安全多播中基于成员行为的 LKH 方法*

许 勇[1,2+], 陈 恺[1,2]

[1](东南大学 计算机科学与工程系,江苏 南京 210096)

[2](计算机网络和信息集成教育部重点实验室(东南大学),江苏 南京 210096)

An LKH Method Based on the Behavior of Group Members in Secure Multicast

XU Yong[1,2+], CHEN Kai[1,2]

[1](Department of Computer Science and Engineering, Southeast University, Nanjing 210096, China)

[2](Key Laboratory of Computer Networks and Information Integration (Southeast University), Ministry of Education, Nanjing 210096, China)

+ Corresponding author: Phn: +86-25-83792360, Fax: +86-25-83792757, E-mail: yxu@seu.edu.cn, http://www.seu.edu.cn

Received 2003-11-25; Accepted 2004-04-01

Xu Y, Chen K. An LKH method based on the behavior of group members in secure multicast. *Journal of Software*, 2005,16(4):601−608. DOI: 10.1360/jos160601

Abstract: LKH (logical key hierarchy) is a basic method in secure multicast group rekeying. LKH is efficient in real time group rekeying since it does not distinguish the different probability among the group members. However when members have diverse changing probability or different changing modes, the gap between LKH and the optimal algorithm will become bigger. If the probabilities of members have been known, LKH can be improved someway, but the changing probability of members can not be known exactly. Based on the basic knowledge of group members, in R-LKH (Refined-LKH), the active members and inactive members are partitioned and set on different locations in the logical key tree firstly. Then the concept "dirty path" is introduced in order to reduce the repeated rekeying overhead in the same path. All these can decrease the number of encryption in group manager and the network communication overhead. The simulation result indicate that R-LKH has a better improvement over LKH if the multicast group members' behavior could be distinguished "approximately".

Key words: logical key tree; rekey; dirty path; active member; inactive member

摘 要: LKH(logical key hierarchy)方法是安全多播实时密钥更新中常用的方法.LKH 对所有成员的行为没有进行区分,在具有相同成员变化概率的情况下,具有较高的效率.但当组成员具有不同的变化概率,或者成员行为模式不同时,LKH 的效率与最佳值的差距将会变大.在已知成员变化概率的情况下,可对 LKH 方法进行一定的改进,但要确切了解每个成员的变化概率,事实上是不可能的.R-LKH(Refined-LKH)方法无须准确了解组成员的变化概率,在已知基本成员变化信息的基础上,先将组成员分为活跃成员和非活跃成员两部分,分别将其安排在密钥树的不同位置,然后通过引入"脏路径"的概念,以尽可能地减少同一路径上密钥的反复更新,从而达到了减少

* Supported by the National Natural Science Foundation of China under Grant No.90104009 (国家自然科学基金)

作者简介: 许勇(1966－),男,安徽六安人,博士,副教授,主要研究领域为高性能计算机网络,网络安全;陈恺(1977－),男,硕士生,主要研究领域为网络安全.

© 中国科学院软件研究所 http://www.jos.org.cn

管理者加密次数,降低通信开销的目的.仿真实验结果表明,在对多播组成员的行为方式进行“大致”区分的情况下,R-LKH 方法比 LKH 有较大的优势.

关键词: 逻辑密钥树;密钥更新;脏路径;活跃成员;非活跃成员

中图法分类号: TP309 **文献标识码**: A

安全多播越来越受到人们的关注,为保证多播自身的优势,安全多播的实现通常采用多播组成员共享同一组密钥的方式实现[1−5].由于多播组的动态性,在实现安全多播时一般有两个限制条件,即前向安全性和后向安全性(新加入的成员不能访问以前的通信内容,离开成员不能访问后继通信内容).这就要求安全多播管理者在组成员发生改变时,对组密钥进行更新,因此多播密钥更新的效率是安全多播实现时需要考虑的一个重要问题.多播密钥更新的效率通常由下列因素决定:管理者存储量和计算量大小、密钥更新时通信开销以及成员方的存储量和计算量大小.这几个因素之间具有相互制约的关系,因此,目前很多文献中所实现的方法均是在这几个因素间进行一定的折衷,以达到满足某种需要的目的.

LKH 方法[1]采用密钥树方式对密钥进行管理.在这棵密钥树中,根结点对应组密钥,叶结点和组成员一一对应,其中的密钥仅为成员和管理者共享,其他结点对应的密钥均为辅助密钥.这样,在增加一定辅助密钥的基础上,对大小为 N 的多播组,可将组密钥更新的开销变为 $O(d\log_d N)$.OFT 方法[2]也是将多播组密钥安排成树型结构(二叉树),但基于单向函数计算,可将密钥更新开销从 $O(d\log_d N)$进一步降低到 $O(\log N)$.

在 LKH 和 OFT 方法中,多播组成员拥有的密钥为对应的叶结点到树根结点路径上的所有密钥,其中,根结点为组密钥.当多播组成员发生变化,需要进行密钥更新时,整条路径上的所有密钥均需要更新.以 LKH 为例,在进行密钥更新时,需要组管理者生成若干新密钥,并采用自底向上的方法逐次用相应的密钥加密,最后形成一个密钥更新报文,多播到整个多播组.因此,密钥更新报文的大小(与加密次数成正比例)和组管理者的计算开销便成为衡量算法优劣的关键.在分析 LKH 实现的基础上,本文提出了一种基于组成员行为的、改进的 LKH 方法——R-LKH(Refined-LKH).此方法在成员离开多播组时,只需更新组密钥,余下的密钥可以暂缓更新.这样,一方面减少了组管理者生成密钥的数量,另一方面也减少了组管理者的计算开销.本文第 1 节给出 R-LKH 的实现思想,并对其实现开销进行具体分析.第 2 节是 R-LKH 方法的仿真结果和分析.第 3 节给出 R-LKH 的实现方法.第 4 节对相关的研究现状进行了分析.最后是结论和未来的工作.

1 R-LKH 方法

R-LKH 方法基于这样的设想,即在不影响多播通信安全性的基础上,延缓辅助密钥的修改,以减少同一辅助密钥反复更新所带来的不必要开销.为此,我们在对多播组成员离开的情况下,只改动组密钥,以保证后向安全性;当有新的成员加入时,再进行加入点到根结点整条路径上密钥的更新.直观上看,这种操作将原先的两次全路径密钥更新变成了一次操作,从而减少了更新开销.

以一棵由 8 个结点的二叉树为例(如图 1 所示),在 LKH 方法中,如果成员 $U4$ 离开多播组,则密钥树中的密钥 K,$K14$,$K34$ 均需要修改,为此,管理者首先生成新的密钥 K',$K14'$,$K34'$,然后生成密钥更新报文($E_{K3}(K34')$,$E_{K34}(K14')$,$E_{K12}(K14')$,$E_{K14}(K')$,$E_{K58}(K')$)多播到整个多播组.而在 R-LKH 方法中,管理者只需产生新的组密钥 K',然后生成密钥更新报文($E_{K3}(K')$,$E_{K12}(K')$,$E_{K58}(K')$)多播到整个多播组.可以看出,在 R-LKH 方法中,组管理者的加密开销和更新报文大小大约减少为原来的一半.

事实上,R-LKH 的实现可以看作对密钥树的一种“剪枝”操作,例如,当成员 $U4$ 离开多播组时(图 1),管理者将 U4 在密钥树中通往根结点的一条路径“剪除”(这条路径中对应的密钥就是成员 $U4$ 拥有的所有密钥),此时,可将整棵密钥树看成若干子树形成的一个森林(子树的个数与树高相同).

但上例只考虑了一种简单情形,当连续有若干成员离开后,按照 R-LKH 方法,逻辑密钥树将被划分成众多的逻辑子树.若将当前的子树数量记为 i,则当再有成员离开时,子树的数量将增加 h(这里的 h 为子树高度.按上述算法,高为 h 的子树中有成员离开时,此树将变为 h 棵子树),所以,此时密钥更新报文将在前一次的基础上再增

© 中国科学院软件研究所 http://www.jos.org.cn

加 h 个,最坏情况下,h 的值只比原树高少 1.因此,随着离开成员数量的增加,这种算法的效率将逐渐降低.

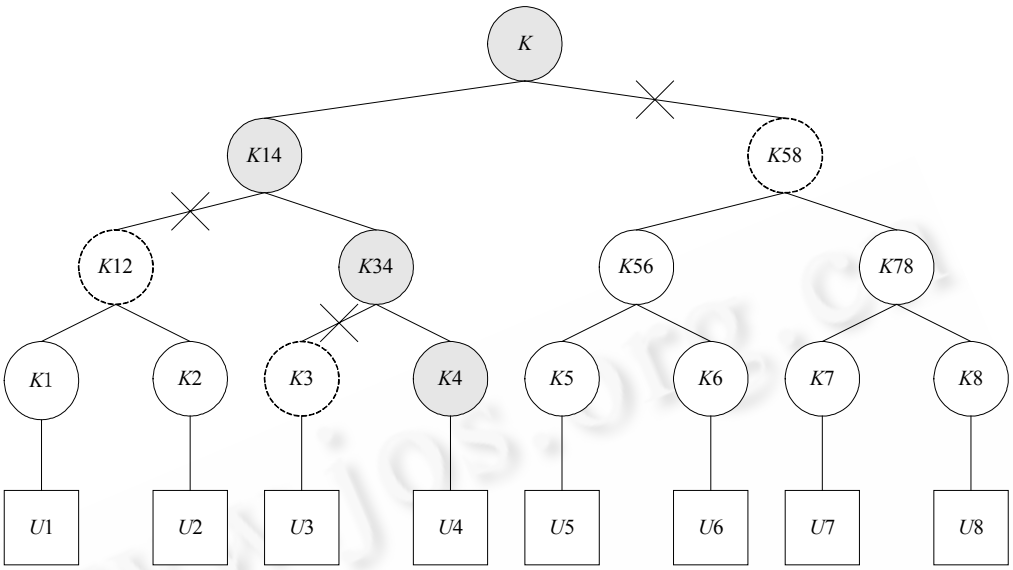

Fig 1 Group key rekeying when $U4$ leaves

图 1 成员 $U4$ 离开的组密钥更新

若考虑既有成员离开,又有成员加入的情形,当成员的加入和离开交替进行时,只要将新加入的成员插入到离开成员的位置,按照上述算法,就可以减少一次全路径的密钥更新.

上述两种情形都带有一定的特殊性.实际情况中,组成员的变化虽然交替进行,但在一段时间内,成员变化往往是随机的,这也就意味着可能有短暂而连续的成员离开或加入.因此,上述算法有一定的局限性.

为了避免上述局限,我们对上述算法进行了改进.首先引入如下定义:

定义 1. 对于密钥树中的从叶结点到根的一条路径,如果其上的辅助密钥在相应的叶结点离开后没有更新,则称此路径为"脏路径(dirty path)",简称 DP.若两条 DP 的交点为根结点,则称它们是无关的,否则称为相关的.

由上述定义可知,LKH 方法中 DP 的数量为 0,而在 R-LKH 方法中,DP 的数量至少为 1.在二叉树中,最多只有两条无关的 DP.一般情况下,一棵 d 叉树最多只有 d 条无关的 DP.对于树高为 h 的二叉密钥树,有以下结论:

结论 1. 在有两条无关 DP 的二叉密钥树中,组密钥更新时,密钥服务器所需的加密次数为 $2(h-1)$.

证明:对新的组密钥的加密只能使用子树的根结点所对应的密钥,而每条 DP 将密钥树分成 $h-1$ 棵子树. □

易见,如果在二叉树中有超过两条的 DP,则至少有两条 DP 是相关的.也就是说,这两条 DP 一定有一段共享路径,且这段路径的起点从两条 DP 的相交点直到根结点.若以相交点为新的根结点,则这棵子树又构成一棵具有两条无关 DP 的二叉树.

引入 DP 的主要目的是减少密钥树相同路径上密钥的重复更新.对于包含 DP 的密钥树,如果有新成员加入,可将其加入到 DP 对应的叶结点处,同时更新 DP 中的所有密钥,此时,密钥树中 DP 的数量减少一条.

结论 2. 对于有一条 DP 的二叉密钥树,如果有新成员加入,按上述方法进行密钥更新,管理者所需的加密次数为 $2h$.若有两条无关的 DP 存在,则加密次数为 $3h-1$.

证明:对于只有一条 DP 的密钥树,其密钥更新过程和 LKH 相似,为 $2h$.对于有两条无关 DP 的密钥树,对于插入点之外的另一棵子树而言,更新组密钥的加密次数为 $h-1$,因而总共需要的加密次数为 $3h-1$. □

从结论 2 和前面的分析可以看出,随着密钥树 DP 数量的增加,密钥更新的代价也会随之增加.因此,在 R-LKH 中,我们将 DP 数量选择为 1 或 2.

注意到新成员的加入可使 DP 数量减 1,而成员离开时使得 DP 数量增加 1.因此,当 DP 值为 2,又有成员离开时,需要进行一些特殊处理.由前面的分析可知,离开成员所产生的 DP 一定和原有的一条 DP 相关.假定这两

© 中国科学院软件研究所 http://www.jos.org.cn

条相关 DP 在结点 M 处相交,在进行组密钥更新的同时,也将 M 的一棵包含原先 DP 的子树中的部分路径进行更新,就可以确保 DP 的数量不变.

结论 3. 对于有 2 条 DP 的二叉密钥树,当有成员离开需要进行密钥更新时,管理者所需的加密次数最坏情况下为 $4h-9$;对于只有 1 条 DP 的情形,管理者所需的加密次数最坏情况下为 $3h-7$.

证明:最坏的情况出现在新产生的 DP 和已有的 DP 相交于根结点的子结点.此时,更新原先 DP 中的部分路径并将新的组密钥发送到相应子树结点所需的加密次数为 $2(h-2)-1$;将新的组密钥发送到与新 DP 相关的结点所需的加密次数为 $(h-2)-1$;另一 DP 相关结点所需的加密次数为 $h-1$.当只有一条 DP 时,最坏情况下,管理者所需的加密次数为 $2(h-2)-1+(h-2)-1+1=3h-7$. □

结论 3 表明,当只有成员离开时,最坏情况下,R-LKH 方法与 LKH 相比没有优势.将结论 3 推广到一般情况,对于有两条 DP 的二叉密钥树来说,假定两条 DP 交叉点的子树高度为 h_1,则它们的共享路径长度为 $h-h_1-1$,此时管理者进行密钥更新的加密次数为 $(2h_1-1)+(h_1-1)+(h-h_1-1-1)+(h-1)=2h+2h_1-5$ *.当 $h_1<2$ 时,R-LKH 需要的加密次数比 LKH 低.同样地,对于只有一条 DP 的二叉密钥树来说,成员离开时,R-LKH 需要的加密次数为 $(2h_1-1)+(h_1-1)+(h-h_1-1-1)+1=h+2h_1-3$.当 $h_1\le(h+2)/2$ 时,R-LKH 需要的加密次数比 LKH 要低.因此,在仅有成员离开的情况下,一条 DP 较两条 DP 有明显的优势.

上述分析给我们的另一个提示是:若当前离开成员所形成的 DP 和前一离开成员在密钥树的同一棵子树中,且相互位置较近,则两者 DP 的共享路径较长,此时进行密钥更新的代价较低.因此,若将离开成员限制在同一棵子树中,R-LKH 方法具有一定的优势.

对于成员加入,由结论 2,当只有一条 DP 时,其开销和 LKH 相同,但对于 R-LKH 方法来说,随着此成员的加入,这条 DP 也随之消除,因此减少了后续离开成员的处理开销.

综合上述两方面的情况可以看到,在构建密钥树时,若将组成员按某种方式,如活跃程度进行适当的分类,也就是将经常变动的成员放在同一棵子树,将不常变动的成员放在另一棵子树,采用 R-LKH 方法进行密钥更新时,所需的开销较小.实际的仿真结果也说明了这一点.

对于多播安全中的前向和后向安全性,R-LKH 同样也是满足的.

定理 1. 在处理多播组成员离开时,R-LKH 的密钥更新方式具有后向安全性.

证明:如图 1 所示,将离开成员对应的路径从密钥树中剪枝(使其和原密钥树分离),此时可以得到若干棵原密钥树的子树(分别以 $K3$,$K12$ 和 $K58$ 为根的子树).分别用这些树的根结点密钥加密新的组密钥,所形成的密钥更新报文中没有离开成员所拥有的密钥,因此,离开成员无法获取后续的通信内容.

定理 2. 在处理多播组成员加入时,R-LKH 的密钥更新方式具有前向安全性.

证明:对于成员的加入,按照新成员插入密钥树的具体位置分为如下 3 种情形:第 1,此前仅有一个成员离开,在离开成员处;第 2,没有成员离开时,对某个叶结点进行分解而得到新位置;第 3,有两个以上成员离开时,最后一个离开成员处.这几种情形和 LKH 的处理方法相似.因此,LKH 的正确性保证了定理 2 的正确性.

2 仿真结果及分析

我们考察了 200 次成员的随机加入和离开.考虑到离开成员的影响较大,在实际选择离开、加入成员数量时,分别设定从(190,10)到(100,100)10 个区间段,括号中前一数据代表离开成员数量,后者表示加入成员数量.多播组大小选择从 1024 到 32K.在实际计算时,为了更好地体现成员变化的随机性,在每一个区间段,我们均进行了 10 次计算,然后取其算术平均值作为最终的结果.

*当有两条 DP 存在时,式中 $2h_1-1$ 为交叉点的一个子树中所有结点密钥更新所需的加密次数(更新后子树中的一段 DP 消失);h_1-1 为其另一棵子树位于 DP 下的结点数量,因而也是为这些结点获取新的组密钥所需的加密次数;$h-h_1-1-1$ 和 $h-1$ 分别为共享路径和另一条 DP 下的结点数量,因此,同样是这些结点获取新的组密钥所需的加密次数.

© 中国科学院软件研究所 http://www.jos.org.cn

2.1 R-LKH方法的主要数据结构(以二叉树为例)

```
typedef struct {
    bool IsDirty;        // “dirty”标志:结点密钥是否需更新
    bool IsEmpty;        // 本结点是否已被删除
    int Father;          // 父结点编号
    int LChild;          // 左孩子结点编号
    int RChild;          // 右孩子结点编号
}
```

R-LKH 与 LKH 对密钥树的构造采用相同的方法.在实现中,我们采用了满二叉树表示法,即从根结点(编号为 1)开始,自上而下,逐层从左到右对所有结点编号.无论根结点、中间结点或者叶结点都采用相同的数据结构.根据二叉树的性质,对于编号为 i 的结点,其父结点为[i/2],左孩子为 2i,右孩子为 2i+1;对于根结点,其父结点编号设置为 0,对于叶结点,其子结点编号设置为 0.

2.2 主要算法描述

(1) 初始化:假设组播组的最大成员个数为 n,建立一棵 2n−1 个结点的满二叉树,对所有结点将 IsDirty 和 IsEmpty 标志置为 false,初始化父结点 Father 和左右孩子结点 LChind、RChild 编号.

(2) 结点离开(成员离开):对于离开的结点将其 IsDirty 和 IsEmpty 标志置为 true,并将此结点至根结点路径上的所有结点的 IsDirty 标志置为 true,若某个结点的左孩子和右孩子结点都为空,则将此结点的 IsEmpty 标志置为 true;若遇到某个结点的 IsDirty 标志已被置为 true,则说明有两条 DP 交汇于此,选择其中任意一条路径更新交汇点以下的结点,然后对其他非“dirty”结点密钥进行更新.

(3) 结点加入(成员加入):在二叉树中寻找 DP,若找到,将新结点添加到此路径上,并更新整条 DP 上的结点;如果没有找到,则随机找一个空结点加入,并更新新结点到根结点路径上的所有结点.

2.3 算法实现框架

(1) 初始化:

```
For 2n−1 个结点{
    将结点的 IsDirty 和 IsEmpty 标志置为 false;
    Father=[结点编号/2];
    If 叶结点        //成员的结点编号从 n 至 2n−1
        LChild=RChild=0;
    Else {
        LChild=结点编号×2;
        RChild=结点编号×2+1;}}
```

(2) 成员离开/加入处理

```
If 当前操作为成员离开 {
    将离开结点的 IsEmpty 标志置为 true;
    当前结点=离开结点;
    While 当前结点 !=根结点 && 当前结点的 IsDirty 标志为 false {
        将当前结点的 IsDirty 标志置为 true;
        If 当前结点的兄弟结点的 IsDirty 标志为 false
            更新以兄弟结点为根的子树上的脏结点密钥并发送到成员结点;
        当前结点=当前结点的父结点;
        If 当前结点的儿子结点都为空
```

© 中国科学院软件研究所 http://www.jos.org.cn

```
                    将当前结点的 IsEmpty 标志置为 true;}
                更新组密钥并发送到各个成员结点;}
        Else {                                          // 处理结点加入情况
                For n 个成员结点 {
                    If 结点为空 {
                        更新位置=当前结点位置;
                        If 父结点的 IsDirty 标志为 true        // 存在脏路径
                            Break;}}
                在当前结点位置加入新结点;
                将新结点至根结点上所有结点的 IsDirty 标记置为 true;
                更新 IsDirty 标记为 false 的结点密钥并发送到各个成员;}
```

2.4 各种情形下的仿真结果

图 2~图 13 是仿真结果,我们可以看出,当变动的组成员位于密钥树相对集中的位置时,R-LKH 方法具有明显的优势,特别是当离开和加入的成员数量接近时更是如此.因此,在构造密钥树时,如何将多播会话期间可能变动的成员安排在相对集中的子树中,是确保 R-LKH 算法实现的关键.

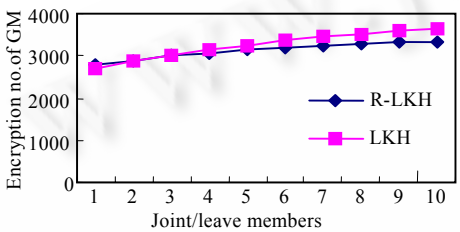

Fig.2 1024 members, range in 1/4 key tree

图 2 成员数为 1024,变化范围为 1/4 子树

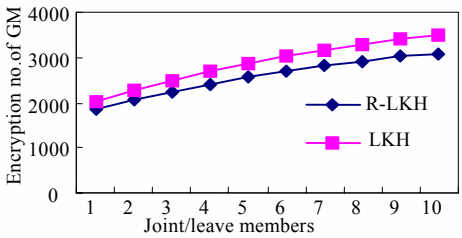

Fig.3 1024 members, range in 1/8 key tree

图 3 成员数为 1024,变化范围为 1/8 子树

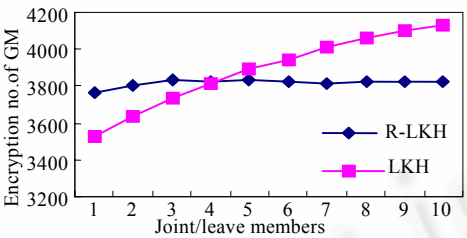

Fig.4 2048 members, range in 1/4 key tree

图 4 成员数为 2048,变化范围为 1/4 子树

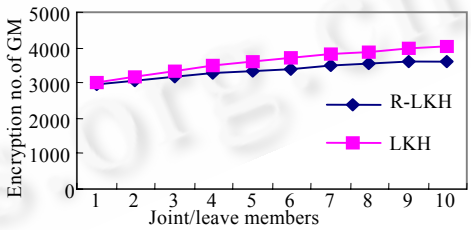

Fig.5 2048 members, range in 1/8 key tree

图 5 成员数为 2048,变化范围为 1/8 子树

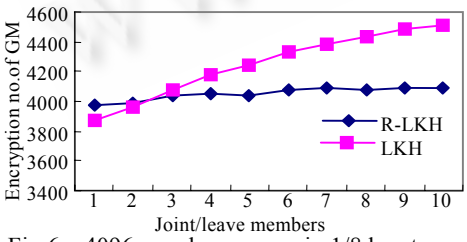

Fig.6 4096 members, range in 1/8 key tree

图 6 成员数为 4096,变化范围为 1/8 子树

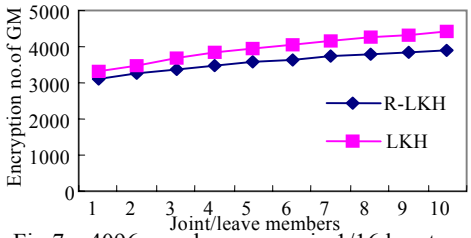

Fig.7 4096 members, range in 1/16 key tree

图 7 成员数为 4096,变化范围为 1/16 子树

© 中国科学院软件研究所 http://www.jos.org.cn

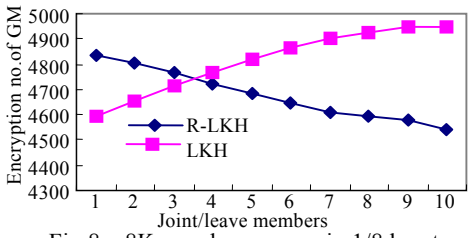

Fig.8 8K members, range in 1/8 key tree

图 8 成员数为 8K,变化范围为 1/8 子树

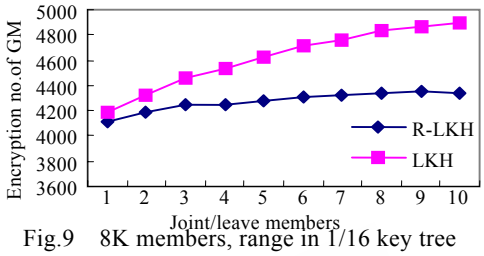

Fig.9 8K members, range in 1/16 key tree

图 9 成员数为 8K,变化范围为 1/16 子树

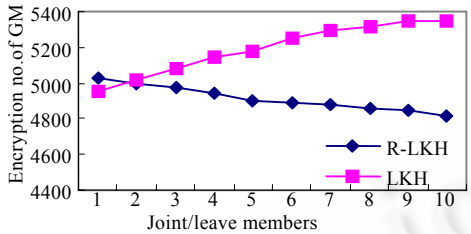

Fig.10 16K members, range in 1/16 key tree

图 10 成员数为 16K,变化范围为 1/16 子树

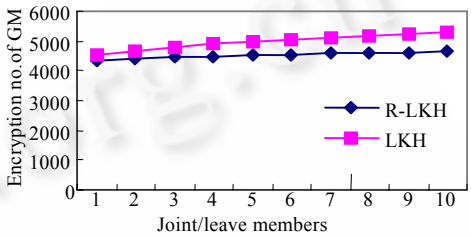

Fig.11 16K members, range in 1/32 key tree

图 11 成员数为 16K,变化范围为 1/32 子树

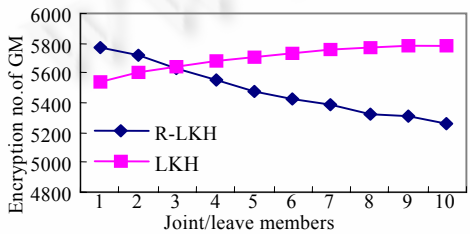

Fig.12 32K members, range in 1/16 key tree

图 12 成员数为 32K,变化范围为 1/16 子树

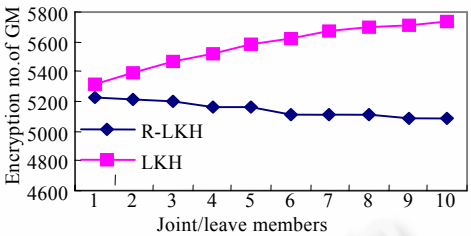

Fig.13 32K members, range in 1/32 key tree

图 13 成员数为 32K,变化范围为 1/32 子树

3 R-LKH 的实现

在多播会话开始之前,确切了解每个成员的离开概率,事实上是不可能的.但在多播组初始化时,要求加入成员给出可能的行为方式是可行的.这里的行为方式主要是指成员参与多播会话的频度和持续时间,以及是否为已注册成员或新注册成员等.通过对这些成员行为的统计和分析,可将组成员分为活跃的(active member)和非活跃的(inactive member)两种类型[6],其中的活跃成员是指停留时间较短或为一些临时成员,而非活跃成员可以是停留时间较长或注册成员.将活跃成员尽量放置在一棵最小的子树中,其余的位置可以放置非活跃成员.从上述仿真过程中可以看出,对于活跃成员在密钥树中的位置,实际的要求并非十分严格,如 8K 个成员的多播组中,只要将这些成员放置在一棵大小为 512 或 1024 的子树中就可以得到较好的结果;在大小为 32K 的多播组中,这样的子树大小可以为 1024 以上.只要在这个范围之内,R-LKH 在处理成员变化时密钥更新的开销均较 LKH 有较大的优势.因此,在实现上,R-LKH 不需要特别的计算,就可以决定各成员对应的密钥树位置,因而具有较大的灵活性和方便性.此外,对于 R-LKH 来说,在密钥更新过程中的很多情况下,往往只有组密钥需要更新,因此,管理者所需产生的密钥量将有所减少,同时管理者所进行的加密工作也只是对同一个实体来进行(即分别用不同的密钥对新的组密钥加密),因而具有较高的效率.

© 中国科学院软件研究所 http://www.jos.org.cn

4 相关研究现状

LKH+[4]和 LKH+2[5]分别对成员加入和离开进行了优化,其中 LKH+将成员的加入开销降低到 $O(1)$,而 LKH+2 在增加一定单向函数计算后将成员的离开开销降低到了 $O(\log N)$(与 OFT 类似).文献[6,7]在 LKH 方法的基础上,提出了一种利用伪随机函数计算进行密钥更新的方法,该方法可将管理者的密钥更新开销降低到 $O(1)$.文献[8]提出基于成员变化概率构造密钥树的方法,当成员加入多播组时,需要按照局部最优或全局最优方式计算插入代价,对成员的变动概率的获得,文献给出了一种基于统计的计算方法.文献[5−8]的共同特点是采用增加计算量的方式来改进密钥更新通信开销.

5 结 论

在对组密钥进行实时更新时,R-LKH 利用多播组成员的行为方式,在密钥更新开销上较 LKH 方法有较大的改进.与文献[8]相比,R-LKH 方法不要求计算每个成员准确的变化频率,只需“大致”了解成员行为的变化情况即可.因此,具体实现时所需考虑因素较少,简单易行.当然,对成员行为变化了解得越多,R-LKH 的优势就越明显.相信随着多播应用的不断发展,人们对参与多播会话成员的行为模式的了解也会不断深入.因此,本文今后的工作将分为两个方面,一是继续完善现有的算法,二是结合具体的多播应用,分析设计相应的安全多播模型,以对 R-LKH 算法加以进一步的改进.

致谢 感谢中国工程院院士顾冠群教授对本文的建议和指导.

References:

[1] Wong CK, Gouda M, Lam SS. Secure group communications using key graphs. IEEE/ACM Trans. on Networking, 2000, 8(1):16−30.

[2] Wallner D, Harder E, Agee R. Key management for multicast: Issues and architectures. RFC 2627, 1999.

[3] Sherman AT, Mcgrew DA. Key establishment in large dynamic groups using one-way function trees. IEEE Trans. on Software Engineering, 2003,29(5):444−458.

[4] Waldvogel M, Caronni G, Sun D, Weiler N, Plattner B. The versaKey framework: Versatile group key management. IEEE Journal on Selected Areas in Communications, 1999,17(9):1614−1631.

[5] Rafaeli S, Mathy L, Hutchison D. LKH+2: An improvement on the LKH+ algorithm for removal operations. Internet draft (work in progress), Internet Eng. Task Force, 2002. http://www.watersprings.org/pub/id/draft-rafaeli-lkh2-00.txt

[6] Pegueroles J, Bin W, Soriano M, Rico-Novella F. Group rekeying algorithm using pseudo-random functions and modular reduction. Grid and Cooperative Computing (GCC), Lecture Notes in Computer Science, 2004,3032:875−882.

[7] Pegueroles J, Rico-Novella F, Hernández-Serrano J, Soriano M. Improved LKH for batch rekeying in multicast groups. In: Proc. of the IEEE Int'l Conf. on Information Technology Research and Education (ITRE 2003). New Jersey: IEEE Press, 2003. 269−273.

[8] Selcuk AA, Sidhu D. Probabilistic optimization techniques for multicast key management. Computer Networks, 2002,40(2): 219−234.

© 中国科学院软件研究所 http://www.jos.org.cn