# Key Trees and the Security of Interval Multicast

Mohamed G. Gouda  Chin-Tser Huang
Department of Computer Sciences
The University of Texas at Austin
Austin, Texas 78712-1188
{gouda, chuang}@cs.utexas.edu

E. N. Elnozahy
System Software Department
IBM Austin Research Lab
11400 Burnet Rd., M/S 9460, Austin, TX 78758
mootaz@us.ibm.com

## Abstract

*A key tree is a distributed data structure of security keys that can be used by a group of users. In this paper, we describe how any user in the group can use the different keys in the key tree to securely multicast data to different subgroups within the group. The cost of securely multicasting data to a subgroup whose users are "consecutive" is O(*log *n) encryptions, where n is the total number of users in the group. The cost of securely multicasting data to an arbitrary subgroup is O(n/2) encryptions. However, this cost can be reduced to one encryption by introducing an additional key tree to the group.*

## 1. Introduction

A key tree $T_n$ is a binary balanced tree with n leaves, where each node has a security key [2]. A group that consists of one server and n users can use a key tree $T_n$ as follows. First, the group server knows all the keys in $T_n$. Second, each user is associated with a leaf of $T_n$ and knows all the keys that are in the path from this leaf to the tree root. As discussed in [2], every user in the group can use its keys in $T_n$ to securely broadcast data to every other user in the group. In this paper, we describe how the keys in $T_n$ can be used to support secure multicast, rather than mere broadcast.

## 2. Interval Multicast Using Key Trees

A $T_n$-group consists of a group server s, a set $\{u_0, u_1, ..., u_{n-1}\}$ of n users, and a key tree $T_n$.

A user interval in a $T_n$-group is a subset of consecutive users in the set of users of the $T_n$-group. For example, consider a $T_8$-group whose set of users is $\{u_0, ..., u_7\}$. In this $T_8$-group, the subset $\{u_0, u_1, u_2\}$ is a user interval whereas the subset $\{u_1, u_3\}$ is not a user interval.

Let U[i, j], where $0 \leq$ i $\leq$ j $<$ n, denote the user interval $\{u_i, u_{i+1}, ..., u_j\}$ in a $T_n$-group. Note that the interval U[x, x] denotes the single user $u_x$.

Each key in a key tree $T_n$ corresponds to a user interval in the $T_n$-group, but not every user interval in the $T_n$-group corresponds to a key in the key tree $T_n$. We refer to any user interval in a $T_n$-group that corresponds to some key in the key tree $T_n$ as a basic interval in the $T_n$-group.

Let K[i, j] denote the key in a key tree $T_n$ that corresponds to the basic interval U[i, j] in the $T_n$-group.

For a user $u_x$ to securely multicast some data d to every user in a basic interval U[i, j] in a $T_n$-group, the following two-step protocol can be used. In the first step, $u_x$ sends a message consisting of two fields to server s of the $T_n$-group. The first field defines the message source, namely user U[x, x]. The second field is an encryption, using key K[x, x] of user U[x, x], of the concatenation of three items: the data d, the ultimate destination of data d, and a checksum computed over the other two items.

In the second step, server s broadcasts a message consisting of two fields to every user in the $T_n$-group. The first field defines the intended ultimate destination of the message, namely the user interval U[i, j]. The second field is an encryption, using key K[i, j] of the basic interval U[i, j], of the concatenation of three items: the data d, the original source of data d, and a checksum computed over the other two items. Note that although the broadcast message is sent to every user in the $T_n$-group, only users in the interval U[i, j] have the key K[i, j] and so only they can decrypt the message.

The above protocol suggests the following question. How can a user $u_x$ securely multicast some data d to every user in an interval U[i, j] that is not basic? As described in Algorithm 1 below, any user interval that is not basic can be partitioned into two or more non-overlapping basic intervals. Thus, if interval U[i, j] can be partitioned into two basic intervals U[i, y] and U[y+1, j], then user $u_x$ can securely multicast d to U[i, j] by securely multicasting d to each of the basic intervals U[i, y] and U[y+1, j] using the following three-step variation of the above protocol. In the first step, user $u_x$ sends the encrypted data to the group server s. Then in the next two steps, the group server s broadcasts two versions of the encrypted data to every user in the group. Each version is encrypted by a different key from the two keys of the basic interval U[i, y] and U[y+1, j]. Thus, the first version is encrypted by the key K[i, y], and the second version is encrypted by the key K[y+1, j].

The following algorithm can be used to partition any interval into one or more basic intervals in a $T_n$-group.

**Algorithm 1**

```
input   A user interval U[i, j] in a T_n-group where
        T_n is a binary key tree and n is a power of 2.
output  A set OUT of basic intervals in the T_n-group
        that partition U[i, j].
        Initially, OUT = the empty set
local   A set IN of basic intervals in the T_n-group.
        Initially, IN = {U[0, n-1]}
begin
0:      while U[x, y] is in IN do
1:          IN := IN - {U[x, y]};
2:          if   U[x, y] does not overlap with U[i, j]
3:          then skip
4:          else if   U[x, y] is contained in U[i, j]
5:               then OUT := OUT ∪ {U[x, y]}
                 else {U[x, y] intersects with U[i, j]}
6:                    partition U[x, y] into two basic
                      intervals U[x, z] and U[z+1, y];
7:                    IN := IN ∪ {U[x, z], U[z+1, y]}
                 endif
          endif
        endwhile
end
```

As an example, if Algorithm 1 is applied to the user interval [1, 6] in a $T_8$-group, then set OUT is computed as {[1, 1], [2, 3], [4, 5], [6, 6]}.

Observe that each of the basic intervals in set OUT, computed by Algorithm 1, is maximal. In other words, no two or more basic intervals in set OUT can be combined into a single basic interval. (For example, if the two basic intervals [2, 3] and [4, 5] in the above set OUT are combined into a single interval, then the resulting interval [2, 5] is not basic in the $T_8$-group.) From this feature, it follows that the set OUT computed by Algorithm 1 has the smallest possible number of elements. Thus, the encrypted message needs to be multicast by the group server to the fewest possible basic intervals in the group. These observations suggest the following definition.

A basic interval U[i, j] contained in a user interval U[x, y] is called a component of U[x, y] iff U[i, j] is not contained in any other basic interval that is contained in U[x, y].

Therefore, Algorithm 1 computes all the components of the input user interval U[i, j].

## 3. Complexity of Interval Multicast

The presentation in the previous section can be summarized as follows. For server s of a $T_n$-group to securely multicast some data d to a user interval U[i, j] in the $T_n$-group, server s needs to perform two steps. First, server s applies Algorithm 1 to interval U[i, j] and computes the components of U[i, j]. Second, server s broadcasts an encrypted version of the data d to every component of interval U[i, j]. Therefore, to establish an upper bound on the number of encryptions that server s needs to perform in order to securely multicast some data to an arbitrary user interval, we need to establish an upper bound on the number of components of a user interval. This upper bound can be established by the following lemma and theorem.

**Lemma 1:** In a $T_n$-group, each user interval has fewer or equal number of components as the user interval U[1, n-2].

**Theorem 1:** The server of a $T_n$-group needs to perform at most 2 $\log_2$ n -2 encryptions to securely multicast some data to any user interval in the $T_n$-group.

## 4. Extensions

The above discussion can be extended in two ways. First, the discussion can be extended to securely multicast data to an arbitrary subgroup of not necessarily consecutive users. Second, the discussion can be extended to key trees of any degree. Details of these two extensions can be found in [1].

## Acknowledgment

This work is supported in part by the DARPA contract F33615–01–C–1901, by two IBM Faculty Partnership Awards for the academic years of 2000-2001 and 2001-2002, and by the grant TARP 14–970823 provided by the Texas Higher Education Coordinating Board.

## References

[1] Gouda, M. G., C.-T. Huang, E. N. Elnozahy, Key Trees and the Security of Interval Multicast, Technical Report TR-02-18, Department of Computer Sciences, The University of Texas at Austin, Austin, Texas, March 2002.

[2] Wong, C. K., M. G. Gouda, S. S. Lam, Secure Group Communications Using Key Graphs, IEEE/ACM Transactions on Networking, Vol. 8, No. 1, February 2000.