基于 PKI 防欺诈的门限密钥托管方案

张春生[1] 王世普[1] 姚绍文[1,2] 张险峰[2]
(云南大学软件学院 昆明 650091)[1] (电子科技大学计算机学院 成都 610054)[2]

摘 要 本文基于 PKI(Public Key Infrastructure)提出了一个防欺诈的密钥托管方案,并提出了密钥托管分配机制。由于该方案的秘密共享采用基于状态树的(t,n)秘密共享,所以具有计算量小,效率高的特点。这个方案不仅有效地解决了用户欺诈的问题,而且托管用户可根据需要而改变托管代理的数量。本文在数字证书的发放与密钥托管之间建立了一种机制,把两者联系在一起,以真正实现密钥托管的目的。

关键词 公钥基础设施(PKI),密钥托管,托管代理,状态树,欺诈

A Key Escrow Scheme to Identify Cheaters Based on PKI

ZHANG Chun-Sheng[1] WANG Shi-Pu[1] YAO Shao-Wen[1,2] ZHANG Xian-Feng[2]
(Software College, Yunnan University, Kunming 650091)
(Computer College, Electronic Science and Technology University, Chengdu 610054)[2]

Abstract In this paper, we propose a secret key escrow scheme to Identify Cheaters based on PKI(Public Key Infrastructure), At the sametime, we propose a distribution mechanism of key escrow. Since this threshold is a status-tree based (t,n)secret sharing scheme, it hold characteristic with the more efficiency and smaller computing. The proposed scheme not only solves the exterior problem to Identify Cheaters , but also escrow user can alter amount of escrow agent according to requirement. The paper build a mechanism between digital certificate and key escrow. The both are consociated so that we can realize actual purpose of key escrow.

Keywords Public key infrastructure (PKI), Key escrow, Escrow agent, Status-tree, Cheating

1 引言

在利用计算机网络进行通信时,特别是电子商务,人们面临的最大问题是如何建立相互之间的信任关系以及如何保证信息的真实性、完整性、机密性和不可否认性。PKI(Public Key Infrastructure)[1]则是解决这一系列问题的技术基础。PKI 就是利用公钥理论和技术建立的提供信息安全服务的基础设施。它作为电子商务和信息系统的安全基础设施,负责其通信系统的安全可靠性。它的基础是加密技术,核心是证书服务,由证书授权认证中心 CA 来完成。ITU(International Telecommunication Union)定义 CA 是一个可信任的第三方机构(Trusted Third Party, TTP),专门制造与发放数字证书[2]。用户使用自己的数字证书,结合加密技术,保证通信内容不被泄露、篡改和破坏。

在 PKI 中,每个人在网络上的身份由他对应的数字证书来确定。数字证书是 CA 签署、由计算机生成的记录,包含由 CA 验证过的该证书持有者的个人信息及其对应公钥,然后 CA 利用本身的密钥为数字证书加上数字签名。CA 目前采用的标准是 X.509 V3 标准,其中包括 IPSec(IP 安全)、SSL、SET、S/MIME。

执行密钥托管功能的机制是密钥托管代理 KEA(Key Escrow Agent),密钥托管 KE 又称为密钥恢复(Key Recovery),它用于保存用户的私钥备份,使得政府职能部门在不被通信双方察觉的情况下,实现强制接入,破译截获的密文。密钥托管的另一个重要用途是当用户丢失、损坏自己的私钥或公司企业内部进行管理监督时,恢复密文。

证书授权认证中心 CA 与密钥托管代理 KEA 是 PKI 的两个重要组成部分,从职责上来说,CA 与 KEA 在 PKI 中担任完全不同的角色。CA 发放数字证书,在每次通信过程中提供鉴别验证服务,但不涉及具体通信内容。其主要操作针对用户的公钥进行。而 KEA 管理用户的私钥,对用户私钥进行操作,负责政府职能部门的强制访问,不参与用户的通信过程。它们分别对用户的公钥与私钥进行操作。

本文从保护用户托管私钥安全性与防欺诈的角度出发,研究一种基于 PKI 的安全密钥托管方案,在数字证书的发放与密钥托管之间建立了一种机制,把两者联系在一起,以真正实现密钥托管的目的。

为了保证信息的机密性,本文的密钥托管技术采用椭圆曲线密码体制(ECC)[3]。同时,为了防止密钥托管代理滥用职权及托管密钥被泄漏,用户的私钥分配采用基于状态树的(t,n)门限秘密共享方案。

2 基于状态树(t,n)门限秘密共享方案[4]

基于状态树的门限秘密共享方案的核心思想是构造一棵状态树,树中每个结点表示某个服务器分配到一份影子时的状态,状态信息包括:服务器编号 ID、影子值 sd、影子序号 sd-

张春生 硕士研究生,主要研究方向为信息和网络安全。王世普 教授,硕士生导师,主要研究方向为网络分布式应用、网络信息处理。姚绍文 博士,教授,主要研究方向为语义 web 技术、网络协议工程、网络分布式计算、着色 Petri 网(CPN)建模、知识工程技术等。张险峰 博士研究生,主要研究方向为信息和网络安全。

No。

2.1 基本定义

定义 1 将一秘密值 d 分割成 n 份，每份称为一个影子(子密钥)，然后将此 n 份影子分别存储在 n 个不同的服务器上，每个服务器称为影子服务器，只有通过 t 份或 t 份以上影子才能恢复秘密值 d，此种秘密共享方案称为(t, n)秘密共享。

定义 2 一棵状态树中每个结点表示某个服务器分配到一份影子时的状态，状态信息包括：服务器编号 ID、影子 sd、影子序号 $sdNo$。

定义 3 有序树，将树中结点的各子树看成从左至右是有次序的(即不能互换的)，树中最左边的子树的根称为第一个孩子，最右边的称为最后一个孩子。

定义 4 结点的层次，根为第 0 层，根的孩子为第 1 层。若某结点在第 l 层，则其孩子就在 $l+1$ 层。

2.2 实现方法

在构建一棵状态树时，可采用如下的方法进行：

(1)采取逐层递进的方式来建立此树，即先建完 i 层所有结点，再建第 i 层所有结点的子结点(即第 $i+1$ 层结点)；

(2)在同一层里，各结点的子结点按服务器编号从小到大生成。

对该状态树的特征可作进一步限制。

(1)根结点 *root* 的服务器编号、影子、影子序号皆为 0，表示还未开始对服务器进行影子分配的状态；

(2)树的深度为 t；

(3)非叶结点的影子 sd 随机产生，在以非叶结点为根的子树中，各结点的服务器编号不能和此非叶结点的服务器编号相同；

(4)不同结点的服务器编号相同时，影子序号 $sdNo$ 可以由该结点服务器编号出现的次数决定；

(5)各叶结点的层次均为 t，叶结点的影子 sd 为共享秘密 d 减去从根结点到此结点的路径上所有其他结点影子的和。

2.3 算法描述

输入：根结点 *root*，非负整数 t、n、d

输出：一棵具有 2.2 节所描述的特性的状态树

算法 1：CreateTree (*root*, t, n, d)

1. 初始化根结点，初始化数组 index[t]、count[t].
2. If $t>n$ OR $t=0$ then stop. //特殊情况
3. For i from 1 to $n-t+1$ do //生成第 1 层结点
   - 3.1 生成根结点的子结点(第 1 层结点)$server_1(i)$
   - 3.2 Set index[i] ← index[i]+1 //表明 ID 为 i 的服务器结点出现的次序
   - 3.3 Set count[1] ← count[1]+1 //统计第 1 层的结点数
   - 3.4 Set $server_1(i).ID \leftarrow i$, $server_1(i).sdNo \leftarrow$ index[i]+1
   - 3.5 If $t>1$ then set $server_1(i).sd \leftarrow$ 随机数, else set $server_1(i).sd \leftarrow d$ and stop
4. For i from 2 to t do //生成第 i ($2 \leqslant i \leqslant t$)层结点
   - 4.1 For k from 1 to count[$i-1$] do //在已建的有序树中，从左至右依次读入第 $i-1$ 层的 count[$i-1$]个结点
     - 4.1.1 If $i=t$ then{从结点 $server_{i-1}(k)$ 回溯到根结点的路径，计算此路径上 t 个结点的 sd 之和 sum(sd)，并且 set $x \leftarrow d-$sum(sd).}
     - 4.1.2 For j from $server_{i-1}(k).ID+1$ to $n-t+i$ do
       - 4.1.2.1 生成以 $server_{i-1}(k)$ 为双亲结点的新结点 $server_i(j)$
       - 4.1.2.2 Set index[j] ← index[j]+1 //统计 ID 为 j 的结点出现的次序
       - 4.1.2.3 Set $server_i(j).ID \leftarrow j$, $server_i(j).sdNo \leftarrow$ index[j]
       - 4.1.2.4 If $i \neq t$ Set $server_i(j).sd \leftarrow$ 随机数, else $server_i(j).sd \leftarrow x$
       - 4.1.2.5 Set count[i] ← count[i]+1. //计数第 i 层生成的结点个数
5. Stop

当状态树生成以后，从根结点到叶子结点的一条路径就对应了一种 t 台服务器参与运算重建秘密的情形。

我们设计了基于状态树的秘密重建算法 Reconstruct( )，其目的是：当指定 t 台服务器，根据生成的状态树，能够给出这 t 台服务器上参与重建的影子。

输入：根结点 *root*，选定的 t 台服务器 SelectedServer[t] (编号由小到大排列)

输出：参与运算的 t 台服务器所使用的影子 sd

算法 2.2 Reconstruct (*root*, SelectedServer)

1. 生成临时结点指针 *current_prt*，将 *current_prt* 指向根结点 *root*.
2. *For* i *from* 1 *to* t−1 *do*
   - 2.1 查找 *current_prt* 的孩子结点中 ID 为 SelectedServer[i]的孩子，并将 *current_prt* 指向此孩子结点.
   - 2.2 输出 *current_prt*. ID 和 *current_prt*. $sdNo$.
3. Stop

2.4 建树实例

设 $t=2$，$n=4$，设参与运算的影子服务器的编号为：1，2，3，4，共享的秘密为 d，r1，r2，，r3 为随机整数，则则基于算法 2.1 生成的状态树如图 1 所示。

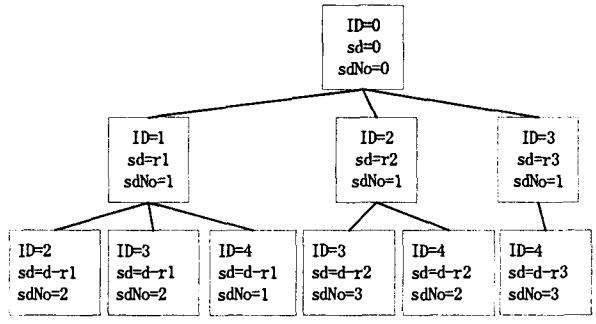

图 1 基于状态树的(2,4)秘密共享

一旦生成该状态树，当要进行秘密值 d 的重建时，非常简单。dealer 只需从 4 台服务器上任指定 2 台服务器，其编号按从小到大的顺序排列，然后执行算法 2.2 得到每台服务器所使用的影子序号，然后 dealer 向这些被选定的服务器请求对应的影子，完成相应的密码操作。

2.5 方案的分析

2.5.1 *方案的性质* (1)直观，简洁，各服务器上不存在冗余。(2)机密性，t 台以下的影子服务器不可能重建秘密值

d。而且本共享方案是完善的，任何小于 t 台的影子服务器组得不到有关秘密的任何信息。(3)可用性：秘密值 d 可以通过 $t-1$ 台以上的影子服务器来进行重建。

2.5.2 *时间复杂度* 对于算法 2.1 生成的树，其层次最高为 t 层，每层的结点数小于等于 $\binom{n}{t}$ 个，其计算复杂度小于 $o(t\binom{n}{t})$，算法为多项式复杂度的算法。

对于算法 2.2，当根据指定的服务器组查找所用的影子时，只是线性查找过程，算法为线性复杂度 $o(t)$。

3 基于 PKI 的密钥托管分配机制

密钥托管技术最关键，最难解决的问题是：如何有效地阻止用户的欺诈行为，使所有用户都无法逃脱托管机构的跟踪。所以为了防止用户逃避托管，密钥托管技术的实施需要通过政府的强制措施进行。用户必须先委托密钥托管代理对自己的私钥进行托管，取得托管证书后才能向证书授权认证中心 CA(Certificate Authority)申请加密证书。证书授权认证中心必须在收到加密公钥对应的私钥托管证书后，才能签发相应的公钥证书。

3.1 密钥托管分配机制

在整个密钥托管系统中，密钥托管的用户，CA 和 KEA，他们共享椭圆曲线 E(Fp)(p 为大于 3 的素数)包含一个循环子群 H，其阶为 $|H|$，在 H 中离散对数问题是难解的，g 为 H 的生成元。

整个密钥托管系统的密钥分配机制框架如图 2 所示。

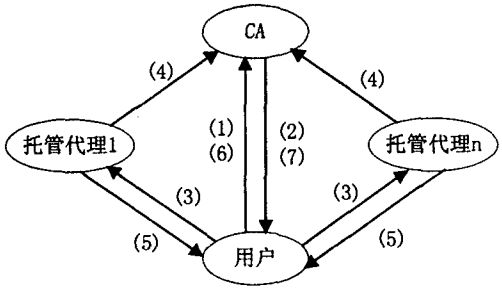

图 2 密钥分配机制框架图

图中箭头上的序号代表密钥分配机制的步骤。

密钥托管技术的实现可分为三个阶段：用户的私钥托管阶段、用户间的通信阶段、监听机构的监听阶段。这里主要分析用户的私钥托管阶段。

(1)首先用户向 CA 申请到公钥签名证书(此证书无需委托签名)；然后利用自己的公钥签名证书证明身份后，继续向 CA 申请公钥加密证书。

(2)若 CA 发现没有收到用户的密钥托管证书，则向用户发送索要信息并附带 CA 推荐的 $m(\geqslant n)$ 个可信赖的托管代理的公钥签名证书供用户选择。否则，执行步骤(7)。

(3)用户选择 n 位托管代理后，执行第二节算法 2.1，将自己的私钥 d 分配给 n 个托管代理，每一位托管代理 $T_i(i=1,2,\cdots,n)$ 的私钥为 d_i，并将 (d_i,P_i)(P_i 为托管代理 T_i 的公钥)秘密地发给托管代理。

(4)托管代理向 CA 验证用户的身份之后，通过公式

$$P_i=d_ig \bmod p$$

验证 (d_i,P_i) 的正确性。

(5)若正确，则托管代理产生相应的托管证书。托管证书包含该用户的特定标示符 UID、公钥 P_i、托管代理的标示符和托管证书号。托管代理用自己的签名私钥对此信息签名后，发给用户。否则，转步骤(3)。

(6)用户收到信息后，验证信息的正确性。然后把托管证书以及与私钥 d 对应的公钥 $Q(=dg \bmod p)$ 交给 CA 申请公钥加密证书。

(7)CA 收到信息后，验证用户身份和数据的完整性，并且验证托管证书的真实性，并进一步验证公式

$$Q=(P_1+P_2+\cdots+P_n) \bmod p$$

若成立，证明与公钥 Q 对应的私钥 d 确实进行了托管。此时，完成所有验证工作后，CA 即产生对应于 Q 的公钥加密证书，并返回给用户。

为了监听，CA 记载了用户所有托管代理 $T_i(i=1,2,\cdots,n)$ 的托管证书号。拥有监听权利的监听机构可访问 t 个托管代理即可重构出用户的私钥 d 而实现监听功能。

3.2 密钥托管分配机制分析

本密钥托管分配机制的信息机密性依赖于椭圆曲线离散对数问题(ECDLP)，用户的私钥采用基于状态树的 (t,n) 门限秘密共享方案，该方案直观、简洁，能够有效地实现秘密的分割与重建。

3.2.1 *安全性* 在本密钥托管分配机制中，敌手要想从 $Q=dg \bmod p$ 中获得用户的私钥，则他必须解椭圆曲线离散对数问题(ECDLP)。同时本密钥托管分配机制还具有防用户欺诈、防外部欺诈和托管代理数量可变的功能。

3.2.2 *防用户欺诈* 防用户欺诈行为就是使所有用户都无法逃脱密钥托管的机制。阻止用户的欺诈行为可以通过以下步骤得到保障：

(1)为确保用户密钥的托管，用户在向 CA 申请公钥加密证书时，必须提交密钥的托管证书。

(2)为了防止用户欺骗托管代理，当托管代理 $T_i(i=1,2,\cdots,n)$ 收到用户的密码片 (d_i,P_i) 时，需验证：

$$P_i=d_ig \bmod p$$

是否成立。

(3)为了防止用户欺骗 CA，当 CA 收到用户的申请与托管代理的证书时，需验证用户是否确实将私钥 d 进行了托管。用 $Q=(P_1+P_2+\cdots+P_n) \bmod p$ 来验证。

证明：
$$\begin{aligned}Q&=(P_1+P_2+\cdots+P_n) \bmod p\\&=(d_1g+d_2g+\cdots+d_ng) \bmod p\\&=(d_1+d_2+\cdots+d_n)g \bmod p\\&=dg \bmod p = Q \bmod p\end{aligned}$$

3.2.3 *防止外部欺诈* 为了防止外部欺诈，即防止敌手获得秘密 d，可以在不改变秘密 d 的情况下，定期(根据共享秘密的需要，确定更新子秘密的周期，可以是一天、一周、一月或一年等)更新各托管代理 $T_i(i=1,2,\cdots,n)$ 所拥有的子秘密 d_i，这样在子秘密更新后，敌手在以前各周期中所窃取的不足 t 个子秘密的信息不再有用。因为为了重构秘密 d，必须在同一个周期内至少获得 t 个子秘密。定期更新只需执行密钥托管分配机制中的步骤(3)～(7)。

3.2.4 *更改托管代理数量* 当用户需要改变托管代理

(下转第 80 页)

与现在 Srinivas Mukkamala[3]等正在作将支持向量机用于入侵检测相比，如表 4。

与 Wenke Lee 等用数据挖掘作入侵检测的结果相比较，在检测新型攻击方式方面，效果更好，因为模糊支持向量机对需要着重分析的数据（攻击数据）进行了优化（参看 3.1），这也是模糊支持向量机用于入侵检测的特点和优势。

与 Srinivas Mukkamala 等的结果相比，差距不小，主要原因是他将各种攻击类型分类，然后再作检测，所以准确率非常高，但从实用的角度讲，不可能攻击会告诉你它是哪个攻击类型，所以我们运用的检测方式可以拿来实际运用。

表 4

Class	Srinivas mukkamala 等的结果	我们的结果
Normal	99.55%	78.81%
Probe	99.70%	
DOS	99.25%	
U2SU	99.87%	
R2L	99.78%	

下一步工作 本文所做贡献主要是首次将模糊支持向量机引入网络入侵检测中，取得了理想的效果，检测新型攻击手段能力更强，也具有实用的价值，进一步推广了模糊支持向量机理论的实际应用。

下一步的工作是在一开源网络入侵检测系统 Bro 上，引入我们的 FSVM 检测模块，争取实现一实用的智能网络入侵检测系统。

参 考 文 献

1 Lin Chun-Fu. Fuzzy Support Vector Machines. IEEE TRANSACTIONS ON NUERAL NETWORKS, 2002, 13(2): 3

2 Vapnik V N. An Overview of Statistical Learning Theory. IEEE TRANSACTIONS O NEURAL NETWORKS, 1999, 10(5)

3 Mukkamala S, Sung A H. Artificial Intelligent Techniques for Intrusion Detection. 0-7803-7952-7/03 2003 IEEE

4 Paxson V. Bro: A System for Detecting Network Intruders in Real-Time. Computer Networks, 1999

5 http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

6 Forrest S. Self-Nonself discrimination in a computer. CA: IEEE Computer Society Press, 1994

7 Ghosh AK. A real-time intrusion detection system based on learning program behavior. Recent Advances in Intrusion Detection (RAID 2000). Toulouse: Spinger-Verlag, 2000

8 Lee W, Stolfo SJ. A data mining framework for building intrusion detection model. CA: IEEE Computer Society Press, 1999

9 http://www.cert.org/stats/cert_stats.html

10 张学工. 关于统计学习理论与支持向量机. 自动化学报, 2000(1)

（上接第 74 页）

$T_i(i=1,2,\cdots,n)$的数量时，可以在执行算法 2.1 时改变输入 t,n 的值，从而改变托管代理 T_i 的数量。

3.2.5 计算量 在计算量上，主要集中在子密钥的产生。本密钥托管分配机制的子密钥产生采用的是基于状态树的(t,n)门限秘密共享方案，我们将此方案与现在广泛采用的基于 Shamir 的(t,n)门限秘密共享方案作一比较。

1. 基于 Shamir 的(t,n)限秘密共享方案

Shamir 秘密共享[5]是一个基于多项式插值的(t, n)门限秘密共享方案。首先选取有限域 Z_q 中的 n 个不同的非零元，记为 x_i，$1\leqslant i\leqslant n$，然后在 Z_q 上随机选择一个次数为$(t-1)$的多项式 $f(x)=f_0+f_1x+\cdots+f_{t-1}x^{t-1}(\mathrm{mod}\ q)$，满足 $f(0)=f_0=d$。$f_1,f_2,\cdots,f_{t-1}$为 Z_q 上的随机数，计算多项式 $d_i=f(x_i)(\mathrm{mod}\ q)$，$1\leqslant i\leqslant n$，$d_i$ 即为托管代理的影子（子密钥）。

从上面的计算过程看出：影子 d_i 的计算共要进行 n 次多项式 $f(x_i)$的计算，算法复杂度为 $t-1$ 次方阶 $o(n^{t-1})$。我们将 $f(x_i)$的模幂计算转化成模乘计算来考虑。多项式 $f(x_i)$的计算共需要$\frac{1}{2}t(t-1)$次模乘和$\frac{1}{2}t(t+1)$次模加运算。因此 d_i 的计算总共需要$\frac{1}{2}t(t-1)\times n$次模乘和$\frac{1}{2}t(t+1)\times n$次模加运算。

2. 基于状态树的(t,n)门限秘密共享方案

基于状态树的(t,n)门限秘密共享方案的存取结构共$\binom{n}{t}$个，每一个存取结构需要进行 t 次模加。因此影子 d_i 的计算总共需要$t\binom{n}{t}$模加运算。算法为多项式复杂度。

从上述分析可以看出我们的方案比基于 Shamir 的(t,n)门限秘密共享方案的计算量要小得多。而且随着 t 和 n 的增大，基于状态树的(t,n)门限秘密共享的密钥分配机制比基于 Shamir 的(t,n)门限秘密共享效率高得多。

4 相关工作

目前，门限密钥托管方案所采用的秘密共享算法都是基于 Shamir 的(t,n)门限秘密共享，而本文提出的门限密钥托管方案是基于状态树的(t,n)秘密共享。由 3.2 节分析可知，无论在计算量上，还是在效率上，本文的方案均优于基于 Shamir 的(t,n)门限密钥托管方案。

结束语 本文基于 PKI(Public Key Infrastructure)提出了一个防欺诈的密钥托管方案，并提出了密钥托管分配机制。由于该方案的秘密共享采用基于状态树的(t,n)秘密共享，所以具有计算量小，效率高的特点。这个方案不仅有效地解决了用户欺诈的问题，而且托管用户可根据需要而改变托管代理的数量。本文在数字证书的发放与密钥托管之间建立了一种机制，把两者联系在一起，以真正实现密钥托管的目的。

下一步，我们将考虑门限密码系统的有效性研究，即在保证安全性的前提下，怎样使得共享密钥长度尽量小、计算速度尽量快。

参 考 文 献

1 National Institute of Standard and Technology (NIST). PKI technical specifications [DB/OL]. PartA: Requirements. Feb. 1996. http//csrc.ncsl.nist.gov/pki/fordrept.ps

2 Phoenix S J D. Cryptography, trusted third parties and escrow[J]. BT Technology Journal, 1997, 15(2): 45～62

3 徐秋亮，李大兴. 椭圆曲线密码体制. 计算机研究与发展，1999，36(11)：1281～1288

4 张险峰. 基于 ECC 的门限密码体制及其应用的研究：[博士论文]. 2003. 36～43

5 Shamir. How to share a secret[J]. Comm. of the ACM, 1979, 22(11): 612～613