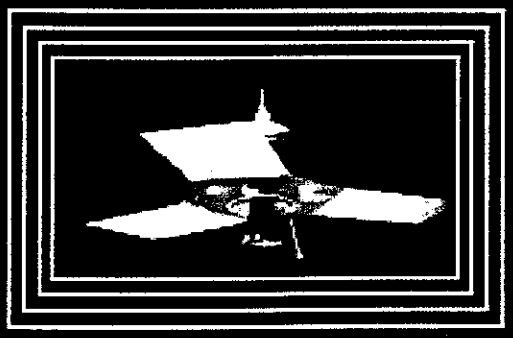

Advanced Telecommunications & Information Distribution

CONSORTIUM

ARL Federated Laboratory 4th Annual Symposium

March 21-23, 2000 College Park, MD

PROCEEDINGS

ANALYSIS AND DESIGN OF ROBUST KEY SCHEMES FOR MULTICAST COMMUNICATIONS *

R. Poovendran and J. S. Baras
Center for Satellite and Hybrid Communication Networks
Department of Electrical and Computer Engineering and the Institute for Systems Research
University of Maryland, College Park, MD 20742

ABSTRACT

Recent literature presents several rooted tree based member deletion/revocation schemes [5, 6, 7, 8, 1, 2] trying to simultaneously minimize the key storage while providing efficient member deletion/revocation. Many of these approaches have different solutions and provide different values for the number of keys to be stored and distributed. In this paper, we show that these problems can be systematically studied using basic concepts from information theory. In particular, we show that the entropy of member revocation event, plays a major role in defining the key allocation requirements. We then relate the entropy of member revocation event to bounds on the key length. We also show that an optimal Huffman coding strategy used in [7, 8] leads to security weaknesses. A method for generating key management schemes to withstand varying degrees of member collusion is also presented.

Keywords: Multicast Security, Collusion, Member Deletion/Revocation, Key Length, Entropy.

1. INTRODUCTION

Many of the distributed applications like Internet newscast, stock quote updates, and distributed conferencing may benefit from secure group communications. Providing an effective key management scheme for these applications is complicated by the nature of a group that is *netgraphically* distributed and exhibits varying degrees of trust, i.e. different parts of the group may have different security strengths that can be assumed in key management. In heterogeneous military networks, group communication and multicast are of paramount importance. Security, and efficient ways to achieve it in the bandwidth limited environment of the battlefield, are critical problem.

A centralized Group Controller (GC) is assumed to be responsible for distributing all the required keys to the group members. In general, two or more members can use their public keys to communicate. However, if the number of messages and the number of members participating in the communication are very large, it is convenient and efficient to use shared keys. The key that is used for session encryption by the participating members is called the Session Key (SK). If the SK needs to be updated over a period of time for a variety of reasons including key lifetime expiration, compromise of the key, and/or temporary failure of one of the members, there has to be a mechanism to securely update the SK of all valid members. Although the use of public keys is one approach to achieve this goal, a common shared key called the Key Encrypting Key (KEK) can be used te reduce the computations at the sender node. Instead of using a single KEK, each member is given a variable number of KEKs for broadcast efficiency while optimizing the user key storage requirements. The main focus of our research has been to find efficient key distribution schemes that minimize the user key storage requirements without introducing vulnerabilities such as user collusion.

Since there is more than one member involved in the communications, the group size may vary during the session due to a variety of reasons. In order to protect communication integrity under group dynamics, the session key may have to be updated due to any of the following reasons:

- Expiration of the lifetime of the session key.
- Join/Admission of a member.
- Deletion/Revocation of a member.
- Voluntary leave of a group member.

Recently, a series of papers utilizing rooted-trees for key distribution have been proposed to minimize the

*Prepared through collaborative participation in the Advanced Telecommunications/Information Distribution Research Program (ATIRP) Consortium sponsored by the U.S. Army Research Laboratory under the Federal Research Laboratory Program, cooperative agreement DAAL01-96-2-0002 and under NASA cooperative agreement NCC3-528.

storage at the group controller and the members while providing a reduction in the amount of encryptions required to update the session key [1, 2, 5, 6, 7, 8].

The main contributions of this paper are the following:

- We show that it is possible to unify these approaches using a common analysis technique.
- We show that the *efficient* key distribution approach that minimizes user key storage can be formulated as an optimization problem.
- We also show that the design of an optimal tree is closely related to the Huffman trees and the *entropy of member revocation event.*
- We then show that this entropy provides a bound on the average length of the key provided, if all the keys are of same length.
- We perform weakness analysis using entropy and show that some of the schemes ([7, 8]) have security vulnerabilities.

As a concrete illustration, Figure 1 presents a KEK distribution based on a binary rooted tree for 16 members. In this approach, each leaf of the tree represents a unique member of the group; i.e. the leaves are in a one-to-one correspondence with members. Each node of the tree represents a key. The set of keys along the path from the root to a particular leaf node are assigned to the member represented by that leaf node. For example, member M_1 in Figure 1 is assigned KEKs $\{K_O, K_{2.1}, K_{1.1}, K_{0.1}\}$.

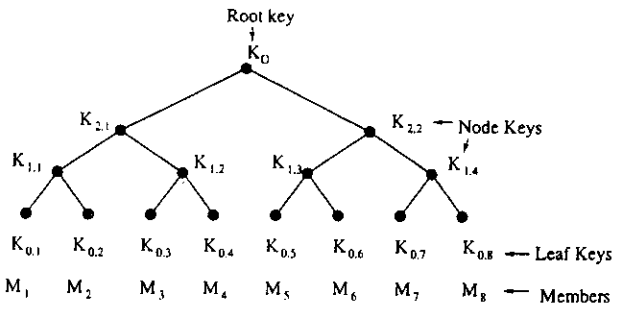

Figure 1: The Logical Key Tree of [1, 5, 6, 7, 8]

In order to be able to selectively disseminate information to a subset of group members, the GC has to ensure that the common key assigned to a subset is not assigned to any member not belonging to that subset. Using the notation $\{m\}_K$ to denote the encryption of m with key K, and the notation A $\longrightarrow$ B : $\{m\}_K$ to denote the secure exchange of message m from A to B, the GC can selectively send a message m to members five through eight by the following transmission:
GC $\longrightarrow M_5, M_6, M_7, M_8 : \{m\}_{K_{2.2}}$

If, however the key $K_{2.2}$ is invalidated for any reason, the GC needs to update the key $K_{2.2}$ before being able to use a common key for members M_5, M_6, M_7, and M_8. It can do so by first generating a new version of $K_{2.2}$, and then performing two encryptions, one with $K_{1.3}$ and the other with $K_{1.4}$.

2. MEMBER REVOCATION IN ROOTED TREES

The following observations can be made towards the rooted tree based key distributions.

- Since each member is assigned $(2 + \log_d N) = \log_d Nd^2$ keys, deletion of a single member requires $(2 + \log_d N)$ keys to be invalidated.
- The GC needs to perform at least $\log_d N$ encryptions per member removal.
- For a $d-ary$ tree with depth $h = \log_d N$, the GC has to store $1 + 1 + d + d^2 + \cdots + d^h = \frac{d(N+1)-2}{(d-1)}$ number of keys.

2.1. REACHABILITY AND THE KRAFT INEQUALITY

At the time of member revocation, the GC has to be able to uniquely identify the set of keys assigned to the revoked member and invalidate the keys. After revoking a member, securely reaching the rest of the group requires that the valid member has one or more keys that are not in the set of keys assigned to the revoked member. We will refer to the ability of the GC to reach the valid members under user revocation as *reachability.* Unlike other works that emphasize UID, we note that the KID plays a major role since it is the keys that need to be invalidated and generated.

One important necessary condition for the rooted tree based key assignment is that the KID of any member should not be a prefix *of the KID of any other member. On the rooted-tree, this leads to the well known Kraft inequality given below.*

Theorem 1: *Kraft Inequality for KIDs*
For a $d-ary$ rooted key tree with N members and KIDs satisfying the prefix condition, if we denote the number of keys assigned for member i by l_i, the sequence $\{l_1, l_2, \cdots l_N\}$ satisfies the Kraft inequality given by

$$\sum_{i=1}^{N} d^{-l_i} \leq 1. \qquad (1)$$

Conversely, given a set of numbers $\{l_1, l_2, \cdots l_N\}$ satisfying this inequality, there is a rooted tree that can be constructed such that each member has a unique KID with no-prefixing.

Proof: Well known and available in [3, 4].

3. PROBABILISTIC MODELING OF MEMBER REVOCATION

Since the key updates are performed in response to member revocation, modeling and analysis of the statistics of *member revocation event*, appropriate for system design and performance characterization. We denote by p_i the probability of revocation of member i.

3.1 Relating the Probability of Member Revocation to the Keys on the Rooted Tree

The process of member revocation is related to the rooted trees via the leaf nodes. Since every member has a unique KID and the KIDs are formed by concatenating the keys assigned to a member, when the member is deleted/revoked the KID is also revoked/invalidated. Hence, the event of member revocation is equivalent to the event of revocation/invalidation of the KID of a member.

The following assumptions are implicit in the models presented in [5, 7, 8], and are useful in the derivation of the optimal number of keys to be assigned to each member.

- *Assumption 1:* Revocation of members is an independent event.
- *Assumption 2:* The number of members N is a fixed quantity.

Definition: We define the $d-ary$ entropy H_d of the member revocation event by

$$H_d = -\sum_{i=1}^{N} p_i \log_d p_i \tag{2}$$

where p_i is the probability of revocation of member i. As mentioned earlier, the entropy expresses the uncertainty as to which member will be revoked in $d-ary$ digits.

3.2 Assigning Optimal Number of Keys per Member

Since the SK and the root key are common to all the members, these two keys don't contribute to the optimal key assignment strategies. In formulating the problem of optimal number of keys per member, the GC should try to minimize the storage requirements without making any explicit assumptions about the nature of the keys to be chosen. If such a formulation is possible, then that optimal key assignment strategy may be used to relate the storage requirements to system parameters such as the probabilities of member revocation.

From the view point of GC, one strategy is to minimize the average number of keys per member with the additional conditions that the KIDs of the members should satisfy the Kraft inequality[1]. We summarize the result as Theorem 2 without repeating the proofs [3].

Theorem 2: For a key assignment satisfying the Kraft inequality, the optimal average number of keys, excluding the root key and the SK, held by a member is given by the $d-ary$ entropy $H_d = -\sum_{i=1}^{N} p_i \log_d p_i$ of the *member revocation* event. For a member i with probability of revocation p_i, satisfying the optimization criteria, the optimal number of keys l_i, excluding the root key and the KEK, is given by

$$l_i^* = -\log_d p_i. \tag{3}$$

Since the SK and the root key are common to all the members, the optimal average number of keys per member is given by $H_d + 2$, and the number of keys assigned to member i with revocation probability p_i, including the SK and the root key is given by

$$l_i^* + 2 = -\log_d p_i + 2 = \log_d \frac{d^2}{p_i}. \tag{4}$$

3.3 Maximum Entropy and Key Assignment

The results reported in [5, 6, 1, 8, 7] present a rooted tree with all members having the same number of keys. Since the optimal number of keys for a member i with probability or revocation p_i is $(2 - \log_d p_i)$, this assignment is equivalent to treating $(2 - \log_d p_i = \text{constant})$ for all values of i. Hence, the results in [5, 6, 8, 7, 1] assume that the probability of revocation is uniform for the entire group. Since the uniform distribution *maximizes* the entropy and entropy is the average number of keys per member under the optimal strategy, the schemes in [5, 6, 1] assign maximal set of keys per member. We summarize these results by the following theorem.

Theorem 3: The average number of keys per member is upper bounded by $(2 - \log_d N)$ and this value is reached when all the members have equal probabilities of being deleted/revoked.

4. BOUNDS ON AVERAGE KEY LENGTH

[1] Although the prefix strategy provides protection against only a single member failure, it is the only one that to our knowledge is mathematically viable to analysis.

When there is a member revocation, the average number of keys to be invalidated is given by $(2+H_d)$. If each key is L $d-ary$ digits long, then in order to update these keys, the total number of digits that need to be generated by the GC after member revocation is $L(2+H_d)$ digits. Since $H_d \leq \log_d N$ (with equality attained *iff* all the members have equal revocation probabilities), the hardware needs to be able to generate on average of $L(2+\log_d N)$ digits within the next unit of time for update to let the session continue. The following theorem summarizes this result.

Theorem 4: For a $d-ary$ rooted tree key distribution scheme in which each key is of length L digits, if the hardware digit generation rate is given by B, then the key length L is bounded by the following inequalities:

$$\frac{B}{2-\log_d p_{min}} \leq L \leq \frac{B}{2-\log_d p_{max}} \qquad (5)$$

5. SECURITY ANALYSIS OF SOME RECENT RESULTS USING MEMBER REVOCATION ENTROPY

We noted that the KID of one member should not be a subset of the KID of any other member. This condition ensures that the revocation of one member does not expose all the keys of a valid member. However, this is not the only case under which member revocation will expose the keys of valid member(s). It is possible that one or two members are simultaneously revoked or compromised. If the set of keys held by the revoked members can cover the set of keys held by one or more valid members, the corresponding keys of the valid members should be treated as exposed.

We assume that i, j, and k are three members of a larger group. They have the sets of keys $S_i = \{K_1, K_2, K_3\}$, $S_j = \{K_1, K_2, K_3, K_4, K_5, K_6\}$, $S_k = \{K_2, K_4, K_5, K_6, K_7\}$ respectively. It can be checked by inspection that none of the sets S_i, S_j and S_k are subsets of the other. Since $(2^{-3}+2^{-5}+2^{-4}) < 1$, by the converse of the Kraft inequality, we can also construct a binary tree for distributing the keys. This tree does ensure that each member has unique KID and if any one of the members is compromised, the group controller can still securely communicate with the other two members. However, if the members i and k are to collaborate, between them they can cover the keys of the member j. Not only can they cover the keys of member j, they can still ensure the integrity of their communication with the group controller if they don't expose the keys K_3 and K_7. Hence, we note that the Kraft KID satisfying prefix coding doesn't imply that the key assignment scheme is free of security vulnerabilities.

5.1 Description of the Schemes in [7, 8]

The authors in [7] noted that given the binary index of a member, each bit in the index takes two values, namely 0 or 1. To follow the example given in [7], when $N = 8$, $\log_2 8 = 3$ bits are needed to uniquely index *all* 8 members. The authors then proceeded to claim that since each bit takes two values, it can be *symbolically* mapped to a distinct pairs of keys. The table below reproduces the mapping between the ID bit # and the key mapping for the case in [7] for $N = 8$:

ID Bit #0	K_{00}	K_{01}
ID Bit #1	K_{10}	K_{11}
ID Bit #2	K_{20}	K_{21}

where, the key pair $(K_{i,0}, K_{i,1})$ symbolically represents the two possible values of the ith bit of the member index. Although this table does provide a one-to-one mapping between the set of keys and the member index using only eight keys, the problem with this approach becomes clear if we map the table to the rooted tree structure. Figure 3 shows the mapping of the keys on the tree. (For the sake of clarity, not all the keys corresponding to the leaves are shown in Figure 3). Adjacent leaves have K_{20}, K_{21} as the keys and this pair is repeated across the level. In fact, at any depth only two specific keys have been used and duplicated across the depth. This is a form of Huffman coding with 2 new alphabets being introduced at each level of the tree.

In approaches such as [7, 8] that use UID for optimal Huffman coding, a special case of member revocation brings these key management schemes to a halt. This happens if the members M_0 and M_7 need to be revoked. The corresponding keys to be revoked are shown in Figure 2. These two members have only the session key in common. However, if these two members need to be simultaneously revoked, the group controller is left with *no key* to securely communicate with the rest of the valid members. This reduces the rooted tree to the GKMP [9]. The compromise recovery for this case requires that the entire group re-key itself by contacting one member at a time.

5.2 Key Management Schemes with Varying Degree of Collusion

From our analysis of the tree based schemes, we note that many different key management schemes with different levels of protection against user collusion can be made. The user collusion being a set cover problem, it is related to the keys assigned to internal nodes. On one extreme, the keys representing the rooted tree have no

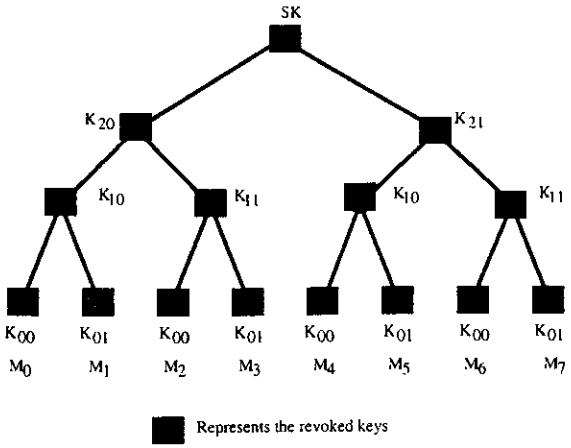

Figure 2: Revocation of Members M_0, M_7 in [7, 8].

relationship, leading to a very high degree of integrity but also higher storage requirements. On the other extreme, all members share the same keys as in GKMP [9] leading to system failure in the event of a single member failure. The schemes in [7, 8] fail with the collusion of two members or can fail at different bit levels depending on the index of the colluding members. Depending on how many digit locations are represented as $k-ary$ digits. Figure 3 shows the comparison between various schemes.

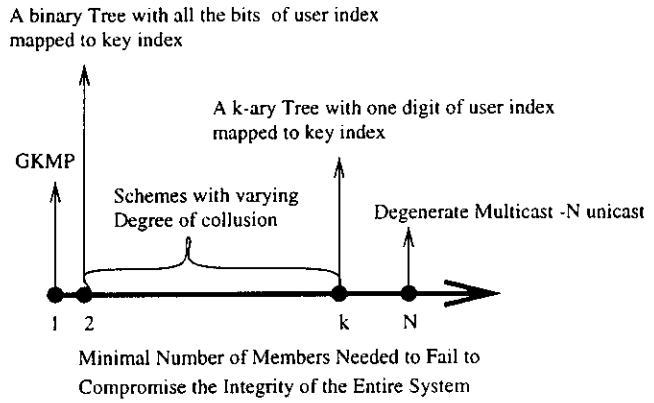

Figure 3: Effect of user failure on different schemes

CONCLUSIONS

We showed that the tree based key distribution schemes for secure multicast can be analyzed using a unified framework. We then showed that logical trees offer a solution to the problem of minimizing the user key storage. We then presented an optimization formulation for user key storage minimization problem. Using our results, we showed that the "entropy" of member revocation plays an important role in sustainable key length, optimal number of keys per member and user collusion.

We note that other efficiency parameters such as sender storage efficiency can be more interesting in the case of mobile devices. We are in the process of formulating problems that can manipulate the internal nodes of the tree to minimize the storage at the sender as well[2].

References

[1] R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor, B. Pinkas, "Multicast Security: A Taxonomy and Efficient Reconstructions", In *Proceedings of IEEE Infocom'99.*

[2] D. A. McGrew and A. Sherman, "Key Establishment in Large Dynamic Groups Using One-Way Function Trees", *Manuscript, 1998.*

[3] T. Cover, J. Thomas, Elements of Information Theory, John Wiley & Sons, Inc, NY, 1991.

[4] R. Gallager, Information theory and reliable communication, Wiley, NY, 1968.

[5] D. M. Wallner, E. C. Harder, and R. C. Agee, "Key Management for Multicast: Issues and Architectures", Internet Draft, September 1998.

[6] C. K. Wong, M. Gouda, S. S. Lam,"Secure Group Communications Using Key Graphs", In *Proceedings of ACM SIGCOMM'98*, September 2-4, Vancouver, Canada.

[7] G. Caronni, M. Waldvogel, D. Sun, and B. Plattner, "Efficient Security for Large and Dynamic Groups", In *Proc. of the Seventh Workshop on Enabling Technologies*, IEEE Computer Society Press, 1998.

[8] I. Chang, R. Engel, D. Kandlur, D. Pendarakis, D. Saha, "Key Management for Secure Internet Multicast Using Boolean Function Minimization Techniques", To apper in Proceedings of IEEE Infocom'99.

[9] H. Harney and C. Muckenhirn, "GKMP Architecture", *Request for Comments(RFC)* 2093, July 1997.

[2]The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied of the Army Research Laboratory or the U.S. Government.