A Scalable Multicast Key Management to Expel a Malicious Receiver in MARKS

Ma Xiaojing¹, Li Zhitang², Li Yao² and Zhang Jiping¹
1Department of Computer Science 2Network and Computing Center
Huazhong University of Science and Technology, 430074, Wuhan, China
lindahust@mail.hust.edu.cn

Abstract

An efficient multicast key management in large dynamic group with preplanned eviction is MARKS in which one member’s join or preplanned leave has no side effect on other members. The goal of this work is to expel a malicious receiver in MARKS scalably. A layered Logical Key Tree based on coincident leave time (called TR-LKH) is proposed here.

1. Introduction

We identify secure group communication as one of the important problems to solve for successful deployment of group-oriented applications. MARKS [1] focused on multicast issues with pre-planned eviction, such as pay-per-view. It slices the multicast data into many “application data unit” (ADU) with the same time-length. A multicast session flow of MARKS [1] is briefly described as follows.

a. The sender constructs a Binary Hash Tree (BHT) with depth \(H = \lceil \log T \rceil\) from one initial seed value \(S_0\). \(T\) is the number of the time slices. The intermediate seeds are calculated by applying respectively the blinding functions \(l()\) and \(r()\) to their parent. In Fig. 1, \(T=4, H=2\); the ADU key sequence is \(K_1, K_2, K_3, K_4\).

b. Each valid receiver can get some intermediate seeds which can be used to construct sub-sequence ADU keys. For example, \(S_{21}\) and \(S_{11}\) are distributed to the receiver who requests for ADU2 to ADU4. The receiver can get \(K_2=S_{21}, K_3=l(S_{11})\) and \(K_4=r(S_{11})\).

c. The sender multicasts the ADUi encrypted by the corresponding key \(K_i\), the receivers can decrypt the data.

In MARKS, a receiver cannot be eliminated without rekeying the entire group. Briscoe et al [1] suggested combining (e.g. XOR) each ADU key with an intermediate key (IK) to get the Traffic Encryption Keys (TEK). Then the TEK_i is used to encrypt/decrypt the i-th ADU. Thus, a receiver would be expelled by informing all members the IK'(new IK) except him.

2. Our approach

2.1. TR-LKH

TR-LKH is composed by sub trees on two layers. The sub tree on the upper layer is called “time logical key tree” (T-LKH) whose number of leaf nodes is equal to the number of the time slices. The sub trees on the lower layer called “receiver’s logical key tree” (R-LKH). Every node of R-LKH, including intermediate node, is linked to a receiver. All receivers planning to leave at \(T_i\) are put on R-LKH one layer after another, and R-LKH_i is linked to the corresponding leaf node \(K_{Ti}\) of T-LKH.

2.2. Key management

We explain our key management operations along with the example in which \(T=4\) shown in Fig. 2. The operations are delegated to a Key manager (KM).

1. KM firstly constructs a BHT and a TR-LKH both with \(H\) high for \(T\) time slices, where \(H = \lceil \log T \rceil\).
2. When a receiver joins, KM firstly shares an individual key (SKRx) with the receiver, and then it distributes per valid receiver the IK, ADU keys which the receiver requested for and keys on the path from the receiver’s node to the TR-LKH root. In Fig. 2, KM
distributes \{IK,S10, S22, K14, K34, KT3, K1[1], K3[3], K3[6]\} to R3[6] who requests for ADU1 to ADU3.

3. When a receiver leaves pre-plannedly, the keys need not to be updated. The branches related to the receiver are logically pruned as the dashed line shows in Fig. 2. The logical root of TR-LKH is dynamically changed. The root shifts to the node K34 when the time is T3.

![Figure 2. Example of a TR-LKH](image)

4. We assume that Rex (the receiver who needs to be expelled) is granted access to time slices from ADUa to ADUb. For example, now is T2 and R3[6] needs to be expelled. Thus, IK used from T2 to T3 and the keys R3[6] knows on the TR-LKH need to be updated.

The multicast rekeying messages include:

1. One message of IK′ encrypted by K0+b+1: \(\{IK′\}K_b\);

2. \(m\) messages of the IK′ and the corresponding new keys on T-LKH encrypted by the intermediate keys on T-LKH corresponding to \(KT_{\text{now}} \sim KT(b-1)\) \((m\) is the number of these intermediate keys, \(1 \leq m \leq 2\left\lfloor \log N \right\rfloor -1\)); in the example, \(m=1\), the message is \(\{IK′,K1′4\}KT2\);,

3. \(q\) messages of the corresponding new keys on T-LKH encrypted by the intermediate keys on T-LKH corresponding to \(KT(b+1) \sim KT\) \((q\) is the number of these intermediate keys, \(1 \leq q \leq \left\lfloor \log T \right\rfloor\)); in the example, \(q=1\), the message is \(\{K1′4,K3′4\}KT4\);

4. \(h\) messages of the IK′ and the corresponding new keys on TR-LKH encrypted by the brother keys of nodes on the path from \(Rex\)’s node to the R-LKHb’s root \((0 \leq h \leq \left\lfloor \log n \right\rfloor, \ h\) is the receiver’s layer on its R-LKH, \(n_i\) is the number of users on R-LKHb\); in the example, \(h=2\), the messages are \(\{IK′,K1′4,K3′4,KT3′,K3[1]\}K3[2]\);

\(\{IK′,K1′4,K3′4,KT3′,K3[1]\}K3[3]\}K3[7]\);

5. Two messages of the IK′ and the corresponding new keys on TR-LKH tree encrypted by the children of the Rex if Rex is not on the leaf node.

The unicast rekeying messages include \(h\) messages of the IK′ and the corresponding new keys on TR-LKH encrypted by the individual keys of each receiver on the path from \(Rex\)’s node to the R-LKHb’s root. \(\{IK′,K1′4,K3′4,KT3′,K3[1]\}SKR3[1]\);

\(\{IK′,K1′4,K3′4,KT3′,K3[1]\}K3[3]\}SKR3[3]\);

All the rekey messages are encrypted by the keys that are unknown to \(Rex\), and all keys that \(Rex\) knows are refreshed. So \(Rex\) can not get IK′, \(Rex\) is deleted.

3. Efficiency

In our scheme, one member’s join or preplanned leave has no side effect on other members. Furthermore, the number of rekeying messages for expelling a receiver has little dependence on the group size.

4. Security

Backward Security Since a receiver cannot get the following ADU keys after his/her pre-planned leave; he/she cannot decrypt the following ADUs.

Forward Security When a receiver joins, he/she cannot get the former ADU keys, and will not get the former TEKs to decrypt the corresponding ADUs.

Collusion After a receiver’s pre-planned leave; all the tree keys he/she knows will never be used again. Thus, Rex can not collude with the left receivers. And the receivers who plan to join after \(T_b\) can not get the IK′; they can not collude with \(Rex\). However, \(Rex\) can collude with the receivers who join after the expulsion time and before \(T_b\).

Acknowledgment

This work was funded by the National Development and Reform Commission (NDRC) People's Republic of China.

Reference