

# 安全多播中密钥更新机制的性能优化

李保红, 侯义斌, 赵银亮

(西安交通大学电子与信息工程学院, 710049, 西安)

**摘要:** 针对安全多播中密钥更新的可扩展性问题, 提出了一种改进的逻辑密钥分层机制. 在更新密钥树时, 由密钥服务器产生随机数, 而多播组成员使用单向散列函数可以直接计算出变动路径中的全部或部分密钥, 减少了更新密钥的计算量和在多播信道中的通信量, 因此使密钥服务器的平均代价减少约 1/3. 在此基础上提出了适合于这种改进机制的批处理更新算法, 可以对多次成员变动仅进行一次更新操作. 实验分析表明, 与原机制的批处理更新算法相比, 该算法又可使密钥服务器的代价至少减少 1/3. 因此, 采用这种改进机制的批处理更新算法可以进一步提高计算和通信性能.

**关键词:** 安全多播; 会话密钥更新; 密钥树; 单向散列函数

**中图分类号:** TP393 **文献标识码:** A **文章编号:** 0253-987X(2004)10-1053-04

## Performance Optimization for Rekeying Mechanism in Secure Multicast

Li Baohong, Hou Yibin, Zhao Yinliang

(School of Electronics and Information Engineering, Xi'an Jiaotong University, Xi'an 710049, China)

**Abstract:** Aiming at the scalability of rekeying in secure multicast, an improved logic key hierarchy mechanism is proposed. When a key tree is updated, the key server produces random numbers, and all or the part of new keys in update paths are directly calculated by the multicast group members using one-way hash function. Therefore, the burdens of the computation and communication for updating keys through multicast channels are decreased. It is estimated that the key server's average cost can be reduced by about 1/3. A batch update algorithm is also presented in the paper based on the improved scheme, in which changing of members several times can be carried out only by update operation once. It is shown by experiments that the key server's cost of this algorithm can be reduced again at least 1/3. So the performance is further improved when this batch update algorithm is applied to the improved scheme.

**Key words:** secure multicast; session key update; key tree; one-way hash function

在安全多播中, 当多播组中成员发生变动(加入或退出)时, 密钥服务器需要对会话密钥(SK)进行更新, 以保证前向、后向机密性. SK的更新机制不仅要求保证安全性, 而且应具有可扩展性. 逻辑密钥分层机制(LKH)<sup>[1,2]</sup>因为具有较好的可扩展性而成为主流的更新机制. 为了进一步提高该机制的性能, 文献[3]建议采用批处理方式更新密钥, 即定期地对多次成员变动进行一次更新操作. 批处理方式不仅使密钥服务器的计算量和带宽消耗(代价)显著下降,

也在一定程度上缓解了更新密钥因失同步而带来的安全问题<sup>[4]</sup>.

本文对LKH机制进行了改进, 称为ILKH, 由此得到了两个成果: 使用单向散列函数直接计算变动路径中的密钥, 使密钥服务器的平均代价减少约1/3, 并提出了实现ILKH批处理更新的算法.

### 1 逻辑密钥分层机制

在LKH机制中, 每个多播组仅有一个密钥服

收稿日期: 2003-12-22. 作者简介: 李保红(1968~), 男, 讲师. 基金项目: 国家自然科学基金资助项目(60173066); 教育部“教育振兴行动计划”西安交通大学培植项目(PZ128).

务器,以对组内成员使用的密钥进行管理. 密钥服务器采用  $d$  叉树的数据结构存放这些密钥,该数据结构称为密钥树. 图 1 是一个有 9 个成员的多播组的密钥树,树高度  $h=3$ ,树根为第 0 层.

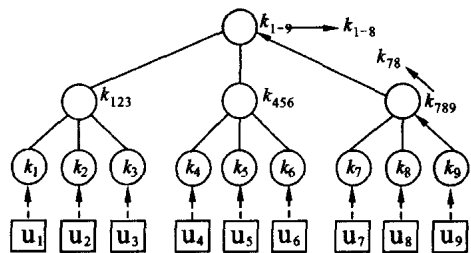


图 1  $d=3, N=9$  和  $h=3$  的逻辑密钥层次

在多播组中,每个成员持有一个只与密钥服务器共享的私有密钥,即图中的  $k_1 \sim k_9$ . 这些密钥作为密钥树的叶子结点,每个叶子对应一个成员,因此称其为  $u$  结点. 树根  $k_{1-9}$  是多播组当前使用的 SK. 密钥树中的中间结点  $k_{123}, k_{456}$  和  $k_{789}$  等是用于密钥更新操作的辅助密钥,它们与树根统称为  $k$  结点. 多播组内每个成员持有从对应的  $u$  结点到达树根时的所经路径的所有密钥.

当成员发生变动时,LKH 利用辅助密钥可以减少密钥服务器发布 SK 所需要的代价. 比如,当成员  $u_9$  退出多播组时,密钥服务器为从  $k_9$  到达树根时的所经路径的所有结点产生的新密钥  $k_{78}, k_{1-8}$ , 其中的  $k_{1-8}$  是新产生的 SK, 然后密钥服务器将更新的密钥发布给组内现有的成员. 若采用面向多播组<sup>[1]</sup>的机制,密钥服务器发布的多播消息为

$$\{k_{1-8}\}_{k_{123}}, \{k_{1-8}\}_{k_{456}}, \{k_{1-8}\}_{k_{78}}, \{k_{78}\}_{k_7}, \{k_{78}\}_{k_8}$$

其中,  $\{k_a\}_{k_b}$  表示用密钥  $k_b$  对  $k_a$  进行加密. 成员  $u_1 \sim u_6$  分别拥有  $k_{123}$  和  $k_{456}$ , 因而通过解密可以获得新的 SK. 成员  $u_7$  和  $u_8$  通过私有密钥可以得到新的辅助密钥  $k_{78}$ , 进而也可获得新的 SK. 成员  $u_9$  无法获得辅助密钥  $k_{78}$ , 因而也无法获得新的 SK.

设多播组成员数为  $N$  (不失一般性,假定  $N$  为  $d$  的幂), 密钥树高度  $h = \log_d N + 1$ . 若以加密的密钥数目来统计密钥服务器的代价<sup>[1]</sup>, 则 LKH 机制在成员加入和退出时, 密钥服务器的代价分别为  $2(h-1)$  和  $d(h-1)$ . 当加入和退出具有同等概率时, 则每次更新的平均代价为  $(d+2)(h-1)/2$ . 容易证明  $d=4$  时可获得最小代价.

## 2 改进的逻辑密钥分层机制

### 2.1 加入操作

以在图 2 的多播组中加入新成员  $u_k$  为例, 密钥服务器完成对新成员的身份认证后选择私有密钥  $k_k$  及一个随机数  $r$ , 通过安全信道颁发给  $u_k$ , 并将  $k_k$  插入到密钥树中. 插入位置应选择层次最小的结点, 以尽可能使密钥树保持平衡, 文献[5]讨论了保持平衡的详细算法. 在本例中,  $k_k$  作为结点  $k_{ij}$  的子节点插入. 私有密钥插入后, 密钥服务器需要更新从插入位置到树根的变动路径中的所有密钥  $k_{a-j}, k_{e-j}$  和  $k_{ij}$ . 因此, 密钥服务器根据随机数  $r$  计算  $k_{ijk} = f(r)$ ,  $k_{e-k} = f(f(r))$ ,  $k_{a-k} = f(f(f(r)))$ , 其中的  $f$  为单向散列函数, 如 MD5 等. 这些值依次作为路径中  $k$  结点和树根的值. 最后, 密钥服务器发布面向多播组的多播消息

$$\{k_{a-k}\}_{k_{a-j}}, \{k_{e-k}\}_{k_{e-j}}, \{k_{ijk}\}_{k_{ij}}$$

其中,  $k_{a-j}$  为原 SK, 因而子树 1 和子树 2 中的成员经解密后可以得到  $k_{a-k}$ , 即新的 SK, 而子树 3 和子树 4 中的成员均拥有原密钥  $k_{e-j}$ , 解密后可得到  $k_{e-k}$ . 通过计算  $f(k_{e-k})$  可以获得  $k_{a-k}$ . 由于这些成员拥有原 SK, 通过解密也可以得到  $k_{a-k}$ , 但解密操作比散列函数的计算量略大一些. 成员  $u_i$  和  $u_j$  获得  $k_{ijk}$  后计算  $f(k_{ijk}), f(f(k_{ijk}))$  可以获得  $k_{e-k}$  及  $k_{a-k}$ . 新加入成员  $u_k$  无需接收多播消息, 根据随机数  $r$  就可以计算出所需密钥.

随机数  $r$  的颁发可以结合认证协议进行, 比如密钥服务器在对加入成员认证时产生的挑战信息为  $\{r_1 \text{ 时间戳}\}_{k_{up}}$ , 成员的应答信息为  $\{r_1 \ r_2 \ \text{时间戳}\}_{k_{sp}}$ , 其中的  $k_{up}$  和  $k_{sp}$  分别为成员和服务器的公钥. 通过上述协议, 既完成认证, 也使双方得到随机数  $r = r_1' \ r_2$ , 无需增加额外通信量.

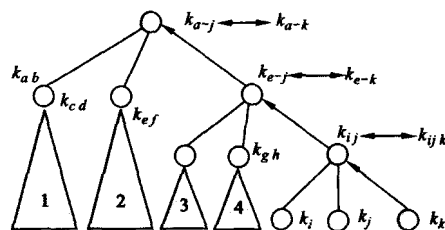


图 2 LKH 机制中成员的加入和退出

### 2.2 退出操作

以图 2 中  $u_k$  退出多播组为例, 密钥服务器删除  $k_k$ , 必要时需对密钥树进行剪枝操作. 由于  $u_k$  退出, 为了保证前向机密性, 密钥服务器需要更新变动路径中的密钥  $k_{ijk}, k_{e-k}$  和  $k_{a-k}$ . 密钥服务器选择另一

随机数  $r$ , 并计算  $k_{ij} = f(r)$ 、 $k_{e-j} = f(f(r))$  和  $k_{a-j} = f(f(f(r)))$ , 然后发布多播消息

$$\{k_{a-j}\}_{k_{a-b}}, \{k_{a-j}\}_{k_{c-d}}, \{k_{e-j}\}_{k_{e-f}}, \{k_{e-j}\}_{k_{g-h}}, \{k_{ij}\}_{k_i}, \{k_{ij}\}_{k_j}$$

与加入方法相似, 子树 1 ~ 子树 4 中的成员及  $u_i$  和  $u_j$  均可得到更新的辅助密钥和 SK

**定理 1** LKH 机制在成员插入和退出时, 密钥服务器的代价分别为  $h - 1$  和  $(d - 1)(h - 1)$ .

**证明** 由上述算法可知, 成员加入时只需将变动路径中的  $h - 1$  个新的密钥用相应原密钥加密, 成员退出时则需将变动路径中的  $h - 1$  个新的密钥用  $d - 1$  个子结点的密钥加密.

当加入和退出等概率时, 则每次更新的平均代价为  $d(h - 1)/2$ , 显然  $d = 3$  时获得最小值. 此时, LKH 与 LKH 的平均代价之比为  $3\log_3 N / 6\log_4 N \approx 0.163$ , 所以得到下述推论 1.

**推论 1** LKH 机制中密钥服务器的平均代价比 LK 约减少  $1/3$ .

**定理 2** LKH 与 LKH 具有相同的安全性.

**证明** 在 LKH 中, 新加入成员拥有的辅助密钥是新生成密钥, 由此无法推导出其他成员以前持有的辅助密钥和 SK, 因而保证了后向机密性. 当成员退出时, 密钥树中该成员拥有的密钥均被更新, 因而保证了前向机密性. 在 LKH 中, 上述情况仍能够得到以证明, 比如在图 2 中, 当  $u_k$  加入时, 同样只获得了新生成的辅助密钥和 SK. 另一方面, 子树 1 和子树 2 中的成员仅获得了新的 SK, 即  $k_{a-k} = f(f(f(r)))$ . 由于单向散列函数的单向性质, 这些成员无法得知  $k_{e-k}$ 、 $k_{ijk}$ , 因而当其中某成员退出时, 只需更新变动路径上的密钥就可以保证退出成员无法获得以后使用的任何辅助密钥和 SK. 出于同样的原因, LKH 与 LKH 抵御共谋攻击的能力也相同.

实际上, LKH 使用单向散列函数与使用一次性口令(S/KEY)、证书废弃系统(CRS)相似, 并且计算密钥时单向散列函数的最大迭代次数  $(h - 1)$  远小于密钥长度, 因而是安全的.

### 3 LKH 的批处理更新算法

在 LKH 中, 每当成员变动时就需要更新变动路径中的所有密钥, 当有多个成员变动时, 则需要更新多条路径中的密钥. 这些路径必然存在一个或多个公共结点, 如果密钥服务器在多个成员变动后才进行一次更新操作, 则可以有效减少代价, 这种方式称为批处理更新. 为了便于描述算法, 假定加入和退

出的成员数相等, 而且这种情况下的密钥树形态不变, 则只需用加入成员替换退出成员即可. 如图 3 中的新成员  $u_2$ 、 $u_3$ 、 $u_7$  和  $u_8$  替换退出成员, 形成了 4 条变动路径. 对加入和退出成员数目不等的情况, 也可类似地进行.

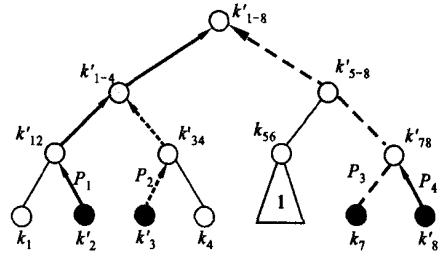


图 3 LKH 机制的批处理更新算法

LKH 批处理更新算法从叶子结点开始逐层向上进行, 对于每一层, 从左到右依次对各变动路径的同层结点进行处理, 处理后的结点再进行标记, 以免重复操作. 对于任一变动路径  $P_i$ , 算法如下.

(1)  $x_i$   $P_i$  中发生变动的叶子结点和  $y_i$   $x_i$  的父结点(表示赋值).

若  $y_i$  已标记, 则算法结束, 否则标记  $y_i$  为已更新结点. 选择随机数  $r_i$  并计算  $R_i = f(r_i)$ , 更新  $y_i$  的密钥  $k_{y_i} = R_i$ .

对于  $y_i$  所有的子结点  $z$   $x_i$ , 做加密操作  $\{k_{y_i}\}_{k_z}$ ,  $k_z$  为  $z$  的密钥.

(2)  $x_i$   $y_i$  结点,  $y_i$   $x_i$  的父结点.

若  $x_i$  为树根或  $y_i$  已被标记则结束, 否则标记  $y_i$  为已更新结点, 计算并赋值  $R_i = f(R_i)$ , 然后更新  $y_i$  的密钥  $k_{y_i} = R_i$ .

对于  $y_i$  所有的子结点  $z$   $x_i$ , 做加密操作  $\{k_{y_i}\}_{k_z}$ .

(3) 重复(2), 直到结束.

各路径更新后, 密钥服务器将算法产生的加密密钥组成一个多播消息发送给多播组. 例如, 在图 3 中最后得到的多播消息为

$$\{k_{12}\}_{k_1}, \{k_{1-4}\}_{k_{34}}, \{k_{1-8}\}_{k_{5-8}}, \{k_{34}\}_{k_4}, \{k_{78}\}_{k_8}, \{k_{5-8}\}_{k_{56}}$$

在路径  $P_1$  中, 成员  $u_2$  在加入时已得到随机数  $r_1$ , 而  $u_1$  由多播消息也可得到  $k_{12}$ , 两成员通过计算可得到  $P_1$  中全部密钥. 在路径  $P_3$  中, 成员  $u_7$  由成员加入时得到的随机数  $r_3$  可计算出  $k_{78}$  和  $k_{5-8}$ , 而算法在处理  $P_4$  时, 由于结点  $k_{78}$  已标记, 故立即结束,  $u_8$  需从多播消息中得到  $k_{78}$ , 经过计算得  $k_{5-8}$ , 进而两成员从多播消息中得到  $k_{1-8}$ .

**定理 3** 设变动的  $k$  结点(图 3 中的灰色结点)数为

$S$ ,则在 LKH 的批处理更新算法中,密钥服务器的代价为  $(d-1)S$ .

证明 由算法可知,对于每个  $k$  结点的密钥,密钥服务器使用  $d-1$  个子结点的密钥进行加密操作,因而得证.

#### 4 LKH 批处理更新算法性能分析

在分析 LKH 的批处理更新算法性能时假定:多播组中各成员以相同概率加入和退出,加入和退出成员数相等.采用文献[3]的方法可以估算密钥服务器的代价,即

$$C = (d-1) \sum_{L=h-2}^0 d^L \left( 1 - \frac{C_{N-m}^n}{C_N^n} \right) \quad (1)$$

式中: $N$  为多播组成员数; $m$  为变动成员数目; $n = d^{h-L-1}$  是以位于  $L$  ( $0 \leq L \leq h-2$ ) 层的  $k$  结点为根的子树所包含的叶子结点数.

由于式(1)不易分析,本文通过模拟程序分析批处理算法的性能.该程序随机选择变动成员并统计变动的  $k$  结点,从而得到密钥服务器的代价.

图4在  $N=7500$  时比较了  $d=2$ 、 $d=3$  和  $d=4$  的 LKH 批处理更新情况.对于  $d$  的不同取值,密钥服务器的代价均随着变动成员数目  $m$  的增加而增加,在  $m=N$  时达到最大值.该最大值相当于密钥服务器通过单播分别向每个成员颁发 SK.在  $d$  的不同取值中, $d=2$  时获得最优性能.改变  $N$ ,该结论仍然成立.

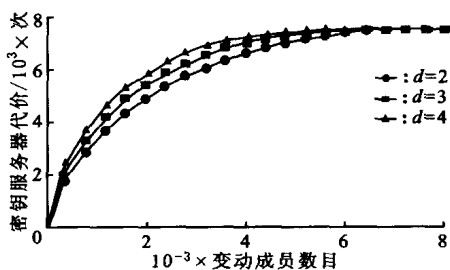


图4  $d=2$ 、 $d=3$  和  $d=4$  时 LKH 批处理更新的性能

图5是在  $N=7500$  时对 LKH、LKH 实时及批处理方式的比较. $m$  的变化范围为  $0 \sim 2000$ ,这是因为批处理方式以延迟成员加入和退出请求为代价,因此  $m$  的取值不宜过大.为了得到最优性能,LKH 在实时及批处理方式中取  $d=4$ <sup>[4]</sup>,而 LKH 在实时及批处理方式中取  $d=2$ 、 $d=3$ .

由图5可知,LKH 与 LKH 在两种实时方式中,密钥服务器代价的比值约为 0.163,与推论 1 相符.改

变  $N$  值,该比值在 0.162 ~ 0.165 之间变动,这是因为密钥树中叶子结点在最低两层的分布变化所致.在  $m=N/4$  处,LKH 批处理方式的代价达到最大值,而在此之前,LKH 与 LKH 的两种批处理方式的密钥服务器代价的比值小于 0.165.对于不同的  $N$ ,该结论仍然成立.因此,LKH 批处理方式使服务器代价至少减少 1/3.

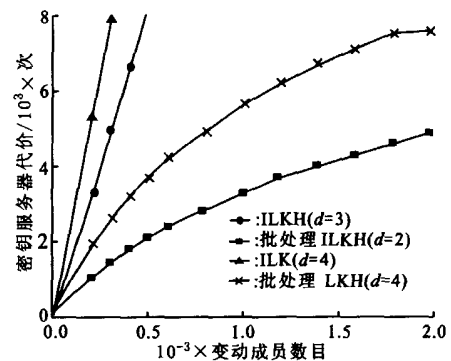


图5 LKH 和 LKH 的性能比较

#### 5 总结

实现会话密钥更新的可扩展性是安全多播中的一个关键问题.本文使用单向散列函数对 LKH 机制进行了改进,改进的机制使密钥服务器的计算量和通信量约减少 1/3.在此基础上实现的批处理更新算法也明显优于 LKH 批处理方式.

#### 参考文献:

- [1] Wong C, Gouda M, Lam S. Secure group communication using key graphs [A]. ACM's Special Interest Group on Data Communication, Vancouver, Canada, 1998.
- [2] Wallner D, Harder E, Agee R. Key management for multicast: issues and architecture [EB/OL]. <http://www.faqs.org/rfcs/rfc2627.html>, 2003-08-22.
- [3] Li S X, Yang R Y, Gouda M G, et al. Batch rekeying for secure group communications [A]. 10th International World Wide Web Conference, Hong Kong, 2001.
- [4] Tanaka S, Sato F. A key distribution and rekeying framework with totally ordered multicast protocols [A]. 15th International Conference on Information Networking, Beppu City, Japan, 2001.
- [5] Moyer M J, Rao J R, Rohatgi P. Maintaining balanced key trees for secure multicast [EB/OL]. <http://www.secure2multicast.org/draft2irtf2smug2key2tree2balance200.txt>, 2003-09-05.

(编辑 苗凌)