可扩展的 P2PSIP 会议密钥管理协议 *

施苑英
(西安邮电学院 通信与信息工程学院,西安 710121)

摘 要: 针对 P2PSIP 多媒体会议低延时、高扩展性需求,提出一种分级的会议密钥管理协议 HOAKA。根据处理能力将会议节点分成两级,由少量高性能节点负责会议管理,所有成员通过单向累加器算法协商会议密钥。研究结果表明,HOAKA 不仅具有较高的安全性,而且计算、存储开销低,密钥更新时延小,具有良好的可扩展性。

关键词: 对等会话初始化协议; 会议密钥; 单向累加器; 可扩展性
中图分类号: TP393.08 **文献标志码**: A **文章编号**: 1001-3695(2012)03-1063-03
doi:10.3969/j.issn.1001-3695.2012.03.072

Scalable P2PSIP conference key management scheme

SHI Yuan-ying
(*School of Telecommunications & Information Engineering, Xi'an University of Posts & Telecommunications, Xi'an* 710121, *China*)

Abstract: In order to meet the requirements of low latency and high scalability for P2PSIP conference, this paper proposed a hierarchical conference key management protocol-HOAKA. In HOAKA, conference nodes were divided into two classes according to their capabilities, a small amount of high-performance nodes managed the conference, and all the conferees agreed on a conference key by using one-way accumulator. The results show that HOAKA has not only high security, but also low computing and storage overhead as well as small key update delay. It has good scalability.

Key words: P2PSIP; conference key; one-way accumulator; scalability

P2PSIP[1,2]将分布式 P2P 技术融入传统集中式的 SIP[3]体系中,利用 P2P 重叠网为 SIP 实体提供资源定位和消息传输服务,提高了网络的可用性、可扩展性和灵活性。利用 P2PSIP 构建 Internet 会议通信系统,不必预先部署会议服务器,所有与会节点共同完成会议的组织和管理,从而避免了服务器的单点故障和性能瓶颈问题,同时也降低了系统的成本。

P2PSIP 通信面临着严重的安全威胁[4],因此会议中需要建立安全密钥,对通信数据进行加/解密。目前会议密钥的管理方式分为集中式、分布式和分层分组式[5]。集中式不适合 P2PSIP 框架;分布式要求所有参会节点具有相同的权限和处理能力,仅适用于小型会议;分层分组式将会议分成若干组,密钥更新操作仅限于组内,降低了密钥更新代价,具备良好的可扩展性,缺点是无统一的会议密钥,组间传送数据时需要多次进行加/解密,从而引入较大的延迟,不适合 VoIP、视频会议等实时应用。

为了满足大型 Internet 多媒体会议的需求,本文提出一种可扩展的 P2PSIP 会议密钥管理协议 HOAKA(hierarchical one-way accumulator based key agreement)。该方案采用分组思想,将会议划分成由少数高性能节点控制的组,同时所有参会成员协商得到一个共同的会议密钥,使数据传送时不必再进行多次加/解密。实验表明,HOAKA 具有良好的可扩展性和较小的密钥更新开销。

1 协议方案

1.1 系统架构

HOAKA 以文献[6]的会议模型为基础,将会议节点分为两类,即控制节点 CP(conference provider)和普通节点。CP 从带宽高、内存大、CPU 处理能力强、在线时间长的节点中选取,可以提供会议服务器的功能。PDA、智能手机等资源受限的设备只能作为普通节点。HOAKA 将会议分成若干组(图 1),每组由一个 CP 控制,CP 负责组成员管理、媒体流混合、会议密钥协商以及与其他组之间的通信。为了实现会议的动态扩展,HOAKA 允许 CP 按需增加,其中第一个加入会议的 CP 称为 PCP(primary CP),由会议发起者指定。随着新成员的加入,PCP 可以在会议中引入新的 CP,后续加入的 CP 均称为 SCP(secondary CP)。为了避免某个组成员过多导致 SCP 负荷过重,HOAKA 允许 SCP 继续引入新的 SCP 作为子节点,新用户将转移到子节点处理。最终,所有 CP 形成一个树型结构,PCP 是树根。

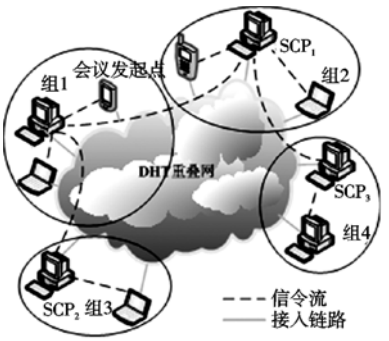

图1 HOAKA系统架构

1.2 会议密钥管理方案

HOAKA 采用单向累加器算法[7]实现会议密钥的协商。

收稿日期: 2011-06-05; **修回日期**: 2011-07-28 **基金项目**: 西安邮电学院中青年科研基金资助项目(ZL2010-13)
作者简介: 施苑英(1976-),女,陕西西安人,讲师,硕士,主要研究方向为下一代网络、P2P 技术(shi_yuanying@163.com).

单向累加器是一类具有准交换性[7]的特殊单向哈希函数,可以将一个集合中的所有元素以任意顺序累加成唯一值,并为每个元素提供一个证人信息,证明其确实参与了累加运算。在 HOAKA 中,每个会议成员必须贡献一个秘密份额,会议密钥是所有成员秘密份额的累加值。

HOAKA 约定使用如下符号:G_i 为会议的第 i 个分组;u_{ij} 为 G_i 的第 j 个成员;y_{ij} 为 u_{ij} 的秘密份额; ω_{ij} 为 u_{ij} 的证人信息;Y_i 为 G_i 的秘密份额;BY_i 为 G_i 的分支份额;$B\omega_i$ 为 G_i 的分支证人信息;$K_{i,j}$ 为节点 i 与 j 的共享对称密钥;CK 为会议密钥。

1.2.1 密钥建立

由 PCP 负责选取建立会议密钥的算法参数:先产生两个足够大的强质数 p、q,计算 $N=pq$ 和 $\Phi(N)=(p-1)(q-1)$;然后选择一个比较大的底数 x,x 与 N 互质。在会议初始化阶段,PCP 根据会议发起者提供的初始会议成员名单,选择 k ($k\geqslant 0$)个 SCP,将会议划分为 $k+1$ 组。k 值取决于初始会议规模。PCP 必须对每个 SCP 进行身份认证,建议采用文献[8,9]的算法,以便认证的同时建立起 PCP 和 SCP_i($1\leqslant i\leqslant k$)之间的共享密钥 K_{PCP,SCP_i}。认证通过后,PCP 将 x、N 和 $\Phi(N)$ 用 K_{PCP,SCP_i} 加密后安全发送给 SCP_i,并提供由 SCP_i 管理的组成员名单。此后,各 CP 邀请自己负责的组成员加入会议,每个新成员也必须进行身份认证。所有组成员都加入后, CP 在组内广播 N,然后开始协商会议密钥。

组内密钥协商:假设 G_i 有 l 个成员。u_{ij}($1\leqslant j\leqslant l$)产生随机质数 y_{ij}($y_{ij}\leqslant N$)作为自己的秘密份额,并安全发送给 CP_i。CP_i 获得所有成员的份额后,计算组份额 Y_i,$Y_i=y_{i1}y_{i2}\cdots y_{il} \bmod \Phi(N)$,是 G_i 所有成员份额之积。每个 SCP_i 还必须计算组分支份额 BY_i,它是以 SCP_i 为根的子树上所有成员份额之积。由于会议建立时,所有 SCP_i 都是叶子节点,因此 $BY_i=Y_i$。

组间密钥协商:SCP_i($1\leqslant i\leqslant k$)将 BY_i 安全发送给 PCP,PCP 据此计算 BY_{PCP} 和 SCP_i 的组分支证人信息 $B\omega_i$。$BY_{PCP}=Y_{PCP}BY_1BY_2\cdots BY_k \bmod \Phi(N)$,是所有会议成员的份额之积。$B\omega_i=x^{Y_{PCP}BY_1BY_2\cdots BY_{i-1}BY_{i+1}\cdots BY_k} \bmod N$,是不在以 SCP_i 为根的子树上的所有成员份额的累加值($B\omega_{PCP}=x$)。PCP 将 $B\omega_i$ 发送给 SCP_i,SCP_i 根据 $y'_{ij}=y_{ij}^{-1} \bmod \Phi(N)$ 和 $\omega_{ij}=B\omega_i^{BY_iy'_{ij}} \bmod N$ 计算 ω_{ij},ω_{ij} 是除 u_{ij} 外所有会议成员份额的累加值。

会议密钥产生:CP 和 u_{ij} 分别利用式(1)(2)计算 CK。

$$CK=B\omega_i^{BY_i} \bmod N \tag{1}$$

$$CK=\omega_{ij}^{y_{ij}} \bmod N \tag{2}$$

1.2.2 密钥更新

成员加入:假设新成员 u_{new} 加入组 G_i。CP_i 在 u_{new} 身份认证通过后,开始更新会议密钥。更新过程通过 Update 消息[3]实现,如图 2 所示。其步骤如下:

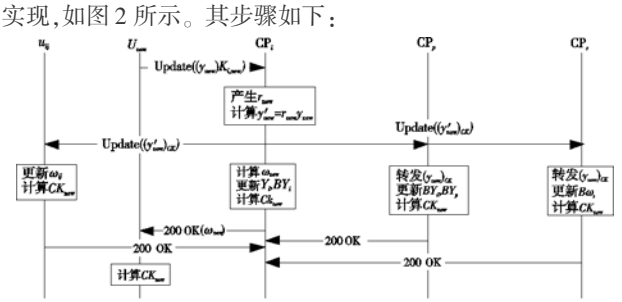

图2 新成员加入时密钥更新流程

a)u_{new} 产生 y_{new},并安全地发送给 CP_i(用与 CP_i 的共享密钥加密 y_{new},即图 2 中的$(y_{new})_{K_{i,new}}$)。CP_i 产生随机质数 r_{new}($r_{new}\leqslant N$),并计算 $y'_{new}=r_{new}y_{new} \bmod \Phi(N)$,然后用当前会议密钥 CK 加密 y'_{new}(图2 中的$(y'_{new})_{CK}$),并将加密值发送给组内所有成员以及 CP_i 的相邻节点。最后,CP_i 计算 ω_{new} 并通过 200 响应发送给 u_{new},同时更新 Y_i 和 BY_i。其中,$\omega_{new}=B\omega_i^{r_{new}BY_i} \bmod N$,$Y'_i=Y_iy'_{new} \bmod\Phi(N)$,$BY'_i=BY_iy'_{new} \bmod \Phi(N)$。

b)相邻节点收到 y'_{new},首先向所有组成员和相邻 CP(除 CP_i)转发。如果本节点是 CP_i 的父节点 CP_p,则更新 BY_i 和 BY_p,$BY'_i=BY_iy'_{new} \bmod \Phi(N)$,$BY'_p=BY_py'_{new} \bmod \Phi(N)$;如果本节点是 CP_i 的子节点 CP_s,则更新 $B\omega_s$,$B\omega'_s=B\omega_s^{y'_{new}} \bmod N$。这一步将重复执行,直到所有 CP 都更新了相关参数。

c)u_{ij} 收到 y'_{new} 后,更新 ω_{ij},$\omega'_{ij}=\omega_{ij}^{y'_{new}} \bmod N$。

d)所有 CP 和 u_{ij} 利用式(1)和(2)计算新的会议密钥。

成员离开:CP_i 发现 u_{ij} 离开后,首先产生一个随机质数 r($r\leqslant N$),计算 $y=ry_{ij}^{-1} \bmod \Phi(N)$;然后将 y 安全发送给所有组成员(除 u_{ij})和相邻 CP。为了防止 u_{ij} 获得计算新密钥 CK_{new} 的信息,y 需要用 CP_i 与各接收者的共享密钥分别进行加密。相邻节点和组成员收到 y 后更新密钥参数,处理过程与图 2 相同。

2 安全性分析

1)正确性 根据单向累加器的准交换性,可以证明 CP 和 u_{ij} 利用式(1)和(2)将计算出相同的 CK。

$$CK=\omega_{ij}^{y_{ij}}=B\omega_i^{BY_iy'_{ij}y_{ij}}=B\omega_i^{BY_iy_{ij}^{-1}y_{ij}}=B\omega_i^{BY_i}(\bmod N)^{-1}=x^{Y_{PCP}BY_1\cdots BY_{i-1}BY_{i+1}\cdots BY_kBY_i}=x^{Y_{PCP}BY_1\cdots BY_k}(\bmod N)$$

同样可以证明,当会议成员变化时,通过执行密钥更新过程,u_{ij} 和 CP 将计算出相同的 CK_{new}。下面仅证明新成员加入时的情况。

对于 u_{new}:

$$CK_{new}=\omega_{new}^{y_{new}}=B\omega_i^{r_{new}BY_iy_{new}}=B\omega_i^{BY_ir_{new}y_{new}}=CK^{r_{new}y_{new}}(\bmod N)$$

对于 u_{ij}:

$$CK_{new}=\omega_{ij}^{'y_{ij}}=\omega_{ij}^{y'_{new}y_{ij}}=\omega_{ij}^{y_{ij}r_{new}y_{new}}=CK^{r_{new}y_{new}}(\bmod N)$$

对于 CP_i:

$$CK_{new}=B\omega_i^{BY'_i}=B\omega_i^{BY_iy'_{new}}=CK^{r_{new}y_{new}}(\bmod N)\ \text{或}$$

$$CK_{new}=B\omega_i^{'BY_i}=B\omega_i^{y'_{new}BY_i}=B\omega_i^{BY_ir_{new}y_{new}}=CK^{r_{new}y_{new}}(\bmod N)$$

2)前向保密性 前向保密性指离开会议的成员无法获得之后的会议密钥。当 u_{ij} 离开时,CP_i 将其份额从会议密钥中移除,并利用 $CK_{new}=CK^{ry_{ij}^{-1}} \bmod N$ 更新密钥。由于 r 是 CP_i 随机产生的秘密质数,同时单向累加器在强 RSA 假设下无碰撞[7](即 CK 等于 CK^r 的概率可忽略不计),因此 u_{ij} 无法计算出 CK_{new},实现了前向保密性。

3)后向保密性 后向保密性指新加入会议的成员无法获得之前的会议密钥。当 u_{new} 加入时,CP_i 为其生成的证人信息是 $\omega_{new}=B\omega_i^{r_{new}BY_i} \bmod N$,而当前会议密钥为 $CK=B\omega_i^{BY_i} \bmod N$。由于 r_{new} 是随机产生的秘密质数,同时在强 RSA 假设下 ω_{new} 等于 CK 的概率可以忽略不计,因此新用户无法获得 CK,实现了后向保密性。

4)密钥新鲜性 由上述分析可知,每当会议成员关系发生变化时,总有一个相应的 CP_i 产生秘密随机数去更新会议密钥,新的会议密钥与旧的会议密钥相同的概率可忽略不计,从

而保证了密钥的新鲜性。

3　性能分析

3.1　理论分析

假设会议节点和 CP 的数目分别为 N 和 k,则每组平均节点数量为 N/k。下面从存储开销和计算开销两方面分析 HOAKA 算法的性能。

1)存储开销　普通节点 u_{ij}需要存储与 CP_i 的共享密钥、y_{ij}、ω_{ij}和 CK,密钥存储复杂度为 O(1)。控制节点 CP_i 需要存储与各相邻 CP 的共享密钥,与本组各成员的共享密钥,本组各成员的秘密份额,各子节点的组分支份额,Y_i、BY_i、$B\omega_i$ 和 CK,密钥存储复杂度为 O(N/k)。

2)计算开销　计算开销用每个参与成员进行的幂运算、逆运算、乘运算的次数来衡量。在密钥建立阶段,SCP 平均执行 N/k 次幂运算,N/k 次逆运算和 $2N/k-1$ 次乘运算,计算复杂度为 O(N/k);PCP 执行 $N/k+k-1$ 次幂运算, N/k 次逆运算和 $2N/k+k-1$ 次乘运算,计算复杂度为 O($N/k+k$);普通节点执行 1 次幂运算,计算复杂度为 O(1)。在密钥更新阶段,当有节点加入时,CP 最多执行 2 次幂运算和 4 次乘运算;当有节点离开时,CP 最多执行 1 次幂运算、3 次乘运算和 1 次逆运算。普通节点仅需要执行 2 次幂运算来更新 CK。

3.2　实验测试

实验环境:将 HOAKA 部署在 SIPDHT[10] 上,通过 C 编程实现会议密钥协商算法。本实验是在由 80 台计算机(Pentium4 3.0 GHz CPU,512 MB 内存,Red Hat Linux 操作系统)组成的局域网环境内完成,每台计算机作为一个 P2PSIP 节点,所有节点构成 CAN 重叠网。

实验 1　CP 与普通节点处理负荷的比较

实验 1 比较了不同会议规模下普通节点与 CP 处理负荷的差异,结果如图 3 所示。当会议规模在 380 变化时,普通节点的处理负荷基本保持不变,大约在 1.5% 左右;而 CP 的处理负荷随着会议节点数的增加而迅速增加,当会议成员数达到 30 后,CP 的负荷基本保持在 20% 左右。这是由于实验中将组的上限设为 30,一旦超过,CP 会引入新的控制节点以分担其负荷。显然,HOAKA 将处理负荷主要集中于 CP,普通节点的计算和存储开销均很低,而 CP 在过载时可以引入新的 CP 以实现负载均衡,因此具有良好的可扩展性。

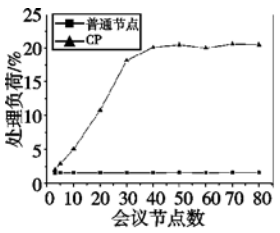

图3　节点处理负荷比较

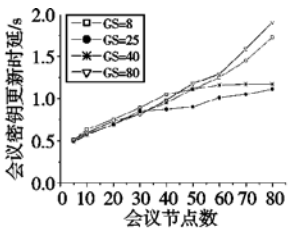

图4　成员加入时密钥更新时延比较

实验 2　组规模对会议密钥更新时延的影响

图 4 和 5 显示了组规模(group size,GS)分别为 8、25、40 和 80 时,会议成员加入和离开时密钥更新时延的性能。实验结果表明,随着会议规模的增加,密钥更新时延均有上升趋势。当会议规模小于 30 时,GS 取值的不同对密钥更新时延的影响并不明显;但是当会议规模大于 40 后, GS = 25 和 GS = 40 的性能较好,GS = 8 次之,GS = 80 最差。因为在本实验中,GS = 80 相当于集中式的会议模型,其更新时延随会议人数的增加而近似线性增加;GS = 8 时的组规模过小,将增加分组数量以及组间的更新开销,从而部分抵消了分组对会议性能的改善。

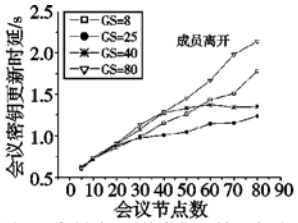

图5　成员离开时密钥更新时延比较

4　结束语

本文以单向累加器算法为基础,提出一种基于分组的 P2PSIP 会议密钥管理方案。该方案不仅数据传输时延低、可扩展性好,而且能够提供会议密钥的前向保密性和后向保密性。实验结果表明,该方案具有较低的密钥更新时延,同时普通节点的处理负荷很轻,能够满足资源受限的移动终端的通信需求。

参考文献:

[1] SINGH K, SCHULZRINNE H. Peer-to-peer Internet telephony using SIP[C]//Proc of International Workshop on Network and Operating Systems Support for Digital Audio and Video. New York:ACM Press, 2005:63-68.

[2] JENNINGS C, LOWEKAMP B, RESCORLA E, *et al*. Resource location and discovery (RELOAD)[EB/OL]. (2008-02-24)[2010-09-12]. http://tools.ietf.org/html/draft-bryan-p2psip-reload-03.

[3] ROSENBERG J, SCHULZRINNE H, CAMARILLO G, *et al*. RFC 3261, SIP:session initiation protocol[S]. [S.l.]:IETF, 2002.

[4] SONG H, MATUSZEWSKI M, YORK D. P2PSIP security overview and risk analysis [EB/OL]. (2009-09-28)[2010-01-11]. http://tools.ietf.org/id/draft-matuszewski-p2psip-security-requirements-06.txt.

[5] CHALLAL Y, SEBA H. Group key management protocols: a novel taxonomy[J]. International Journal of Information Technology, 2005, 2(1):105-118.

[6] WANG Yao, ZHANG Chun-hong, MA Tao, *et al*. Design and evaluation of reliability mechanisms in P2PSIP-based conference system [C]//Proc of the 4th International Conference on Wireless Communications, Networking and Mobile Computing. Washington DC: IEEE Press, 2008: 3261-3266.

[7] BARIC N, PFITZMANN B. Collision-free accumulators and fail-stop signature schemes without trees[C]//Proc of EUROCRYPT'97. Berlin: Springer-Verlag, 1997: 480-494.

[8] RING J, CHOO K-K R, FOO E, *et al*. A new authentication mechanism and key agreement protocol for SIP using identity-based cryptography[C]//Proc of AusCERT Asia Pacific Information Technology Security Conference. Brisbane: University of Queensland Publication, 2006:57-72.

[9] 施苑英. 一种高效的 P2PSIP 认证与密钥协商机制[J]. 计算机应用研究,2011,28(1): 235-237.

[10] FANDRIANTO A. SIPDHT: an open source project for P2PSIP[EB/OL]. (2007-07)[2010-10-02]. http://sipdht.sourceforge.net/sipdht2/report-fandrianto-sipdht2.pdf.