Security Analysis and Extensions of the PCB Algorithm for Distributed Key Generation

Radha Poovendran[1] and Brian Matt[2]

[1] University of Washington, Seattle, WA 98195-2500 radha@ee.washington.edu
[2] McAffee Research, Rockville, MD 20850-4601 brian_matt@mcafee.com

Dedicated to Professor Carlos Berenstein on his 60^{th} birthday.

Summary. Poovendran, Corson and Baras presented a distributed cryptographic key generation algorithm that was suitable for wireless networking environment. However, the security as well as the computational complexity of their scheme were never analyzed. In this work, we present information theoretic analysis of their work and derive the properties of the cryptographic keys that are generated by their scheme. We also present efficient computational schemes that would require only logarithmic number of steps in group size to compute the common keys.

1 Introduction

Broadcast is the inherent mode of communication in wireless networks that deploy omnidirectional antennas. In broadcast mode, all members who are within the communication range of the transmitting node can receive the message, thus making it resource-efficient for the sender as well as the network. However, in many applications the set of users that have access to the communication must be restricted. The use of cryptography is one way to restrict the set of members who can access the communication. When the amount of data is high, the use of symmetric keys will help reduce the computational overhead due to the encryption and decryption. However, the use of symmetric keys require that all members share the same keys for decryption. Several methods have been proposed to generate and distribute a single common key to all the members of a communicating group. Among these methods is the distributed key generation method proposed by Poovendran, Corson and Baras in [PCB],which we call the PCB scheme in this paper. The PCB scheme made use of modulo arithmetic and generalized the property of one-time pad, proposed by Shannon [CS]. However, as of now there is no analysis on the security properties of the PCB method. In this work we enhance the original PCB algorithm and present the security analysis based on infor-

mation theoretic techniques. We also show how to develop a computationally efficient algorithm for computing the PCB keys.

The organization of the chapter is as follows: we first review the one-time pad and its properties using probabilistic as well as information theoretic approaches. We then present the PCB algorithm. We provide detailed analysis of the PCB algorithm using probabilistic as well as information theoretic techniques. We also show how to develop computationally efficient techniques that will enable efficient calculation of the group's shared key.

2 Properties of the One-time Pad based Encryption

We use the notations in [DS] to define a cryptosystem. A cryptographic system is a pentuple $\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D}$, where the following conditions are satisfied:

1. $\mathcal{P}$ is a finite set of possible messages or *plaintexts*.
2. $\mathcal{C}$ is a finite set of possible encrypted messages or *ciphertexts*.
3. $\mathcal{K}$ is the finite set of keys or the *keyspace*.
4. $\mathcal{E}_{\mathcal{K}}$ is the encryption rule for a given key K. We denote $E_K : \mathcal{P} \rightarrow \mathcal{C}$.
5. $\mathcal{D}_{\mathcal{K}}$ is the decryption rule for a given key K. We denote $D_K : \mathcal{C} \rightarrow \mathcal{P}$.

2.1 One-time Pad Cryptosystem

Let p be a large prime. The plaintext and the encryption key are of the same length and chosen independently and are assumed to be picked uniformly in the interval $[0, p-1]$. The encryption rule is the modulo addition w.r.t. p. The one-time pad scheme is given below:

1. $\mathcal{P} \in \{0, 1, \cdots, p-1\}$.
2. $\mathcal{C} \in \{0, 1, \cdots, p-1\}$.
3. $\mathcal{K} \in \{0, 1, \cdots, p-1\}$.
4. $\mathcal{E}_{\mathcal{K}}$ is a rule $\forall X \in \mathcal{P}$, $E_K(X) = X + K$ mod p.
5. $\mathcal{D}_{\mathcal{K}}$ is a rule $\forall Y \in \mathcal{C}$, $D_K(Y) = Y - K$ mod p.

If it can be shown that the ciphertext Y is independent of the encryption key or plaintext X, then observing the ciphertext Y reveals no information about the plaintext X, and hence the mutual information ([CT]) is $I(X \wedge Y) = E_{P_{XY}}[\log \frac{P_{XY}}{P_X P_Y}] = 0$. The main idea behind the one-time pad based encryption is stated in the following theorem:

Theorem 1. Let p be a large prime number, A, B be two random variables that are mutually independent and uniformly distributed over the interval $[0, p-1]$. Let $C = A + B$ mod p. Then the random variable C is uniformly distributed in the interval $[0, p-1]$, and the random variables A, B, C are mutually independent.

Proof:: We compute the distribution of C using

$$P(C=k) = \sum_{i=0}^{p-1} P(C=k|a=i)P(A=i) \tag{1}$$
$$= \sum_{i=0}^{p-1} P(A+B=k|A=i)P(A=i) \tag{2}$$
$$= \frac{1}{p}\sum_{i=0}^{p-1} P(B=k-i|A=i) \tag{3}$$
$$= \frac{1}{p}\sum_{i=0}^{p-1} P(B=k-i) \tag{4}$$
$$= \frac{1}{p}. \tag{5}$$

Hence, C is uniformly distributed over the range $[0, p-1]$. We now show that C is independent of A, B.

$$P(C=k|A=i) = P(A+B=k|A=i) \tag{6}$$
$$= P(B=k-i|A=i) \tag{7}$$
$$= P(B=k-i) \tag{8}$$
$$= \frac{1}{p} \tag{9}$$
$$= P(C=k). \tag{10}$$

Hence, C is not only uniformly distributed in the interval $[0, p-1]$, but also independent of A (as well as B).

A direct consequence of these derivations is the fact that the random variable C is uncorrelated to random variable A or B. Hence, observing random variable C provides no information ([CT]) about random variables A or B. This idea can be expressed in terms of the mutual information between random variables C and A as:

$$I(C \wedge A) = E_{P_{AC}}[\log \frac{P_{AC}}{P_A P_C}]. \tag{11}$$

Noting that $P_{AC} = P_A P_C$, since A is independent of C, we find that $\log \frac{P_{AC}}{P_A P_C} = \log(1) = 0$. Hence, the mutual information between the random variables A and C is zero. Substituting $A = X, B = K$, and $C = Y$ in the proof above shows that the one-time pad encryption leads to ciphertext that is uniformly distributed in the interval $[0, p-1]$ and satisfies $I(Y \wedge X) = 0$ as well as $I(Y \wedge K) = 0$.

3 Review of the PCB Scheme

The PCB scheme presented in [PCB] can be viewed as a generalized version of one-time pad encryption. The PCB scheme consists of a Trusted Third Party based initialization step followed by distributed key generation step. We first define relevant notations. Let $E_{K_i}(m)$ denote the encryption of message m with key K_i, and $A \rightarrow B : m$ to denote a message m sent from entity A to entity B. The PCB scheme is described below.

3.1 Initialization

In the initializaiton step, a Trusted Third Party (TTP) selects n participants of the distributed key generation scheme labeled $\{M_i\}_{i=1}^{n}$. It is assumed that the TTP shares a pairwise key K_i with member M_i of the group. The TTP chooses a large prime p, generates n uniformly distributed and independent random variables denoted $\alpha_{i,0}$, with $i = 1, \cdots, n$. The TTP computes

$$\theta_0 = \sum_{i=1}^{n} \alpha_{i,0}. \tag{12}$$

The TTP initializes each entity M_i using the following message transfer

$$TTP \rightarrow M_i : E_{K_i}(\alpha_{i,0}, \theta_0). \tag{13}$$

3.2 Broadcast Enhanced Distributed Key Generation

The distributed key generation consists of two stages. In the first stage each node generates its contribution, and secures and transmits it. In the second stage, each node collects contributions of all other nodes and combines them to generate the group key and its future onetime pad. The original PCB scheme in [PCB] assumed pairwise links between nodes. This procedure is computationally intensive and can be avoided in wireless broadcast environments. We also note that in the original PCB scheme there was no mechanism to make the participants commit to the shares they would contribute to the group key generation. Lack of comittment makes the original PCB scheme vulnerable to attacks by participants who can bias the final outcome. While we do not elaborate on the key space bias in this work, we eliminate it using a committment. These two changes are reflected in steps 4 and 5 of the algorithm presented below. At the iteration step j, a participant M_i performs the following operations to generate its share of the distributed key:

1. M_i generates a Fractional Key $FK_{i,j}$.
2. M_i generates a Hidden Fractional Key $HFK_{i,j} = FK_{i,j} + \alpha_{i,j-1}$.
3. M_i generates a commitment $com_{i,j} = g^{HFK_{i,j}}$.
4. $M_i \rightarrow * : com_{i,j}$.

5. $M_i \rightarrow * : E_{\theta_{j-1}}(HFK_{i,j})$.

A participant M_i then combines the shares to compute the group key and the fresh one-time pad for its computations. A participant M_i performs the following operations:

1. $\forall l \in \{1, \cdots, n\}$, obtain $HFK_{l,j}$, compute and verify that $g^{HFK_{l,j}} = com_{l,j}$. If true, proceed to the next steps, else, terminate.
2. Compute the sum of all the Hidden Fractional Keys $\sum_{l=1}^{n} HFK_{l,j} = \sum_{l=1}^{n} FK_{l,j} + \sum_{l=1}^{n} \alpha_{l,j-1}$.
3. Compute the new group key as

$$\theta_j = \sum_{l=1}^{n} HFK_{l,j} + (p-1)\alpha_{l,j-1} = \sum_{l=1}^{n} FK_{l,j} \text{ mod p.} \tag{14}$$

4. Compute $\alpha_{i,j} = \theta_j + (p-1)FK_{i,j} \text{mod p.}$

The PCB scheme is represented in a schematic diagram given below:

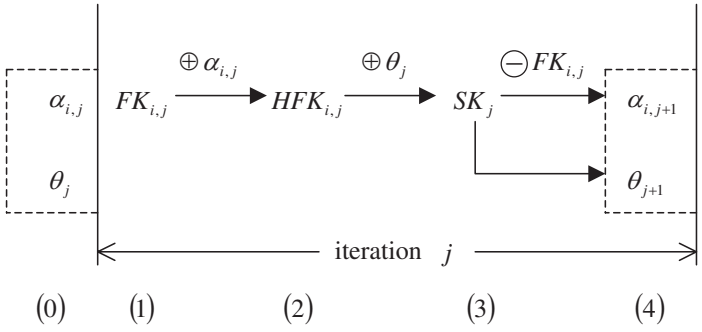

Fig. 1. Iteration and Mappings of the Key Generation Algorithm

4 Security Analysis of the PCB Scheme

As noted earlier, the PCB paper did not provide analysis of the scheme. We provide the security analysis of the PCB scheme in this section. We make the following claims about the security of the PCB scheme:

Theorem 2. If random variables $\alpha_{i,0}$ are mutually independent and uniformly distributed in the interval $[0, p-1]$, then the group key θ_0, defined by:

$$\theta_0 = \sum_{i=1}^{n} \alpha_{i,0} \tag{15}$$

is uniform in the interval $[0, p-1]$ and is mutually independent with respect to any subset consisting of $(n-1)$ of the random variables $\alpha_{i,0}; i = 1, \cdots, n$.

Proof: We first show that θ_0 is uniformly distributed and then show that θ_0 is mutually independent of any set of (n-1) $\alpha_{i,0}$. We prove that θ_0 is uniformly distributed using induction. Let $U_i = U_{i-1} + \alpha_{i,0}$ with $U_0 = 0$. Then $U_1 = \alpha_{1,0}; U_2 = \alpha_{1,0} + \alpha_{2,0}; \cdots U_n = \theta_0$. We now show that $U_i; i = 0, 1, \cdots n$ are uniformly distributed. Note that for $i = 1$, $U_1 = \alpha_{1,0}$ is by definition of $\alpha_{1,0}$ is uniform over the interval $[0, p-1]$. For $i = 2$, we have

$$\begin{aligned}
P(U_2 = k) &= \sum_{s_1=0}^{p-1} P(U_2 = k | \alpha_{1,0} = s_1) P(\alpha_{1,0} = s_1) && (16)\\
&\overset{(i)}{=} \frac{1}{p} \sum_{s_1=0}^{p-1} P(\alpha_{1,0} = s_1 + \alpha_{2,0} = k | \alpha_{1,0} = s_1) && (17)\\
&= \frac{1}{p} \sum_{s_1=0}^{p-1} P(\alpha_{2,0} = k - s_1 | \alpha_1 = s_1) && (18)\\
&\overset{(ii)}{=} \frac{1}{p} \sum_{s_1=0}^{p-1} P(\alpha_{2,0} = k - s_1) && (19)\\
&= \frac{1}{p} && (20)
\end{aligned}$$

The step (i) follows from the definition of U_2 and the step (ii) follows from the observation that under modulo arithmetic as the summation includes all the p terms, even if there is an index shift. Hence, U_2 is uniformly distributed. Now we show that U_2 is independent of $\alpha_{1,2}$.

$$\begin{aligned}
P(U_2 = k | \alpha_{1,0} = s_1) &= P(\alpha_{1,0} = s_1 + \alpha_{2,0} = k | \alpha_{1,0} = s_1) && (21)\\
&= P(\alpha_{2,0} = k - s_1 | \alpha_{1,0} = s_1) && (22)\\
&\overset{(i)}{=} P(\alpha_{2,0} = k - s_1) && (23)\\
&= \frac{1}{p} && (24)\\
&= P(U_2 = k). && (25)
\end{aligned}$$

The step (i) follows from the fact that $\alpha_{2,0}$ is independent of $\alpha_{1,0}$. Hence, U_2 is independent of U_1; however $U_2 = \alpha_{2,0} + \alpha_{1,0}$ and $U_1 = \alpha_{1,0}$. Since $\alpha_{1,0}$ and $\alpha_{2,0}$ are mutually independent, interchanging them does not change the result; hence, $\sum_{l=1}^{2} \alpha_{l,0}$ is independent of $\alpha_{1,0}$ as well as $\alpha_{2,0}$.

Having illustrated the proof for two variables, lets prove the result for the case that $i = n$, when $\theta_0 = U_n$. We first prove that θ_0 is uniformly distributed

and then show it is independent of any subset of $(n-1)$ $\alpha's$. For simplicity, we define the notation that $\{Y=y\} = \{\alpha_{i_1,0} = s_{i_1}, \cdots, \alpha_{i_{n-1},0} = s_{i_{n-1}}\}$.

$$P(\theta_0 = k) = P(U_n = k) \tag{26}$$

$$= \sum_{s_1=0}^{p-1} \cdots \sum_{s_{n-1}=0}^{p-1} P(U_n = k|Y=y)P(Y=y)$$

$$= \sum_{s_1=0}^{p-1} \cdots \sum_{s_{n-1}=0}^{p-1} P(Y=y) \prod_{l=1}^{n-1} P(\alpha_{l,0} = s_l) \tag{27}$$

$$= \sum_{s_1=0}^{p-1} \cdots \sum_{s_{n-1}=0}^{p-1} \{ \sum_{s_{n-1}=0}^{p-1} P(\alpha_{i_n,0} = k - \sum_{i=1}^{n-1} s_i)\}/p^{n-1} \tag{28}$$

$$= \frac{1}{p} \tag{29}$$

Hence, we note that θ_0 is uniformly distributed in the interval $[0, p-1]$. We now show that θ_0 is independent of any subset of $(n-1)$ $\alpha's$.

$$P(\theta_0 = k|Y=y) = P(U_n = k|\alpha_{i_1,0} = s_{i_1}, \cdots, \alpha_{i_{n-1},0} = s_{i_{n-1}}) \tag{30}$$

$$= P(\sum_{i=1}^{n} \alpha_{i,0} = k|Y=y) \tag{31}$$

$$= P(\alpha_{i_n,0} = k - \sum_{j=1}^{n-1} s_{i_j}|Y=y)$$

$$\stackrel{(i)}{=} P(\alpha_{i_n,0} = k - \sum_{j=1}^{n-1} s_{i_j}) \tag{32}$$

$$= \frac{1}{p} \tag{33}$$

$$= P(\theta_0 = k). \tag{34}$$

The step (i) uses the mutual independent property of the $\alpha's$. Note that the order of picking the $\alpha's$ was random. Hence, θ_0 is independent of any arbitrary subset consisting $(n-1)$ $\alpha's$. We now state the following property of the PCB scheme as a theorem.

Theorem 3. If random variables $FK_{i,j}$ are mutually independent and uniformly distributed in the interval $[0, p-1]$, θ_{j+1}, defined by

$$\theta_j = \sum_{i=1}^{n} FK_{i,j} \tag{35}$$

is uniform in the interval $[0, p-1]$ and is mutually independent with respect to any subset consisting of $(n-1)$ of the random variables $FK_{i,0}; i = 1, \cdots, n$.

Proof: Follows the similar inductive argument as above with $\alpha_{i,0}$ replaced with $FK_{i,j}$ and θ_0 replaced with θ_j.

The above theorems show that observing any $(n-1)$ fractional keys does not reveal any information about the group key. Hence, an adversary needs to know all n fractional keys to compute the group key θ at any iteration. In terms of the mutual information, we can write

$$I(\theta_j \wedge FK_{i_1,j}, \cdots, FK_{i_{n-1},j}) = 0, \tag{36}$$

where the subset of $(n-1)$ fractional keys are chosen arbitrarily.

Theorem 4. The intermediate pads $\alpha_{i,j}$, computed using the formula

$$\alpha_{i,j} = \theta_j + (p-1)FK_{i,j} \text{mod p} \tag{37}$$

satisfy the property $I(\alpha_{i,j} \wedge FK_{i,j}) = 0, \qquad i \in \{1, 2, \cdots, n\}$.

Proof:

$$\begin{aligned} I(\alpha_{i,j} \wedge FK_{i,j}) &= H(FK_{i,j}) - H(FK_{i,j}|\alpha_{i,j}) && (38)\\ &\overset{(i)}{=} H(FK_{i,j}) - H(FK_{i,j}) && (39)\\ &= 0. && (40) \end{aligned}$$

The step (i) follows from the fact that all $FK'_{i,j}s$ are mutually independent, and hence $FK_{l,j}$ is independent of the sum of $\alpha_{i,j} = \sum_{l=1;l\neq i}^{n} FK_{l,j}$.

Theorem 5. If the initial parameters $\alpha'_{i,0}s$ as well as the Fractional Keys $FK'_{i,j}s$ at every computational round j are mutually independent and are uniformly distributed in the interval $[0, p-1]$, then $\forall j$ $\theta'_j s$ then the $\theta'_j s$ are uncorrelated.

Proof: We first show that $I(\theta_j \wedge \theta_m) = 0$ for any arbitrary j, m.

$$\begin{aligned} I(\theta_j \wedge \theta_m) &= H(\theta_j) - H(\theta_j|\theta_m) && (41)\\ &= H(\sum_{i=1}^{n} FK_{i,j}) - H(\sum_{i=1}^{n} FK_{i,j}|\sum_{i=1}^{n} FK_{i,m}) && (42)\\ &\overset{(i)}{=} H(\sum_{i=1}^{n} FK_{i,j}) - H(\sum_{i=1}^{n} FK_{i,j}) && (43)\\ &= 0. && (44) \end{aligned}$$

The step (i) follows from the fact that given random variables $FK_{1,j}, \cdots, FK_{n,j}$ as well as $FK_{1,m}, \cdots, FK_{n,m}$ that are mutually independent, any function $f(FK_{1,j}, \cdots, FK_{n,j})$ of random variables $FK_{1,j}, \cdots FK_{n,j}$ is independent of any function $g(FK_{1,m}, \cdots, FK_{n,m})$ of random variables. For clarity, we use the following notations: $\{Z\} = \{\sum_{i=1}^{n} FK_{i,i_1}, \cdots \sum_{i=1}^{n} FK_{i,i_m}\}$.

In order to prove the general case considering the mutual information between a given θ_j and a set $S = \{\theta_{i_1}, \theta_{i_2}, \cdots, \theta_{i_m}\}$ where $\theta_j \in S$. We claim that

$$I(\theta_j \wedge \theta_{i_1}, \theta_{i_2}, \cdots, \theta_{i_m}) = 0. \tag{45}$$

Proof: The proof is similar to the case above but will be presented for completeness.

$$\begin{aligned} I(\theta_j \wedge \theta_{i_1}, \cdots, \theta_{i_m}) &= H(\theta_j) - H(\theta_j|\theta_{i_1}, \cdots, \theta_{i_m}) && (46)\\ &= H(\sum_{i=1}^{n} FK_{i,j}) - H(\sum_{i=1}^{n} FK_{i,j}|Z) && (47)\\ &\overset{(i)}{=} H(\sum_{i=1}^{n} FK_{i,j}) - H(\sum_{i=1}^{n} FK_{i,j}) && (48)\\ &= 0. && (49) \end{aligned}$$

Again, the step (i) follows from the fact that given random variables $FK_{1,j}, \cdots FK_{n,j}$ as well as $S = \{FK_{1,i_h}, \cdots, FK_{n,i_h}\}_{h=1}^{m}$ that are mutually independent, any function $f(FK_{1,j}, \cdots, FK_{n,j})$ of random variables $FK_{1,j}, \cdots, FK_{n,j}$ is independent of any function $g(FK_{1,i_h}, \cdots, FK_{n,i_m})$ of random variables.

5 Extensions and Complexity

Not all wireless can be represented by a pure broadcast model. Many networks use multi-hop communications as well as directional antennas. The impact of directional antennas and multi-hop communications changes the communication complexity of distributed key generation for some algorithms more than others.

In this section we describe alternative PCB algorithms better tailored for some non-broadcast wireless networks. These alternative algorithms are motivated by point-to-point communications in wireless network. A point-to-point model corresponds to scenarios such as a group of widely distributed members communicating using cell phones, or a localized group communicating using pencil beam directional antennas.

We explore three alternative algorithms for distributed key generation based on hypercube, octopus, and tree structures. We then analyze the communication complexity of the original PCB algorithm, broadcast-enhanced PCB and the alternative algorithms. Our analysis has shown that for the point-to-point network, these alternative algorithms have lower communication complexity than either the original or broadcast-enhanced versions of the PCB algorithm. The broadcast-enhanced PCB algorithm has lower complexity than any other algorithm in a pure broadcast network while the original PCB algorithm has the highest complexity.

Each of the alternative algorithms uses the same initialization phase as the original and broadcast PCB algorithms.

5.1 Hypercube

For simplicity we assume that the group has size $n = 2^r$. Each group member has an identifier i in the range $0, \ldots, n-1$. In a hypercube, two nodes are connected if their identifiers, represented as binary strings, differ in precisely one position. In the hypercube algorithm, during phase $k = 0, \ldots, r-1$, each group member communicates with the group member whose identifier differs only in the k^{th} position. After all r phases, each node will have sent and received a message from those group members with which it shares an edge of the hypercube. See Fig. 2.

Hypercube Algorithm — At the iteration step j, a participant M_i performs the following operations to generate its share of the distributed key:

1. M_i generates a Fractional Key $FK_{i,j}$.
2. M_i generates a Hidden Fractional Key $HFK_{i,j} = FK_{i,j} + \alpha_{i,j-1}$.
3. For the first set of exchanges in step j, which we call phase $k = 0$,

$$M_i \longrightarrow M_{(\hat{i}=bin(i)\otimes bin(2^{k=0}))} : E_{\theta_{j-1}}(KK_{\hat{i},j,0} = HFK_{i,j})$$

where $bin(t)$ is the r-bit binary representation of t and $\otimes$ is the exclusive-or operation. Member M_i then computes $TK_{i,j,1} = KK_{i,j,0} + HFK_{i,j}$.
For phases $k = 1, \ldots, r-1$, of step j,

$$M_i \longrightarrow M_{(\hat{i}=bin(i)\otimes bin(2^{k=0}))} : E_{\theta_{j-1}}(KK_{\hat{i},j,k} = TK_{i,j,k-1})$$

Member M_i then computes $TK_{i,j,k} = KK_{i,j,k-1} + TK_{i,j,k-1}$.
Phase 2 of the hypercube algorithm is shown in Fig. 2.

Once the r phases of the exchange are complete, a participant M_i has its combined shares, $\sum_{l=1}^{n} HFK_{l,j} = TK_{i,j,r-1}$. M_i then computes the group key and the fresh one-time pad for its computations. M_i performs the following operations:

1. Compute the new group key as

$$\theta_j = \sum_{l=1}^{n} HFK_{l,j} + (p-1)\alpha_{l,j-1} = \sum_{l=1}^{n} FK_{l,j} \text{ mod p.}$$

2. Compute $\alpha_{i,j} = \theta_j + (p-1)FK_{i,j}$ mod p.

5.2 Octopus-d

The hypercube algorithm provides substantially lower communication complexity than either the original PCB or the broadcast-enhanced PCB algorithms. Further improvement can be achieved by using an octopus network [BW]. An octopus consists of a d-dimension hypercube connecting a core subset of the group with each core member directly connected to a $(2^r - 2^d)/2^d$

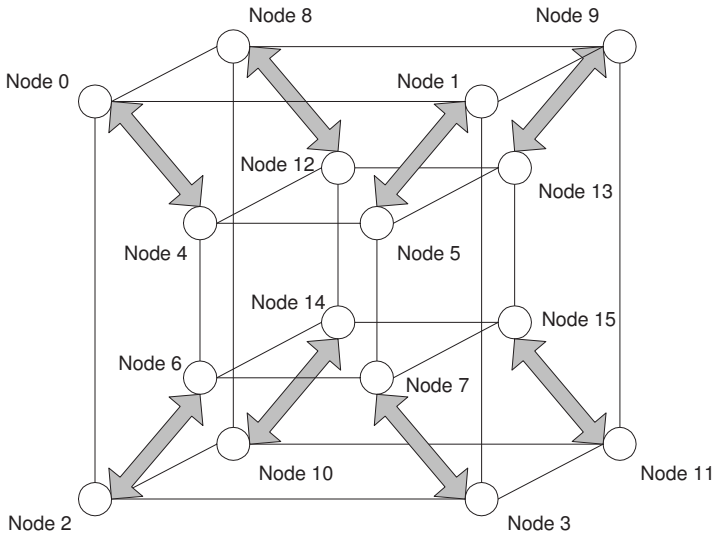

Fig. 2. The Hypercube Key Generation Algorithm with Point-to-Point Communications

size subset of the non-core group members. In Fig. 3 a d-dimension hypercube ($d = 2$) is used interconnect 2^d core group members of a group of size $2^r = 16$. Note that if $d = 0$ the octopus network collapses into a star network with a single group member connected to the other $2^r - 1$ members.

In the octopus-d algorithm each iteration has three passes. During the first pass each non-core group member transmits its key share to its corresponding core node. In the second pass the core members perform the exchanges of the hypercube algorithm. During the third pass each core node passes the sum of the $HFK_{i,j}$ to its corresponding non-core nodes. See Fig. 3.

Octopus-d Algorithm — In the algorithm each core node is numbered according to the hypercube algorithm. Each non-code node is numbered by multiplying the identifier of its corresponding core node by the number of core nodes, 2^d, and adding an index value which runs from 1 to $2^d - 1$.

At the iteration step j, a participant M_i performs the following operations to generate its share of the distributed key:

1. M_i generates a Fractional Key $FK_{i,j}$.
2. M_i generates a Hidden Fractional Key $HFK_{i,j} = FK_{i,j} + \alpha_{i,j-1}$.
3. Exchanges
   Pass One — If $2^d \leq i \leq 2^r - 1$,

$$M_i \longrightarrow M_{core(i)} : E_{\theta_{j-1}}(HFK_{i,j})$$

   where $core(i)$ is the core group member of i. Pass one communications are shown in Fig. 3. At the end of pass one each core node computes the

sum of its $HFK_{i,j}$ and those of its dependent non-core group members. Core member i computes

$$KK_{i,j,0} = HFK_{i,j} + \sum_{l=i\cdot 2^d+1}^{i\cdot 2^d+2^d-1} HFK_{l,j}.$$

Pass Two — Use a modified version of the exchanges of the hypercube algorithm on the core group members of the octopus. In the octopus the values $KK_{i,j,0}$ are distributed in the first set of exchanges, instead of $HFK_{i,j}$ used by the standard hypercube algorithm.
Pass Three — If member i is a core member, then depending on the communication model M_i broadcasts:

$$M_i \longrightarrow * : E_{\theta_{j-1}}(TK_{i,j,d-1})$$

or M_i uses point-to-point messages to exchange $E_{\theta_{j-1}}(TK_{i,j,d-1})$

$$M_i \longrightarrow M_{(dependent_k(i))} : E_{\theta_{j-1}}(TK_{i,j,d-1})$$

where $dependent_k(i)$ is the k^{th} dependent of member i. This phase is shown in Fig. 3.

Once the exchanges of this iteration are complete, a participant M_i has its combined shares, $\sum_{l=1}^{n} HFK_{l,j} = TK_{i,j,r-1}$. M_i then computes the group key and the fresh one-time pad for its computations. M_i performs the following operations:

1. Compute the new group key as

$$\theta_j = \sum_{l=1}^{n} HFK_{l,j} + (p-1)\alpha_{l,j-1} = \sum_{l=1}^{n} FK_{l,j} \bmod \text{p}.$$

2. Compute $\alpha_{i,j} = \theta_j + (p-1)FK_{i,j} \bmod \text{p}$.

5.3 Binary Tree

For simplicity we assume that the group has $n = 2^r - 1$ group members. Each group member has an identifier i in the range $0, \ldots, n-1$ and is a node (interior node or leaf node) of a binary tree. The group members are numbered in order of a preorder tree traversal. Group member 0 is the root of the tree, group member 1 is the left sibling of the root, member 2 is the right sibling of the root, and so on.

Binary Tree Algorithm — In the tree algorithm each iteration has two passes. During the first pass each node of the tree (working from the leaf nodes up toward the root) communicates the sum of the Hidden Fraction Key

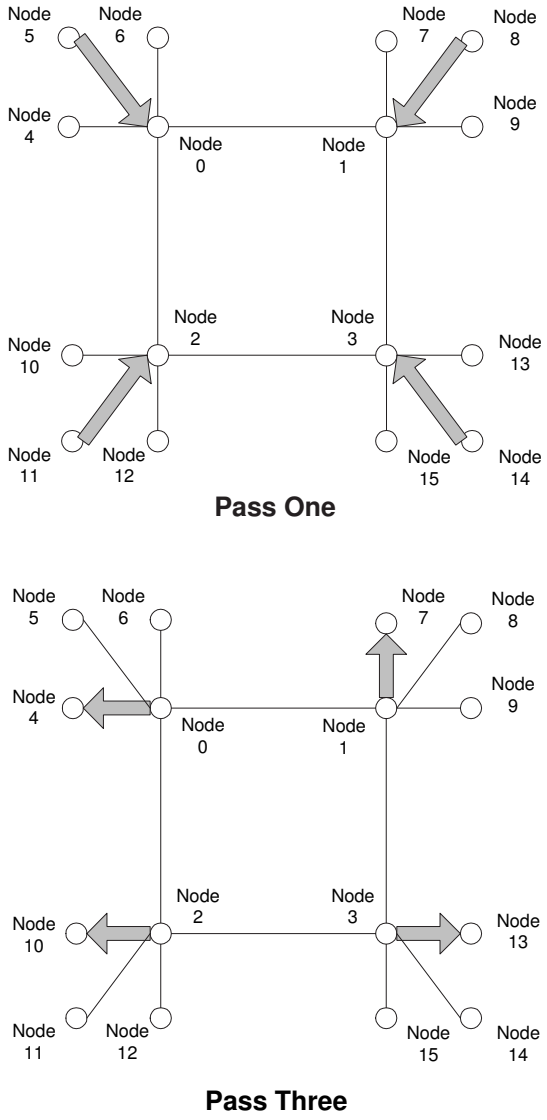

Fig. 3. The Octopus-d Key Generation Algorithm with Point-to-Point Communications, (r = 4, d = 2)

of all of the node's decedents to its parent. During the second pass the sum of the Hidden Fractional Keys for the group is distributed by the root. In the point-to-point model each node of the tree communicates with its children, working from the root down toward the leaf nodes. In the broadcast model the root distributes the sum directly to each member using a single broadcast. See Fig. 4.

At the iteration step j, a participant M_i performs the following operations to generate its share of the distributed key:

1. M_i generates a Fractional Key $FK_{i,j}$.
2. M_i generates a Hidden Fractional Key $HFK_{i,j} = FK_{i,j} + \alpha_{i,j-1}$.
3. Exchanges
   Pass One — Pass one propagates Hidden Fractional Keys to the root of the tree. If M_i is a leaf node, i.e., M_I is represented by a level 0 node then

$$i \longrightarrow parent(i) : E_{\theta_{j-1}}(HFK_{i,j})$$

   where $parent(i)$ the group member who is represented by the parent node of node i. Group member $M_{parent(i)}$ then computes $KK_{parent(i),j,1} = HFK_{i,j} + HFK_{sibling(i),j} + HFK_{parent(i),j}$.
   If M_i is represented by an level k interior node, it must wait until it can compute $KK_{i,j,k} = KK_{left\_decendent(i),j,k-1} + KK_{right\_decendent(i),j,k-1} + HFK_{parent(i),j}$. If M_i is not represented by the root of the tree then it sends $KK_{i,j,k}$ to its parent, i.e.,

$$i \longrightarrow parent(i) : E_{\theta_{j-1}}(KK_{i,j,k})$$

   Group member $M_{parent(i)}$ then computes $KK_{parent(i),j,k+1} = KK_{i,j,k} + KK_{sibling(i),j,k} + HFK_{parent(i),j}$. Pass one is shown in Fig. 4.
   Pass Two — In the broadcast communication model the root of the tree broadcasts:

$$M_0 \longrightarrow * : E_{\theta_{j-1}}(KK_{0,j,r-1})$$

   In the point to point model the root can distribute $KK_{0,j,r-1}$ to the group using the tree. Each non-leaf, non-root node M_i receives $KK_{0,j,r-1}$ from its parent and then distributes to its decedents by

$$i \longrightarrow left\_child(i) : E_{\theta_{j-1}}(KK_{i,j,r-1})$$

$$i \longrightarrow right\_child(i) : E_{\theta_{j-1}}(KK_{i,j,r-1})$$

Once the exchanges of this iteration are complete, a participant M_i has its combined shares, $\sum_{l=1}^{n} HFK_{l,j} = KK_{i,j,r-1}$. M_i then computes the group key and the fresh one-time pad for its computations. M_i performs the following operations:

1. Compute the new group key as

$$\theta_j = \sum_{l=1}^{n} HFK_{l,j} + (p-1)\alpha_{l,j-1} = \sum_{l=1}^{n} FK_{l,j} \bmod \text{p}.$$

2. Compute $\alpha_{i,j} = \theta_j + (p-1)FK_{i,j} \bmod \text{p}$.

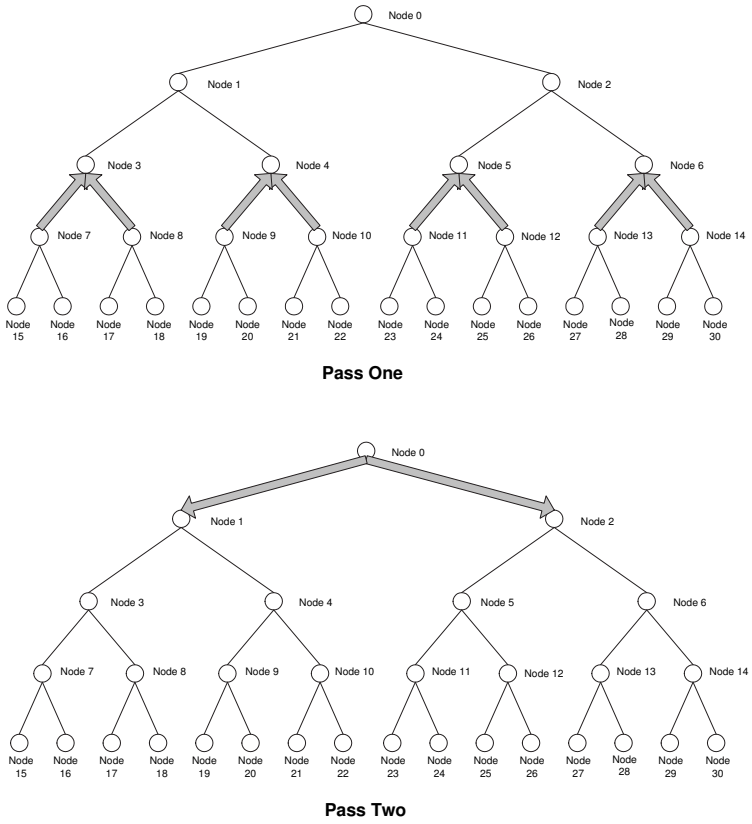

Fig. 4. The Tree Key Generation Algorithm with Point-to-Point Communications

5.4 Comparison

The following tables compare the communication complexity of the algorithms in the pure broadcast and pure point-to-point models. For each combination

of algorithm and model, we give the average number of messages (ignoring commitments) that each group member sends and receives as well as the maximum number of messages sent and received by a group member.

Algorithm	Group Size	Ave. Trans. per Member	Ave. Recv. per Member	Max. Trans.	Max. Recv.
Orginal PCB	2^r	$2^r - 1$	$(2^r - 1)^2$	$2^r - 1$	$(2^r - 1)^2$
Bcast. PCB	2^r	1	$2^r - 1$	1	$2^r - 1$
Hypercube	2^r	r	$r \cdot (2^r - 1)$	r	$r \cdot (2^r - 1)$
Octopus-d	2^r	$1 + \frac{d}{2^{r-d}}$	$2^r + d \cdot 2^d - 1 - \frac{d-1}{2^{r-d}}$	$d + 1$	$2^d \cdot d + 2^r - 1$
Tree	$2^r - 1$	1	$2^r - 2$	1	$2^r - 2$

Table 1. Key Generation Communication Costs — Broadcast

Algorithm	Group Size	Ave. Trans. per Member	Ave. Recv. per Member	Max. Trans.	Max. Recv.
Orginal PCB	2^r	$2^r - 1$	$2^r - 1$	$2^r - 1$	$2^r - 1$
Bcast. PCB	2^r	$2^r - 1$	$2^r - 1$	$2^r - 1$	$2^r - 1$
Hypercube	2^r	r	r	r	r
Octopus-d	2^r	$2 + \frac{d-2}{2^{r-d}}$	$2 + \frac{d-2}{2^{r-d}}$	$2^{r-d} + d - 1$	$2^{r-d} + d - 1$
Tree	$2^r - 1$	$2 + \frac{2}{2^r - 1}$	$2 + \frac{2}{2^r - 1}$	3	3

Table 2. Key Generation Communication Costs — Point-to-Point

Acknowledgements

Authors thank Mingyan Li for useful comments. RP gratefully acknowledges the support of the NSF under ANI-0093187, ARO under DAAD-19-02-1-0242 and ONR under N00014-04-1-0479. Both authors also acknowledge the support of ARL under DAAD19-01-2-0011.[3]

[3] This document was prepared through collaborative participation in the Communications and Networks Consortium sponsored by the U. S. Army Research Laboratory under the Collaborative Technology Alliance Program, DAAD19-01-2-0011. The U. S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U. S. Government.

References

[BW] Becker, K. and Wille, U.: Communication Complexity of Group Key Distribution, in *Proceedings of 5th ACM Conference on Compiuter and Communications Security*, ACM, New York, 1998.

[CT] Cover, T. and Thomas, J.: *Elements of Information Theory*, Wiley Interscience, New York, 1991.

[CS] Shannon, C.: Communication Theory of Secrecy Systems, *Bell Systems Technical Journal*, **28**(1949), 656-715.

[DS] Stinson, D.: *Cryptography: Theory and Practice*, CRC Press, New York, 2002.

[PCB] Poovendran, R., Corson, S., and Baras, J.: A Distributed Shared Key Generation Procedure using Fractional Keys , in *Proceedings of IEEE Milcom*, IEEE, New York, 1998.