=1}^k H(X_{B_l}) - H(X_1, \dots, X_m) \\ &= \sum_{\substack{i,j:i \in B_1; \\ j \in B_r; l < r}} [H(Y_{i,j}) + H(Y_{j,i}) - H(Y_{i,j}, Y_{j,i})] \\ &= \sum_{\substack{i,j:i \in B_1; \\ j \in B_r; l < r}} I(Y_{i,j}; Y_{j,i}). \end{aligned}$$

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

The conjecture in [3] suggests that the upper bound (3) is always tight for the pair-wise independent network; we demonstrate that this conjecture holds for the special cases considered in this paper.

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 $\{2, \dots, m\}$ 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 $j \neq 1$.

Theorem 1: The secret key capacity of the broadcast case is given by

$$C_{SK}(\{1, \dots, m\}) = \min_{2 \leq i \leq m} I(Y_{1,i}; Y_{i,1}). \quad (4)$$

Proof: The upper bound (converse part) is obtained by restricting the set of partitions $\mathcal{B}_k(A)$ in (3) to the set of 2-partitions

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

To show achievability we consider the following protocol. Terminals $2, \dots, m$ 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 $K_{1,i}$, $2 \leq i \leq m$, 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 $(Y_{1,i}^n, Y_{i,1}^n)$, satisfies the secrecy condition

$$I(K_{1,i}; V_{1,i}) \leq \epsilon, \quad (5)$$

and the uniformity condition

$$H(K_{1,i}) \geq |K_{1,i}| - \epsilon, \quad (6)$$

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

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

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

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

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

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

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 uniformity of the group secret key is obvious. 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

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

Proof: To simplify notation, we denote $K_{1,i^*} \oplus \bar{K}_{1,i}$ by $\bar{V}_{1,i}$. Then the left side of (10) becomes

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

The first term in (11) is upper bounded by

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

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

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

where the latter 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 (4), completing the proof of Theorem 1. ■

To conclude this section, we add that it is not difficult to show that the protocol is also optimal for the broadcast case with rate constraints (c.f., [2]) on the public transmissions. We omit the proof of this result due to space constraints.

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 and prove its optimality. This protocol is based on existing graph algorithms.

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 prove the secret key capacity of the sub-group key case.

Theorem 2: The secret key capacity of the sub-group key case is given by

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

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

Proof: The upper bound (converse part) follows directly from Lemma 1.

Next, 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 (12) is equivalent to the minimal $s-t$ cut of G_1 .

To show achievability we consider the following protocol. All terminals begin by establishing pair-wise secret keys using 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 conditions and uniformity conditions as in (5), (6). Further, for any $\epsilon > 0$,

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

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

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

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.

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 (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 wishing to generate a common secret key. We start with an overview of undirected graphs. We then provide a proof that there exists an optimal algorithm that generates pair-wise keys first and one-time pad operations over the network graph thereafter. Finally, we propose a protocol for the group secret key generation problem that, while not demonstrably optimal, is shown to be efficient in many examples.

A *multi-graph* is a graph that is permitted to have parallel edges, i.e., edges that have the same end nodes. Let $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ be an unweighted undirected multi-graph. Define a *multi-cut* of \mathcal{G} to be a partition of the nodes \mathcal{N} into several sets $\mathcal{N}_1, \dots, \mathcal{N}_L$, $2 \leq L \leq m$, 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 of \mathcal{G} is the total number 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* in 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 in an undirected graph has been well studied and can be solved efficiently by several greedy algorithm-based methods (cf., e.g., [6]).

We prove the secret key capacity of the group key case.

Theorem 3: The secret key capacity of the group key case is given by

$$C_{SK}(\{1, \dots, m\}) = \min_{2 \leq k \leq m} \frac{1}{k-1} I'_k(\{1, \dots, m\}), \quad (15)$$

where

$$I'_k(\{1, \dots, m\}) = \min_{(B_1, \dots, B_k) \in \mathcal{B}_k(\{1, \dots, m\})} \sum_{\substack{i, j: i \in B_l; \\ j \in B_r; l < r}} I(Y_{i,j}; Y_{j,i}).$$

Proof: The upper bound (converse part) follows directly from Lemma 1.

To show achievability we begin with a lemma regarding the generation of a single secret bit on a 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 for 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. \square

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. \blacksquare

Next, consider secret key generation over a block length of n source observations. Let all terminals establish pair-wise secret keys using the standard techniques [4], [1]. Let $K_{i,j}$ ($=K_{j,i}$) denote the secret key (with an integer number $\lfloor nI(Y_{i,j}; Y_{j,i}) \rfloor$ of bits) shared by terminals i and j . These secret keys satisfy certain secrecy and uniformity conditions and (13), (14).

Consider an unweighted multi-graph G_3 with m nodes, each corresponding to a terminal. The number of edges connecting nodes i and j in G_3 is equal to the length of the corresponding pair-wise secret key $K_{i,j}$, i.e., $|K_{i,j}|$. Here, each edge is associated with a unique, random, independent secret bit.

It follows from Lemma 2 that all the nodes in G_3 can generate one common secret bit from a spanning tree in G_3 .

Let the secret key be a collection of secret bits resulting from all edge-disjoint spanning trees in G_3 .

The uniformity of this secret key is obvious. By arguments similar to those used in the proof of Proposition 1, it is easy to show that the secret key is nearly statistically independent of the public transmissions.

The length of this secret key is equal to the number of edge-disjoint spanning trees contained in G_3 . It is known from a theorem of Nash-Williams and Tutte [6, Corollary 51.1a] that the number of edge-disjoint spanning trees contained in G_3 is equal to

$$\min_{2 \leq k \leq m} \min_{(B_1, \dots, B_k) \in \mathcal{B}_k(\{1, \dots, m\})} \bar{W}(B_1, \dots, B_k), \quad (16)$$

where $\bar{W}(B_1, \dots, B_k)$ is the normalized weight of the multi-cut generated by the partition (B_1, \dots, B_k) of G_3 . Clearly,

$$\bar{W}(B_1, \dots, B_k) = \frac{1}{k-1} \sum_{\substack{i, j: i \in B_l; \\ j \in B_r; l < r}} \lfloor nI(Y_{i,j}; Y_{j,i}) \rfloor.$$

The rate of this secret key is equal to (16) divided by n , which is arbitrarily close to (15) for sufficiently large n . This completes the proof of the theorem. \blacksquare

The proof of Theorem 3 relies on Nash-Williams and Tutte's result, which is non-constructive. Therefore, unlike our first two cases, this proof does not provide an algorithm for achieving the secret key capacity. Finding such an algorithm remains an open problem. However, we propose a simple algorithm that is observed to perform well in many examples.

Group Key Generation Algorithm:

All terminals begin by establishing pair-wise secret keys using standard techniques [4], [1]. Let G_4 be a weighted undirected graph with m nodes, each corresponding to a terminal. The weight of an edge (i, j) in the graph is equal to the length of the corresponding pair-wise secret key $K_{i,j}$.

Step 1: Determine a maximum spanning tree in G_4 , 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_4 is unconnected, then set the group secret key as the collection of all generated secret bits. Otherwise, return to Step 1. \square

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

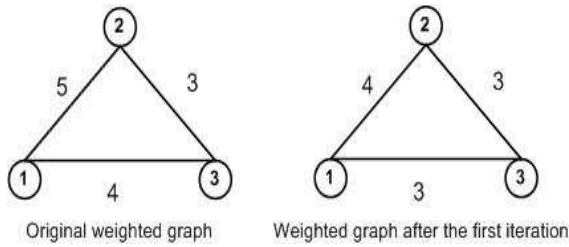


Fig. 1. Example network with 3 nodes

weights in the weight reduction procedure. We illustrate the proposed algorithm through the following examples.

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 $K_{1,2} = (K_{1,2}^1, \dots, K_{1,2}^5)$, $K_{1,3} = (K_{1,3}^1, \dots, K_{1,3}^4)$, and $K_{2,3} = (K_{2,3}^1, \dots, K_{2,3}^3)$, 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 $Y_{i,j} = Y_{j,i} = K_{i,j}^k$ 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, remain open questions.

ACKNOWLEDGMENT

The authors would like to thank Sirin Nitinawarat (U. of Maryland) for pointing out the theorem by Nash-Williams and Tutte, which was used in the proof of Theorem 3. Additionally, the authors would like to thank Yogendra Shah and Inhyok Cha (InterDigital Comm. Corp.) 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.