[19] 中华人民共和国国家知识产权局

[51] Int. Cl[7]
H04L 9/32
H04L 9/10
H04L 9/14

[12] 发 明 专 利 说 明 书

[21] ZL 专利号 01136017.8

[45] 授权公告日 2005 年 6 月 22 日

[11] 授权公告号 CN 1207866C

[22] 申请日 2001.9.28 [21] 申请号 01136017.8
[71] 专利权人 中国科学院研究生院
地址 100039 北京市玉泉路(甲)19 号
[72] 发明人 荆继武 冯登国 周天阳
审查员 刘 斌

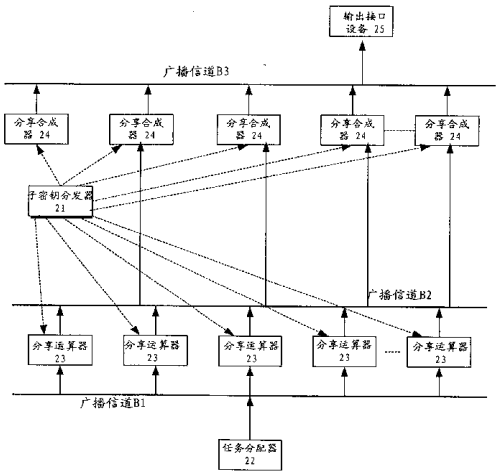
[74] 专利代理机构 北京德琦知识产权代理有限公司
代理人 王丽琴

权利要求书 5 页 说明书 23 页 附图 5 页

[54] 发明名称 一种安全的数字签名系统与方法

[57] 摘要

本发明涉及一种安全的数字签名系统与方法。系统包括通过广播信道连接的任务分配器、k 个运算器、m 个合成器和子密钥分发器。子密钥的发放操作包括：由子密钥分发器将私钥 d 表示成由 t 个子密钥 d_j 及一个子密钥 c_i 的和，并使 $t < k$；分发器将 k 个随机的 d_j 分发到 k 个运算器中，并获得一组 c_i 的方程组合表示，将所有方程组合表示及其对应的 c_i 按合成器安全条件穷搜获得 m 分组，分送到 m 个合成器中预存。数字签名操作包括：任务分配器将需要签名的 HASH 值 M 广播送 k 个运算器；运算器计算升幂 M^{dj}，并广播送合成器；合成器通过与预存的 c_i 的方程组合表示进行比较，找到匹配的方程组合表示并得到相应的 c_i，通过将对应的升幂 M^{dj} 相乘得到结果 R，求出 M^{ci}，最后得到数字签名 $S = M^d$。



ISSN1008-4274

1. 一种安全的数字签名系统，用于得到数字签名，其特征在于：

包括至少一个在线的任务分配器、k个在线的秘密分享运算器、m个在线的秘密分享合成器和离线的子密钥分发器；离线的子密钥分发器在系统初始化或进行系统配置时与 k 个秘密分享运算器及 m 个秘密分享合成器分别连接进行子密钥发放，包括：对应 k 个秘密分享运算器生成随机数，并作为第一子密钥分发到 k 个秘密分享运算器中，和对应 m 个秘密分享合成器计算第二子密钥并预存在 m 个秘密分享合成器中；在线的任务分配器通过第一广播信道与 k 个秘密分享运算器连接，在计算数字签名处理过程中将需要签名的 M 值送 k 个秘密分享运算器，每个秘密分享运算器根据 M 值和第一子密钥进行计算，并将计算结果和 M 值通过第二广播信道送 m 个秘密分享合成器，每个秘密分享合成器根据接收结果、M 值及预存的第二子密钥计算得到数字签名，k、m 为正整数。

2. 根据权利要求 1 所述的一种安全的数字签名系统，其特征在于：还包括一单独设置的输出接口设备，通过第三广播信道与 m 个秘密分享合成器连接。

3. 根据权利要求 1 所述的一种安全的数字签名系统，其特征在于：所述在线的任务分配器中还设置有一输出接口设备，通过所述的第一广播信道与所述的 m 个秘密分享合成器连接。

4. 根据权利要求 1 或 2 或 3 所述的一种安全的数字签名系统，其特征在于：所述的至少一个在线的任务分配器、k 个在线的秘密分享运算器、m 个在线的秘密分享合成器、离线的子密钥分发器均采用普通计算机或服务器。

5. 根据权利要求 1 或 2 或 3 所述的一种安全的数字签名系统，其特征在于：所述的第一广播信道、第二广播信道及第三广播信道是物理相连的一个信道或完全不相连的独立信道。

6. 一种安全的数字签名方法，包括子密钥的发放和计算数字签名，其特征在于：

所述的子密钥的发放包括以下处理步骤：

A. 设置一个安全数字签名系统，包括在线的任务分配器、在线的 k 个秘密分享运算器、在线的 m 个秘密分享合成器、广播信道和离线的子密钥分发器，k、m 为正整数；

B. 离线的子密钥分发器将保存的数字签名私钥 d 表示成由 t 个第一子密钥 d_j 及一个第二子密钥 c_i 的和，d、t、c、j、i 均为正整数并让 t<k，j 是第一子密钥 d_j 变量的序号；

C. 离线的子密钥分发器将 k 个随机数作为第一子密钥 d_j，对应分发到 k 个秘密分享运算器中，并根据步骤 B 中的和式通过相减获得一组第二子密钥 c_i 及其方程组合表示，将全部第二子密钥 c_i 及其方程组合表示放在一个大组中；

D. 离线的子密钥分发器按合成器安全条件对该大组中的全部第二子密钥 c_i 及其方程组合表示进行穷搜，获得 m 个分组的第二子密钥 c_i 及其方程组合表示：

E 离线的子密钥分发器将 m 个分组的第二子密钥 c_i 及其方程组合表示对应送到 m 个秘密分享合成器中预存；

所述的计算数字签名包括以下处理步骤：

F. 在线的任务分配器将需要签名的 HASH 值 M 通过广播数据包经第一广播信道送往 k 个秘密分享运算器；

G. k 个秘密分享运算器中的 t 个或 t 个以上的秘密分享运算器根据接收的 M 值计算升幂 M^{d_j}，并将本秘密分享运算器的代号 j、需要签名的 HASH 值 M 及计算结果 M^{d_j} 通过广播数据包经第二广播信道送往 m 个秘密分享合成器；

H. m 个秘密分享合成器将接收结果与预存分组的第二子密钥 c_i 的方程

组合表示进行比较，找到可以匹配的方程组合表示并得到相应的第二子密钥 c_i，再将与组合匹配的 t 个秘密分享运算器的升幂运算结果相乘得到结果 R，根据找到的第二子密钥 c_i 求出 M^{c_i}，最后将 M^{c_i} 与 R 相乘得到数字签名 $S=M^d$。

7. 根据权利要求 6 所述的一种安全的数字签名方法，其特征在于：所述的步骤 A 中还包括给在线的任务分配器任意给定一个代号、给 k 个秘密分享运算器分别给定不相同的代号、给 m 个秘密分享合成器分别给定不相同的代号和设定 t 的初始值。

8. 根据权利要求 6 所述的一种安全的数字签名方法，其特征在于：所述的步骤 C，进一步包括：

c1.由离线的子密钥分发器取 k 个小于 d/t 的随机数作为第一子密钥 d_j;

c2.由离线的子密钥分发器通过管理许可的方式将 k 个第一子密钥 d_j 对应送到 k 个秘密分享运算器中；

c3.由离线的子密钥分发器从 k 个第一子密钥(d_1、d_2、d_3、…、d_k)中取出 t 个 d_j，按组合算式形成 $n=C_k^t$ 种组合表示，并根据所述的和式计算所有组合表示所对应的第二子密钥 c_i，(i=1、2、…、n)，构成所述的方程组合表示大组。

9. 根据权利要求 6 所述的一种安全的数字签名方法，其特征在于：

所述的步骤 D 进一步包括：

d1.对所述的大组中的组合随机设定一个顺序，并随机设定一个空的分组 B;

d2.从大组中顺序取出一个第二子密钥 c_i 及其方程组合表示，记为组合表示 A，判断 A 放入 B 中是否满足合成器安全条件，如果满足就放入分组 B 中并从大组中删除该组合及对应的第二子密钥，如果不满足合成器安全条件，就将该组合及对应的第二子密钥仍旧保持在大组中；

d3.针对大组中所有的组合重复执行步骤 d2;

d4.如果大组中还有组合表示，确定第二个空的分组为 B，并重复执行步骤 d2、d3，直至大组中已没有任何第二子密钥 c_i 及其方程组合表示时为止；

d5.统计形成的不空的分组个数 m，是所述的 m 个分组。

10. 根据权利要求 9 所述的一种安全的数字签名方法，其特征在于所述的判断是否满足合成器安全条件，进一步包括：将组合表示 A 中变量的序号与分组 B 中的组合表示 q_s 中变量的序号进行比较，如果组合表示 A 中有且组合表示 q_s 中没有的变量的序号个数大于 t/2，该组合表示 A 对组合表示 q_s 是满足安全条件的；如果组合表示 A 对分组 B 中的每一个组合都满足安全条件，则组合表示 A 放入分组 B 中是满足合成器安全条件的。

11. 根据权利要求 6 所述的一种安全的数字签名方法，其特征在于：所述的步骤 E，由离线的子密钥分发器通过管理允许的方式将所述 m 个分组的第二子密钥 c_i 及其对应的方程组合表示对应分送到所述的 m 个秘密分享合成器中。

12. 根据权利要求 6 所述的一种安全的数字签名方法，其特征在于：所述的离线的子密钥分发器在执行完步骤 A、B、C、D、E 完成子密钥的分发后，处于物理隔离状态或处于关机状态。

13. 根据权利要求 6 所述的一种安全的数字签名方法，其特征在于：所述的步骤 F 进一步包括：

f1.由在线的任务分配器接收数字签名任务，并进行安全检查与核对；

f2.由在线的任务分配器为该任务确定一个在预定的时间内对任务分配器是唯一的任务号；

f3.由在线的任务分配器将其代号及任务号随需要签名的 HASH 值 M 组成所述的广播数据包广播到所述的第一广播信道上；

所述的步骤 G 进一步包括：

g1.接收到广播后的 t 个或 t 个以上的秘密分享运算器向该在线的任务分配器发出已经接收的通知；

g2.t 个或 t 个以上的秘密分享运算器检查任务的唯一性，在确定为新任务时进行所述升幂的计算；

g3.t 个或 t 个以上的秘密分享运算器还将该在线的任务分配器的代号、任务的任务号随本秘密分享运算器的代号 j、需要签名的 HASH 值 M 及计算结果 M^{d_j} 组成所述的广播数据包广播到所述的第二广播信道上；

所述的步骤 H，进一步包括：

h1.接收到广播的秘密分享合成器将具有相同任务分配器代号及任务号的广播数据包放在一组中；

h2.上述秘密分享合成器至少找出 t 个广播数据包，再从中找出与所述预存的方程组合表示匹配的一个方程组合表示，并得到对应的第二子密钥 c_i。

14. 根据权利要求 6 或 13 所述的一种安全的数字签名方法，其特征在于：所述的步骤 F、G、H 是顺序执行完成数字签名的计算操作。

15. 根据权利要求 6 所述的一种安全的数字签名方法，其特征在于：

所述的步骤 H 后还包括一步骤 I，由在线的秘密分享合成器，将数字签名 $S=M^d$ 与任务分配器代号、任务号组成广播数据包广播到一在线的输出接口设备上，并由该输出接口设备用公开密钥检查签名结果，是正确时结束数字签名，是错误时进行错误处理或告警。

16. 根据权利要求 15 所述的一种安全的数字签名方法，其特征在于：所述步骤 I，是由在线的秘密分享合成器通过一第三广播信道将所述的广播数据包广播到一单独设置的在线的输出接口设备上。

17. 根据权利要求 15 所述的一种安全的数字签名方法，其特征在于：所述步骤 I，是由在线的秘密分享合成器通过所述的第一广播信道将所述的广播数据包广播到设置在任务分配器中的输出接口设备上。

一种安全的数字签名系统与方法

技术领域

本发明涉及网络通信安全技术，更确切地说是涉及一种能容忍入侵的确保数字签名安全的系统与方法。

背景技术

数字签名(Digital Signature)是当今网络通信安全中最基本的技术，是利用非对称算法(Asymmetric Algorithm)，来达到其他人可以验证该签名但却无法假冒该签名的目的。最常用的非对称算法有RSA、DSA和椭圆曲线算法等，目前不少数字签名系统都是基于RSA算法的。

所谓非对称算法，就是一个人无法通过已知的正向计算参数推导出反向计算的参数，即已知正向计算过程，无反向计算能力。这种非对称算法本身是公开的，但每个人可以选择不同的参数，参数不同，所构成的变换函数就不同。对某个人来说，他可以选择一组参数，其中一部分是用于逆向计算的，称为秘密参数，技术上称为保密密钥或私钥；另一部分是用于正向计算的，是公开的参数，技术上称为公开密钥或公钥。

数字签名就是基于这种非对称算法得以实施的。一方面保护自己的秘密参数－私钥，以保证他人无法冒充自己进行签名，另一方面通过公开可以公开的部分－公钥，供相关人士能够验证该签名（从理论上说，利用公开的参数来推导秘密的参数，在计算上是不可行的）。

数字签名首先是网络通信与交互的保证，可以保证通信对方是真实的，可以保证在网上的就是自己，还可作为签署电子文档时的工具，以保护自己的文档与签名。当今许多国家，已将数字签名视作手写签名，两者具有同样的法律效率。

数字签名算法还可以用于协商秘密（secret）参数。如A用户需要与B用户进行秘密通信时，A用户可以先设定一些秘密参数，然后用B用户的公开密钥（公钥）作用一下，此时只有B用户能够恢复原来的值，因为只有B用户知道自己的保密(private)密钥（私钥）。

此外，数字签名还可以用于需要保密的场所、需要身份认证的场所及其他需要不可否认服务的场所。

现在国际上大力推行的PKI（Public Key Infrastructure: 公开密钥基础设施）就是数字签名的一种应用，是网络数字社会中最重要的一种基础设施，这种技术的重要性，就象电力基础设施对于工业的作用一样。

为了保证数字签名的安全和速度，在签名前，必须对所签的内容进行HASH计算，求出一个HASH值M（HASH值有时又叫文摘值或杂凑值），然后利用保密密钥（私钥）对文摘结果作变换得到签名值；验证签名时，先对内容作HASH计算，然后再用公开的参数(公钥)作用于签名值，再将得到的结果与上述HASH结果进行比较，若相等，就表示签名正确，否则为验证不通过。

然而，当一用户A有了自己的私钥并公开了公钥时，如果有一个攻击者B重新生成公钥和私钥，并用攻击者B的公钥替换了A用户公开的公钥，此时A用户的朋友发给A用户的信息就只有攻击者B知道了，因为A用户已无法知道由攻击者B假冒的公钥所对应的私钥。这时就需要由一个CA（Certificate Authority: 证书权威或证书机构），来证明哪个公钥是A用户的，或者证明哪个公钥不是A用户的。

CA拥有一个比一般用户私钥更长的私钥，也就是说，CA的私钥更安全，而CA的公钥可以通过各种方式广为人知，这样各个用户就可以验证什么是CA所签发的。当一个用户将其身份与其公钥绑定在一起，并由CA签名后，就得到了一个电子证书。这个证书能够证明该公钥拥有者的身份，且任何人都可以验证而任何人却都不能假冒。

目前，电子证书已成为PKI基础设施中最为关键的部分，CA是PKI中最为

重要的基本单元。利用电子证书，就可以有效解决网络上的保密性、完整性以及不可否认性等安全问题。

基于 PKI 的安全性，最终都会归结到对 CA 私钥的保护，一旦 CA 的私钥泄露，由 CA 所颁发的所有证书就都必须作废，整个由 CA 管辖的网络安全就会变得毫无安全可言。随着网络上各种攻击手段的不断增加，系统漏洞不断被发现，因此如何保证既能在线提供数字签名服务又能保证安全，已成为现代网络安全研究的一个重要课题。

对 RSA 算法来说，秘密的密钥一般记为 d，是一个可以长达 2048bits 的整数（一般认为，1024bits 长度的私钥已经能够保证安全，但对 CA 来说，采用 2048bits 或以上则是必须的）。在 RSA 算法中，用到了对一个数 N 的模幂计算，即求 $M^d \bmod N$，该计算是完成数字签名所需要的，当所需公开的参数（公钥）公开后，保护数字 d 就是保护了私钥。

安全的数字签名所追求的目标是既完成签名又不能泄露私钥，针对该课题国际上已有许多理论方面的研究，其中不少理论与方法因难度太大而无法实现。另一方面，国际上目前开发的重点也是如何实现相互兼容的数字签名产品，而在入侵容忍的安全签名方面则开发的非常少，工业界也没有推出相关的产品。所谓入侵容忍(Intrusion Tolerance)是相对于从检测入侵出发来保证 CA 安全的技术而言的，即在系统的部分部件被攻击或被占领后，仍不会暴露 CA 系统的机密信息。

综上所述，PKI 是基于公开密钥密码算法的，在 PKI 系统中，CA 又是一个域中的信任中心，网络上设备或个人间的通信与验证都依赖于 CA 颁发的数字证书（签名）。数字证书是一个将公开密钥与个人身份绑在一起、用 CA 的私钥签名后得到的数据。在一方要验证通信对方的身份时，可以通过两个步骤完成，首先验证对方的证书签名是否正确（签名只能由 CA 的私钥完成，只有 CA 才能够进行签名）；再验证对方是否拥有与证书中公钥相对应的私钥，如果两项都能成立，则对方的身份就能被确定，是可以信任的。

因此CA的私钥是CA安全的核心，保护其不泄露是整个CA域安全的基础。一般来说，CA必须是一个在线的网络设备，特别是直接面向用户以便提供相应证书服务的 CA，则其遭遇网络攻击就不可避免，当一个黑客成功攻击一台 CA 时，攻击者就可能获得它的内部资源，从而找到CA的私钥，会给PKI系统造成致命打击，同时在PKI系统的内部雇员完全控制一台CA设备的情况下，为预防起见，也应该确保其没有得到CA的私钥。

下面仅从将签名等价于实现公式 $M^d \bmod N$（其中 d 为应该保密的私钥）方面，说明现有的几种安全数字签名方法。

参见图 1，图中示出美国 Stanford 大学的 ITTC 项目方案的原型系统框图。该系统通过门槛密码学技术实现系统入侵容忍，系统包括签名方（Clients）、服务器方（Servers）和管理方（Admin）。签名方的 Web Server 是指需要签名的Web服务器，CA是证书权威或证书机构；服务器方包括多个分享服务器(Share Server，也可称分享计算器或分享运算器)，如图中的分享服务器 1 至 3，负责进行安全的数字签名；管理方(Admin)是一个可选的用于管理各分享服务器的设备。该方案的特点是：结构简单和具有很好的安全性，系统中采用了成单层结构的多个分享服务器。

该系统实现安全性的原理是：先将私钥 d 分解成 t 个数的和：$d=d_1+d_2+\ldots+d_t$；再将各数（d_i）分配到各个分享服务器中。当需要签名时，客户机（Web 或 CA）将需要的签名信息－HASH 结果 M 值发给 t 个分享服务器；各分享服务器再将计算结果 $M_i = M^{d_i}$ 送回客户机（Web 或 CA）；由客户机（Web 或 CA）进行乘法计算：

$$S=\prod_{i=1}^{t} M^{d_i} = M^{\sum_{i=1}^{t} d_i} = M^d$$

就得到了需要的结果。

由于不能实现冗余，可进一步采用多组方程来产生冗余配置，即随机找到若干组数，如：

第一组：$d=d_{11}+d_{12}+\ldots+d_{1t}$；

第二组：$d=d_{21}+d_{22}+\ldots+d_{2t}$；

……，

然后将多组数（d_{ij}）分到不同的分享服务器中，每个分享服务器得到多个d_{ij}，但只得到同组数据中的一个数据（如分享服务器 1 得到一个第一组数据 d_{11} 和另一个第二组数据 d_{23}）。如，对于有 4 个分享服务器的情况，在 t=3 时，可以按下表进行分配：

分享服务器 1	分享服务器 2	分享服务器 3	分享服务器 4
d_{11}	d_{12}	d_{13}	d_{13}
d_{23}	d_{21}	d_{22}	d_{23}

当客户机（Web 或 CA）需要计算时，由客户机（Web 或 CA）选定 t 个完好的分享服务器，然后告诉这些分享服务器使用第几组数据（参数），然后，分享服务器就可以计算相应的签名了。

该技术方案的优点是明显的，其不足之处有如下几点：

1. 密钥的分发和管理比较困难，当增加一个分享服务器时，必须对每一个在线的分享服务器分配密钥数据，同时还要让客户机也知道这个增加；

2. 当分享服务器很多时，会迅速增加分享服务器的密钥存储量，若设分享服务器的总数量为 k，要求完好的分享服务器的总数量为 t，则每个分享服务器存储密钥的数量至少为C_k^{t-1}个（C 表示组合），当 k=10, t=3 时，每个分享服务器存储密钥的数量就为 45 个；

3. 存在必须先同步的问题，在进行计算前，客户机必须先选定 t 个分享服务器，当 t 个分享服务器选择完成后，还必须找到与这 t 个分享服务器匹配的数据组并通知他们，当其中有一个服务器被破坏时必须重复上述选择分享服务器的整个过程；

4. 让客户机（CA 或 Web 服务器）记住每次分享服务器的改变是困难的，不容易进行管理与扩展，当客户机在线时，扩展就更加困难，导致每次改变分享服务器时都必须更新客户机。

IBM Research-Zurich 研究院的 Victor Shoup，在 2000 年的欧洲密码学年会上发表了名称为“*Practical threshold signatures*”的文章，介绍的也是一种理论上的方案。该方案使用 RSA 强素数，其保密的素数 p=2p'+1，q=2q'+1。所有的插值方程都在模 m=p'q'的环中进行。因为 M^{4m} 模 N 等于 1，分开计算时增加一次平方，由合成器（Combiner）再分别对各个结果平方一次，从而得到 $M^{4\Delta(gm+d)}$。其中 $\Delta=(k!)^2$，而 g 为整数。该方案中 c_i 是由合成器（Combiner）完成的，从而消除了计算前的同步问题，但带来了合成器的计算难度，并使计算性能下降。如合成器必须计算：

$$W = y_{i_1}^{2\lambda c_{i_1}} y_{i_2}^{2\lambda c_{i_2}} \ldots y_{i_t}^{2\lambda c_{i_t}}$$

其中 y_i 为各分享服务器的计算结果，$\lambda=k!$。

可以看出该方案的特点及存在的不足是：

1. 仍是一种由各分享服务器组成的单层式分享结构，其合成器不存储任何秘密，可以由任何设备完成合成工作；

2. 合成器的计算量接近或相当于 t 次签名，这样的计算量远远大于分享服务器的计算量，即使能实现该算法，也是不能用于实际签名的；

3. 要求使用强素数，会对某些应用带来限制；

4. 仅从理论上说明当增加或删除分享服务器时不会影响其他设备，但只提供了数学公式，没有任何实现上和系统结构上的说明。

纽约 CertCo 公司的 Yair Frankel 等人提出了又一种方案，但没有提供系统实现的框架和细节。在他们的方案中，提出让多项式的系数 a_i 在 $\{0, L, \ldots, 2L^3N^{2+e}t\}$ 中，其中 L=k!。并取 x_i 属于 $[1, 2, \ldots, k-1]$。由于所有 $f(x_i)$ 都能够整除以 L，故 b_i 的计算去掉了求逆的操作，可以在整数中进行。该方案可以用一般的 RSA 算法进行而不需要 RSA 的强素数。由于其参数的选取大大受限，带来了算法原理及安全证明的复杂性。当分享服务器计算 b_i 时，因为 b_i 依赖 x_i 的选取，也存在着同步问题。

从其数学方案的描述中可以看出，该方案具有以下缺点：

1. 采用平等的秘密分享，即分享为单层式结构；

2. 参数的选取是受限的，证明其安全性十分复杂，增加了出现漏洞的可能性；

3. 存在着需要同步的问题，而如果去掉同步，则又会大大增加合成器的计算量。

以上 IBM Research-Zurich 研究院的方案及纽约 CertCo 公司的方案，都是基于 Shamir 秘密共享的方案，使用 LaGrange 插值方程。在原 Shamir 秘密共享方案中，任意取 t 个密钥就能够生成秘密密钥。但在原 Shamir 方案中必须先恢复秘密密钥，这是任何方案都不希望的。因为安全签名方案必须首先保证在任何一台设备内都不能恢复出秘密密钥。

这种基于 Shamir 秘密共享方案的基本原理是：

给出一个多项式 $f(x)=\sum_{i=0}^{t-1}a_i x_i$，利用 LaGrange 插值公式，有：

$$f(x)=\sum_{i=1}^{t}(f(x_i)\prod_{j=1,i}^{j\neq i}\frac{x-x_j}{x_i-x_j}) \qquad (1)$$

任选 t 个 x_i 和 $f(x_i)$，可以得到：

$$a_0=f(0)=\sum_{i=1}^{t}(f(x_i)\prod_{j=1,i}^{j\neq i}\frac{-x_j}{x_i-x_j}) \qquad (2)$$

可以设置 a_0 为秘密密钥 d，此时，对一个 HASH 值 M 的签名计算为：

$$M^d=M^{f(0)}=\prod_{i=1}^{t}M^{f(x_i)\prod_{j=1,i}^{j\neq i}\frac{-x_j}{x_i-x_j}}=\prod_{i=1}^{t}M^{b_i} \qquad (3)$$

其中 $b_i=f(x_i)*c_i=f(x_i)\prod_{j=1,i}^{j\neq i}\frac{-x_j}{x_i-x_j}$ (4)

这样就可以将秘密密钥 d 分到 k 个分享服务器中去($k\geq t$)。每个分享服务器计算 M^{b_i}，然后由一个合成器将各个分享服务器的计算结果乘起来，就得到 M^d，而任何分享服务器都不会泄露秘密密钥 d。由于式（4）的 b_i 中有除法计算，这很容易让人想到找一个域或环 Z_v 来进行计算。其中，必须满足的条件是：v 为素数，或者保证所有由 x_i 构成的 t 阶（Vandermonde）矩阵的行列式的值与 v

互素。

在一般情况下，分开计算M^{b_i}后带来的结果就是要做乘法：$\prod_{i=1}^{t} M^{b_i} = M^{d+wv}$。如何去掉 v 的影响，许多人想到了用Φ(N)，因为$M^{\Phi(N)}=1$。但选择 v=Φ(N)会使x_i的选取大大受限于上述条件，并且知道了某元素 o 与其对Φ(N)的逆o^{-1}，就可以求出Φ(N)，显然这是不安全的。

因此，从理论上可以看出上述方案都有明显的弱点，距离实际应用还有很多问题需要解决。

发明内容

本发明的目的是设计一种数字签名系统与安全签名方法，是一种入侵容忍的安全签名系统与方法，在现有实现 CA 安全的基本原理基础上，消除预先同步的问题，能满足在线 CA 的安全要求，当系统的部分部件被攻击或被占领，或当系统的部分关键部件进行合谋时也能确保系统机密不泄露。

本发明的安全数字签名方法，CA 以 RSA 算法为基础。该方法至少应该满足以下条件：

1. 入侵者虽然攻击或占有了系统中的若干部件，或某些关键部件合谋起来进行攻击，也不能得到私钥；

2. 系统很容易扩充，当需要增加一个分享运算器时，不影响整个系统的工作；

3. 运行时的管理简单，管理包括增加，撤除，修改服务硬件与软件等，在有系统管理与配置活动时，不能影响系统运行；

4. 一个或多个设备损坏时不影响正常的服务；

5. 当一个分享运算器损坏时，整个系统的运行效率不会降低太多，即任务分配单元不需要知道任务执行者的任何信息；

6. 开始计算时，任何设备都不需要知道他的合作者是谁，故不需要一个合作群的指定机构(预先同步)；

7. 该系统的算法和原理都应该非常简单;

8. 设计完成后，整个系统的运行效率应该与常规系统保持在同一个档次。

实现本发明目的的技术方案是这样的：一种安全的数字签名系统，用于得到数字签名，其特征在于：

包括至少一个在线的任务分配器、k个在线的秘密分享运算器、m个在线的秘密分享合成器和离线的子密钥分发器；离线的子密钥分发器在系统初始化或进行系统配置时与 k 个秘密分享运算器及 m 个秘密分享合成器分别连接进行子密钥发放，包括：对应 k 个秘密分享运算器生成随机数，并作为第一子密钥分发到 k 个秘密分享运算器中，和对应 m 个秘密分享合成器计算第二子密钥并预存在 m 个秘密分享合成器中；在线的任务分配器通过第一广播信道与 k 个秘密分享运算器连接，在计算数字签名处理过程中将需要签名的 M 值送 k 个秘密分享运算器，每个秘密分享运算器根据 M 值和第一子密钥进行计算，并将计算结果和 M 值通过第二广播信道送 m 个秘密分享合成器， 每个秘密分享合成器根据接收结果、M 值及预存的第二子密钥计算得到数字签名，k、m 为正整数。

更佳的是还包括一单独设置的输出接口设备，通过第三广播信道与 m 个秘密分享合成器连接。

更佳的是在所述的在线的任务分配器中设置一输出接口设备，通过所述的第一广播信道与所述的 m 个秘密分享合成器连接。

所述的至少一个在线的任务分配器、k 个在线的秘密分享运算器、m 个在线的秘密分享合成器、离线的子密钥分发器均采用普通计算机或服务器。

所述的第一广播信道、第二广播信道及第三广播信道是物理相连的一个信道或完全不相连的独立信道。

实现本发明目的的技术方案还是这样的：一种安全的数字签名方法，包括子密钥的发放和计算数字签名，其特征在于：

所述的子密钥的发放包括以下处理步骤：

A. 设置一个安全数字签名系统，包括在线的任务分配器、在线的 k 个秘密分享运算器、在线的 m 个秘密分享合成器、广播信道和离线的子密钥分发器，k、m 为正整数；

B. 离线的子密钥分发器将保存的数字签名私钥 d 表示成由 t 个第一子密钥 d_j 及一个第二子密钥 c_i 的和，d、t、c、j、i 均为正整数并让 t<k，j 是第一子密钥 d_j 变量的序号；

C. 离线的子密钥分发器将 k 个随机数作为第一子密钥 d_j 对应分发到 k 个秘密分享运算器中，并根据步骤 B 中的和式通过相减获得一组第二子密钥 c_i 及其方程组合表示，将全部第二子密钥 c_i 及其方程组合表示放在一个大组中；

D. 离线的子密钥分发器按合成器安全条件对该大组中的全部第二子密钥 c_i 及其方程组合表示进行穷搜，获得 m 个分组的第二子密钥 c_i 及其方程组合表示：

E 离线的子密钥分发器将 m 个分组的第二子密钥 c_i 及其方程组合表示对应送到 m 个秘密分享合成器中预存；

所述的计算数字签名包括以下处理步骤：

F. 在线的任务分配器将需要签名的 HASH 值 M 通过广播数据包经第一广播信道送往 k 个秘密分享运算器；

G. k 个秘密分享运算器中的 t 个或 t 个以上的秘密分享运算器根据接收的 M 值计算升幂 M^{d_j}，并将本秘密分享运算器的代号 j、需要签名的 HASH 值 M 及计算结果 M^{d_j} 通过广播数据包经第二广播信道送往 m 个秘密分享合成器；

H. m 个秘密分享合成器将接收结果与预存分组的第二子密钥 c_i 的方程组合表示进行比较，找到可以匹配的方程组合表示并得到相应的第二子密钥 c_i，再将与组合匹配的 t 个秘密分享运算器的升幂运算结果相乘得到结果 R，根据找到的第二子密钥 c_i 求出 M^{c_i}，最后将 M^{c_i} 与 R 相乘得到数字签名

$S=\dot{M}^{d}$。

更佳的是，所述的步骤 A 中还包括给在线的任务分配器任意给定一个代号、给 k 个秘密分享运算器分别给定不相同的代号、给 m 个秘密分享合成器分别给定不相同的代号和设定 t 的初始值。

更佳的是，所述的步骤 C，进一步包括:

c1.由离线的子密钥分发器取 k 个小于 d/t 的随机数作为第一子密钥 d_j;

c2.由离线的子密钥分发器通过管理许可的方式将 k 个第一子密钥 d_j 对应送到 k 个秘密分享运算器中;

c3.由离线的子密钥分发器从 k 个第一子密钥(d_1、d_2、d_3、…、d_k)中取出 t 个 d_j，按组合算式形成 $n=C_k^t$ 种组合表示，并根据所述的和式计算所有组合表示所对应的第二子密钥 c_i，(i=1、2、…、n)，构成所述的方程组合表示大组。

更佳的是，所述的步骤 D 进一步包括:

d1.对所述的大组中的组合随机设定一个顺序，并随机设定一个空的分组 B;

d2.从大组中顺序取出一个第二子密钥 c_i 及其方程组合表示，记为组合表示 A，判断 A 放入 B 中是否满足合成器安全条件，如果满足就放入分组 B 中并从大组中删除该组合及对应的第二子密钥，如果不满足合成器安全条件，就将该组合及对应的第二子密钥仍旧保持在大组中;

d3.针对大组中所有的组合重复执行步骤 d2;

d4.如果大组中还有组合表示，确定第二个空的分组为 B，并重复执行步骤 d2、d3，直至大组中已没有任何第二子密钥 c_i 及其方程组合表示时为止;

d5.统计形成的不空的分组个数 m，是所述的 m 个分组。

所述的判断是否满足合成器安全条件，进一步包括: 将组合表示 A 中变量

的序号与分组 B 中的组合表示 q_s 中变量的序号进行比较，如果组合表示 A 中有且组合表示 q_s 中没有的变量的序号个数大于 t/2，该组合表示 A 对组合表示 q_s 是满足安全条件的；如果组合表示 A 对分组 B 中的每一个组合都满足安全条件，则组合表示 A 放入分组 B 中是满足合成器安全条件的。

更佳的是，所述的步骤 E，由离线的子密钥分发器通过管理允许的方式将所述 m 个分组的第二子密钥 c_i 及其对应的方程组合表示对应分送到所述的 m 个秘密分享合成器中。

更佳的是，所述的离线的子密钥分发器在执行完步骤 A、B、C、D、E 完成子密钥的分发后，处于物理隔离状态或处于关机状态。

更佳的是，所述的步骤 F 进一步包括：

f1. 由在线的任务分配器接收数字签名任务，并进行安全检查与核对；

f2.由在线的任务分配器为该任务确定一个在预定的时间内对任务分配器是唯一的任务号；

f3.由在线的任务分配器将其代号及任务号随需要签名的 HASH 值 M 组成所述的广播数据包广播到所述的第一广播信道上；

所述的步骤 G 进一步包括：

g1.接收到广播后的 t 个或 t 个以上的秘密分享运算器向该在线的任务分配器发出已经接收的通知；

g2.t 个或 t 个以上的秘密分享运算器检查任务的唯一性，在确定为新任务时进行所述升幂的计算；

g3.t 个或 t 个以上的秘密分享运算器还将该在线的任务分配器的代号、任务的任务号随本秘密分享运算器的代号 j、需要签名的 HASH 值 M 及计算结果 M^{d_j} 组成所述的广播数据包广播到所述的第二广播信道上；

所述的步骤 H，进一步包括：

h1.接收到广播的秘密分享合成器将具有相同任务分配器代号及任务号的广播数据包放在一组中；

h2.上述秘密分享合成器至少找出 t 个广播数据包，再从中找出与所述预存的方程组合表示匹配的一个方程组合表示，并得到对应的第二子密钥 c_i。

所述的步骤 F、G、H 是顺序执行完成数字签名的计算操作。

更佳的是，所述的步骤 H 后还包括一步骤 I，由在线的秘密分享合成器，将数字签名 $S=M^d$ 与任务分配器代号、任务号组成广播数据包广播到一在线的输出接口设备上，并由该输出接口设备用公开密钥检查签名结果，是正确时结束数字签名，是错误时进行错误处理或告警。

更佳的是，所述步骤 I，是由在线的秘密分享合成器通过一第三广播信道将所述的广播数据包广播到一单独设置的在线的输出接口设备上。

更佳的是，所述步骤 I，是由在线的秘密分享合成器通过所述的第一广播信道将所述的广播数据包广播到设置在任务分配器中的输出接口设备上。

本发明的安全的数字签名系统及方法，基于 RSA 算法，通过一种不平衡的双层结构的秘密分享，克服了背景技术中系统管理与实现上的困难，达到发明的目的。

本发明的方法与系统具有下列特点:

1. 在线的任务分配器无须选定秘密分享运算器和指定密钥就可以广播数字签名任务，在系统更新时可保证不影响在线任务分配器的工作，在某台秘密分享运算器突然损坏时也不会影响广播任务的执行时间（如果指定由某台秘密分享运算器计算，而此台设备最后并没有响应，则必须重新指定设备与密钥并重新开始）;

2. 在增加一台秘密分享运算器时只需给它生成一个子密钥，离线的子密钥分发器会根据新加的秘密分享运算器的序号与已有的秘密分享运算器的序号进行方程组合，计算出相应的第二子密钥 c，然后将这些新增的方程组合表示与计算出的 c 以密钥管理认可的方式添加到秘密分享合成器中去，这种添加是不影响系统工作的;

3. 在撤除一台秘密分享运算器时，只需关闭该设备，为保证效率起见，可

以删除秘密分享合成器中有关该秘密分享计算器序号的组合表示和相应的 c_n;

4. 同时，本发明同其他背景技术一样，也具有入侵容忍的能力，当小于 t 台秘密分享计算器被入侵以后，不会泄露系统秘密 d，而且由于增加了秘密分享合成器的结构，即使所有的秘密分享计算器都被入侵，也不会泄露系统秘密 d（从理论上也可以证明，攻击秘密分享合成器也无法得到系统秘密 d，尽管方程个数很多，但所有方程的系数矩阵的秩小于变量的个数）；

5. 可抵制秘密分享运算器与秘密分享合成器合谋对系统的攻击，即当一台秘密分享运算器与一台秘密分享合成器联合攻击系统时也不能求出秘密 d.

附图说明

图 1 是 Stanford 大学研究的 ITTC 项目方案中关于 CA 系统的原型结构示意图。

图 2 是本发明的一种安全数字签名系统的结构示意图。

图 3 是本发明的另一种安全数字签名系统的结构示意图。

图 4 是一种秘密分享运算器的实施流程框图。

图 5 是一种秘密分享合成器的实施流程框图。

具体实施方式

参见图 2，图中示意出一种安全的数字签名系统结构，该系统结构采用 RSA 算法为基本算法。

该结构中包含一台离线的子密钥分发机（器，以下同）21，至少一台在线的任务分配机 22、k 台在线的秘密分享运算器 23、m 台在线的秘密分享合成器 24 和一台独立设置的在线的输出接口设备 25。这些设备都可以由不同的普通计算机或服务器来担任。在线的任务分配机 22 通过广播信道 B1（如 UDP 协议信道）与 k 台子秘密分享运算器 23 连接，k 台子秘密分享运算器 23 通过广播信道 B2 与 m 台秘密分享合成器 24 连接，m 台秘密分享合成器 24 通过广播信道 B3 与输出接口设备 25 连接，离线的子密钥分发机在系统初始化或进行系统配置时分别连接 k 台子秘密分享运算器 23 及 m 台秘密分享合成器 24（如图中虚线所

示）。

离线的子密钥分发器 21 保存秘密 d，不与其他系统有网络连接。广播信道 B1、B2、B3 可以是物理相连的一个广播信道或完全不相连、彼此独立的广播信道。

实现整个系统结构的基础原理是将一个大整数表示成若干个整数的和，采用如 $d=d_1+d_2+d_3+\ldots+d_t+c_j$ 的表达式，用 d_j 表示方程中 d_1 至 d_t 中的任意一项，将数字签名中的私钥设为方程中的 d，d_j 的个数有 t 个，d、d_1、d_2、d_3...d_t、c_j 均为正整数，d_j 采用随机数以实现简单管理。该方程表达式不同于背景技术中方程表达式 $d=d_1+d_2+d_3+\ldots+d_t$ 的地方是增加了一个 c_j 项（i=1, 2, ..., n），从而形成一种新的安全的且易于管理的系统结构。

该系统结构对秘密 d 的分享采用由两层部件共同作用完成，一层部件是由简单的秘密分享运算器 23 组成的，另一层部件是由秘密分享合成器 24 组成的，在秘密分享运算器 23 中分别保存各 d_j 项，在秘密分享合成器中保存 c_j 项，从而形成两层式的秘密分享结构，这两层设备中都存有子密钥，分别为第一子密钥 d_j 与第二子密钥 c_j。该两层式的分享结构还表现在：只有第一层的秘密分享运算器完成计算后，第二层的秘密分享合成器才能开始工作；由于秘密分享合成器中也保存有子密钥 c_j，因而秘密分享合成器是不可能被其他设备取代的。

系统首先完成子密钥的分发，采用的操作过程如下：

离线的子密钥分发器 21 为每个秘密分享运算器 23 产生一个随机的 d_j，对有 k 台秘密运算分享器 23 的系统来说，就会有 k 个第一子密钥 d_j, j=1, 2, 3, 4, ..., k。将这些第一子密钥通过密钥管理认可的方式送至对应的秘密分享运算器 23。

需预先设计合适的 t，t 的含义就是当 t-1 台秘密分享运算器 23 被入侵后不会影响系统的安全性。应该保证 t<k，以保证系统可以从 k 个秘密分享运算器 23 中取到 t 个结果完成运算。

离线的子密钥分发器 21 从 k 个子密钥中取出 t 个子密钥，根据方程

$d=d_1+d_2+d_3+\ldots+d_t+c_i$再通过作减法可以求出 c_i。由于从 k 个中取 t 个共有C_k^t种取法，故可以计算出C_k^t个方程的 c_i。这里，$n=C_k^t$代表组合，如$C_5^3=10$，就有 10 个 c_i值和 10 个方程组合表示，每一个方程组合表示是所对应的 t 个第一子密钥 d_j下标标号（或者说是变量的序号）的组合，这样的一个方程所对应的密钥 d_j 的标号为密钥组合。显然，一个方程组合内的不同的第一子密钥是在不同的秘密分享运算器内，方程组合表示只与第一子密钥 d_j下标标号 j（或者说是变量的序号）有关，不泄露任何子密钥的信息。

d_j的比特数远小于 c_i比特数的 1/4 ，如当 d 为 2048bits 的数时，c_i为 2048bits 的数，d_j为 500bits 的数或更少，以保证秘密分享运算器 23 的运算速度,从而提高整个数字签名系统的运算速度。

离线的子密钥分发器 21 对所有的方程组合表示以及对应的 c_i值，按合成器安全条件通过穷搜进行分组，每一分组内的方程组合表示是有限的，再按分组的个数（m）对应设置秘密分享合成器（m 个），来对应存储这些分组的方程组合表示以及对应的 c_i值。以图中设置 5 台秘密分享运算器 23(k=5)和设 t = 3 为例说明。

如将十个方程分成如下 5 组:

第一组

$c_1 = d - d_1-d_2-d_3$　　方程组合表示 (1, 2, 3)

$c_6 = d - d_1-d_4-d_5$　　方程组合表示 (1, 4, 5)

第二组

$c_{10} = d - d_3-d_4-d_5$　　方程组合表示 (3, 4, 5)

$c_3 = d - d_1-d_2-d_5$　　方程组合表示 (1, 2, 5)

第三组

$c_2 = d - d_1-d_2-d_4$　　方程组合表示 (1, 2, 4)

$c_8 = d - d_2-d_3-d_5$　　方程组合表示 (2, 3, 5)

第四组

$c_4 = d - d_1 - d_3 - d_4$ 方程组合表示 (1, 3, 4)

$c_9 = d - d_2 - d_4 - d_5$ 方程组合表示 (2, 4, 5)

第五组

$c_7 = d - d_2 - d_3 - d_5$ 方程组合表示 (2, 3, 4)

$c_5 = d - d_1 - d_3 - d_5$ 方程组合表示 (1, 3, 5)

则选用 5 台秘密分享合成器 24，分别存储这五组参数。

如何分组是实施本发明方案的关键问题。从秘密分享合成器 24 看来，其 c_i 是已知的，其他都是未知的变量。合成器安全条件是根据其方程和系统安全性要求提出的，即：一个秘密分享合成器内所含方程经线性组合得到的新的方程，其变量的个数大于 t。

满足该条件，就可以避免合成器与分享运算器对系统的合谋攻击。但是，该条件过于复杂，无法用程序或流程实现。因此为了实现安全的合成，必须有一个可行的算法。

利用方程的特殊性，可使安全条件简化为：一个合成器内由任意两个方程的线性组合得到的方程，其变量的个数大于 t。

对于该简化的安全条件，其实施包括以下步骤：

步骤 1，根据 $d=d_j+\ldots+d_t+c_i$ 的表示式及组合式 C_k^t，有全部 10 种 c_i 的方程组合表示，分别为：

$c_1 = d - d_1 - d_2 - d_3$ 方程组合表示 (1, 2, 3)

$c_2 = d - d_1 - d_2 - d_4$ 方程组合表示 (1, 2, 4)

$c_3 = d - d_1 - d_2 - d_5$ 方程组合表示 (1, 2, 5)

$c_4 = d - d_1 - d_3 - d_4$ 方程组合表示 (1, 3, 4)

$c_5 = d - d_1 - d_3 - d_5$ 方程组合表示 (1, 3, 5)

$c_6 = d - d_1 - d_4 - d_5$ 方程组合表示 (1, 4, 5)

$c_7 = d - d_2 - d_3 - d_5$ 方程组合表示 (2, 3, 4)

$c_8 = d - d_2 - d_3 - d_5$ 方程组合表示 (2, 3, 5)

$c_9 = d - d_2-d_4-d_5$ 方程组合表示 (2, 4, 5)

$c_{10} = d - d_3-d_4-d_5$ 方程组合表示 (3, 4, 5)

将他们全部放入一大组中，并指定以上顺序；

步骤 2，从该大组中顺序取出一个组合，如方程组合表示(1, 2, 3)，$c_1 = d - d_1-d_2-d_3$，并将其放入第一个分组中，该分组记为 B；

步骤 3，再从大组中顺序取出一个组合，如方程组合表示(1, 2, 4)，$c_2 = d - d_1-d_2-d_4$，记为组合表示 A，并将分组 B 中组合表示均记为 q_s；

步骤 4，将该组合表示 A 与第一分组 B 中的已有组合表示进行比较，判断比较结果是否满足安全条件，包括：

步骤 41，将组合表示 A 中变量的序号与分组中各组合表示 q_s 中变量的序号进行比较，如果组合表示 A 中有的且组合表示 q_s 中没有的变量的序号个数大于 3/2（t/2），该组合表示 q_s 就算满足安全条件，如组合表示（1，2，3）与（3，4，5）比较，其不同序号的个数为 2，在 t=3 时组合表示（3，4，5）就满足安全条件，显然方程组合表示(1, 2, 3)，$c_1 = d - d_1-d_2-d_3$ 与组合表示 A (1, 3, 5)，$c_5 = d - d_1-d_3-d_5$，相比较时，其不同序号的个数为 1，不能满足安全条件；将对 B 中所有组合满足安全条件的组合表示 A 放在分组 B 中，将不能满足安全条件的分组表示仍保持在大组中；

步骤 42，对大组中的所有组合表示逐一比较，让满足安全条件的组合表示继续放入分组 B 中，操作的结果是大组中已没有一个组合表示能符合合成器安全条件，即此时分组 B 中能满足合成器安全条件的所有组合表示只是（1，2，3）与（3，4，5）；

步骤 5，然后确定第二个分组，并重复步骤 2、3、4，直至大组中已没有组合表示时为止，而形成如前所述的五个分组。

该过程执行完成后，将所有不空分组的组合表示取出，每个分组中的组合表示通过管理允许的方式对应预存入一个秘密分享合成器中。

根据随机选取组合表示的不同，所形成的 m 个分组的组合表示也会是不同

的，如：

第一组

$c_1 = d - d_1 - d_2 - d_3$ 方程组合表示 (1, 2, 3)

$c_9 = d - d_2 - d_4 - d_5$ 方程组合表示 (2, 4, 5)

第二组

$c_2 = d - d_1 - d_2 - d_4$ 方程组合表示 (1, 2, 4)

$c_5 = d - d_1 - d_3 - d_5$ 方程组合表示 (1, 3, 5)

第三组

$c_3 = d - d_1 - d_2 - d_5$ 方程组合表示 (1, 2, 5)

$c_4 = d - d_1 - d_3 - d_4$ 方程组合表示 (1, 3, 4)

第四组

$c_6 = d - d_1 - d_4 - d_5$ 方程组合表示 (1, 4, 5)

$c_7 = d - d_2 - d_3 - d_5$ 方程组合表示 (2, 3, 4)

第五组

$c_8 = d - d_2 - d_3 - d_5$ 方程组合表示 (2, 3, 5)

第六组

$c_{10} = d - d_3 - d_4 - d_5$ 方程组合表示 (3, 4, 5)

这种分组方法的最后结果就是要设置六个秘密分享合成器，来分别预存这六组 c_i 及其方程组合表示。

因此，每个秘密分享合成器 24 并不存储针对所有子密钥 d_j 的所有组合，但所有秘密分享合成器 24 存储内容和的结果能保证包含针对秘密分享运算器的所有 d_j 组合。

系统进行计算数字签名的操作过程，运算时，秘密分享运算器 23 针对其拥有的子密钥计算升幂，秘密分享合成器寻找匹配组合，计算并合成结果。

计算数字签名时：任务分配器将需要签名的 HASH 值 M，经广播信道 B1 通

过广播方式送往 k 个秘密分享运算器 23，只要有超过 t 个正确的秘密分享运算器（指未被攻击的秘密分享运算器）收到数据就可以保证得到计算结果；

其中的第 j 个秘密分享运算器在收到数据后先计算升幂M^{d_j}，每一个秘密分享运算器针对其拥有的第一子密钥计算升幂，并将自己的代号 j、需要签名的 HASH 值 M 及计算结果M^{d_j}经广播信道 B2 通过广播方式送到秘密分享合成器 24；

秘密分享合成器 24 对接收的数据包采用遍历方法与自己预存的方程组合表示进行比较，找到可以匹配的 t 个组合表示并得到相应的子密钥 c_i，再将经组合匹配后的几个升幂结果相乘得到结果 R，再根据找到的 c_i 求出 M^{c_i}，最后将 M^{c_i} 与 R 相乘就得到应该得到的数字签名 $S=M^d$；

秘密分享合成器 24 将数字签名结果 S 送到输出接口设备 25，输出接口设备 25 根据公开密钥检查签名结果的正确性。

以上计算流程中，所有广播的数据包内还应包括：任务分配器的代号，任务分配器分配的任务代号等。以保证系统能够区分不同的任务，支持多任务并行操作。

参见图 3，是一个安全数字签名系统的实施例结构，使用一台离线的子密钥分发器 31，5 台秘密分享运算器 33（k=5），五台秘密分享合成器 34，一台在线的任务分配器 32 并兼做输出接口设备，在线的任务分配器 32 通过广播信道 B1 与 5 台秘密分享运算器 33 连接，5 台秘密分享运算器 33 通过广播信道 B2 与五台秘密分享合成器 34 连接，五台秘密分享合成器 34 还通过广播信道 B1 与在线的任务分配器 32 连接，一台离线的子密钥分发器 31 在系统初始化或系统配置时分别连接 5 台秘密分享运算器 33 及两台秘密分享合成器 34。

先进行子密钥的发放操作：

给定在线的任务分配器 32 任意一个代号，如 22，给定各秘密分享计算器 33 一个代号，如 1，2，3，4，5，给定各秘密分享合成器 34 一个代号，如 1，2，3，4，5，系统设定初始值 t=3，由离线的子密钥分发器 31 掌握秘密密钥 d；

离线的子密钥分发器 31 任意选定 5 个小于 d/3（d/t）的随机数

d_1, d_2, d_3, d_4, d_5，通过某种管理许可的方式将 d_1 送到 1 号秘密分享计算器，将 d_2 送到 2 号秘密分享计算器，将 d_3 送到 3 号秘密分享计算器，将 d_4 送到 4 号秘密分享计算器，将 d_5 送到 5 号秘密分享计算器；

根据 $d=d_j+\ldots+d_t+c_i$ 的表示式及组合式 C_k^t，组成一个大组，含有 10 种 c_i 的方程组合表示及 10 个 c_i；

按分享器安全条件通过穷搜获得五个分组及其 c_i 的组合表示；

离线的子密钥分发器 31 将这五个分组的 c_i 值及相应的方程组合表示一起以管理允许的方式对应送到五个秘密分享合成器 34 中，即每个秘密分享合成器 34 中各有二个方程组合表示和二个对应的 c_i 值。

子密钥发放完成后，离线的子密钥分发器 31 就可以关机了。

计算数字签名的工作流程为：

由在线的任务分配器 32 接收必须签名的任务，并做相应的安全检查和核对，然后求需要签名的 HASH 值 M；

由在线的任务分配器 32 为当前的签名确定一个任务号，该任务号应该在一定时间范围内（如两天）对该任务分配器来说是唯一的；

任务分配器将自己的代号（22），该任务的任务号，HASH 值 M 组成一个数据包广播到广播信道 B1 上；

秘密分享计算器 j（秘密分享计算器 33 中的至少 3 个秘密分享计算器，t=3）收到广播后，通知任务分配器 32 已经成功接收；

秘密分享计算器 j 检查任务的唯一性，当该任务为新任务时，计算 M^{d_j}；

秘密分享计算器 j 将自己的代号 j，任务分配器代号 22，任务号，HASH 值 M 和计算结果 M^{d_j} 向广播信道 B2 广播；

秘密分享合成器 34 接收该广播数据包，并将具有相同任务分配器代号和任务号的信息放在同一组中；

秘密分享合成器 34 在一组结果中检查是否有三个以上结果且有 3 个与存储的方程组合表示匹配，如果有，找到相应匹配的方程组合表示，得到对应的 c_i，

根据 c_i 求 M^{c_i}，然后计算 R（是对应的三个与组合表示匹配的结果的乘积），最后将 M^{c_i} 与 R 相乘就得到应该得到的数字签名 $S=M^d$；

秘密分享合成器 34 将任务分配器的代号，任务号和数字签名结果 $S=M^d$ 送到广播信道 B1 上；

任务分配器 32 接收数字签名结果后，用公开密钥检查签名结果是否正确，如果错误，进入错误处理或告警，如果检查合格即结束计算任务。

本实施例中各细化的执行步骤同样适用于图 2 实施例。

本发明的方法能有效地抵制秘密分享合成器和秘密分享计算器联合时对系统的合谋攻击。如上例中，任何一个分享合成器要与至少三台秘密分享运算器合谋才能得到秘密 d。而能有力地抵制分享合成器与分享运算器的合谋攻击。

参见图 4、图 5，分别为图 2、图 3 中秘密分享运算器和秘密分享合成器的实施流程框图，是一种具体的实用流程。其实现是采用多线程技术，即计算机操作系统中使用的多任务并行技术，如图中所示的三个线程：任务处理程序、监听程序和界面程序，来实现一个秘密分享运算器或秘密分享合成器。所有需要计算的任务由一个任务队列来进行管理，而计算所需的系统参数（包括子密钥）也是三个线程都能够存取的。

其中的任务处理程序是专门用于计算的，由于计算是最费时的一项工作，所以单独创建一个线程进行计算，以保证在计算时能够继续保持与用户间的交互，同时能够保持对网络数据的监听从而不丢失网络数据。该线程首先利用事件同步机制进行等候，可以在没有任务的时候节约 CPU 的时间，可以使用象 NT 中的 EVENT 之类的系统事件来进行同步；当任务执行线程被唤醒后，它扫描系统的任务队列，找到应该处理的任务进行处理，包括应该重新发送的数据包的发送。

监听程序是为了保证网络通信的数据不丢失而设置的，单独采用一个线程来监听网络的广播数据，该流程只进行简单的任务处理，如删除一个任务或增加一个任务，当有新的计算任务时，唤醒任务处理线程。

界面程序用于用户界面处理，也单独采用一个线程，以保证在不影响系统运行的情况下修改系统的参数、增加子密钥等。

按本发明技术方案设计的具有安全功能的数字签名设备，可以广泛应用于企事业中，每个设备针对每个企业存储一套相应数据，任务分配器在广播任务时就将企业代号放在数据包中，以区分不同的企业，秘密分享运算器根据企业代号找到相应的密钥并计算，计算结果连同企业代号一起送往合成器。

整个系统增加企业代号的管理以后，系统可以成为多个不同层次的用户、企业的代理，代理他们进行安全的数字签名。而子密钥分发器和任务分配器可以由企业或个人自己控制。在这种结构下，就构成了多个子密钥分发中心、多个任务分配器、多个密秘分享运算器，多个秘密分享合成器的格局，而成为一个完善的安全数字签名服务体系。

本发明的系统结构简单，易于实现，有入侵容忍能力，虽然签一次名的总工作量与一般签名相比是增加了，但是能确保系统安全，通过采用并行计算并使 d_j 的比特数远小于 c_i 的比特数（至少 4 倍），而能使总的签名时间基本与一般签名时间相等，更重要的是能对付秘密分享运算器与秘密分享合成器联合对系统的攻击。

本发明方案的不足之处是：为抵制秘密分享合成器与秘密分享运算器合谋对系统进行攻击，会使秘密分享合成器的数量大幅攀升，如当秘密分享计算器的台数变为 6 时，秘密分享合成器的台数会涨到 8，从而消耗了太多的硬件资源；此外，如此组合秘密分享合成器与秘密分享运算器，会导致任务的不均衡，即一个数字签名任务需要至少 t 个秘密分享运算器计算一次，但却只需要一台秘密分享合成器参与工作，如上述举例中，秘密分享合成器的个数多于秘密分享运算器的个数，这存在着两者工作不均衡的问题。

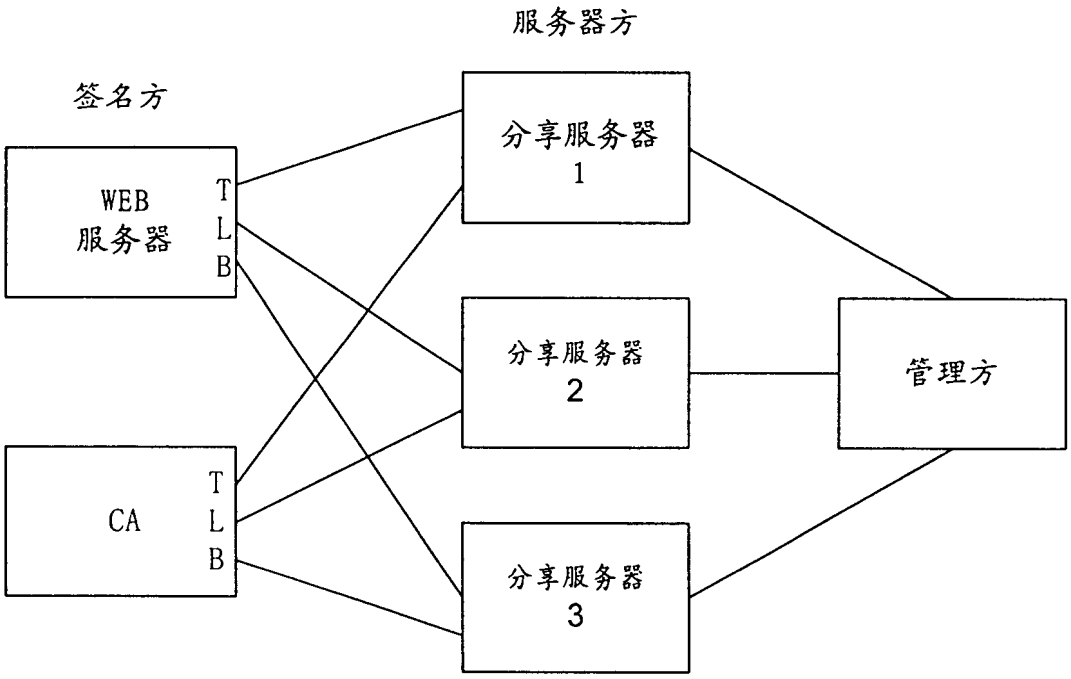


图 1

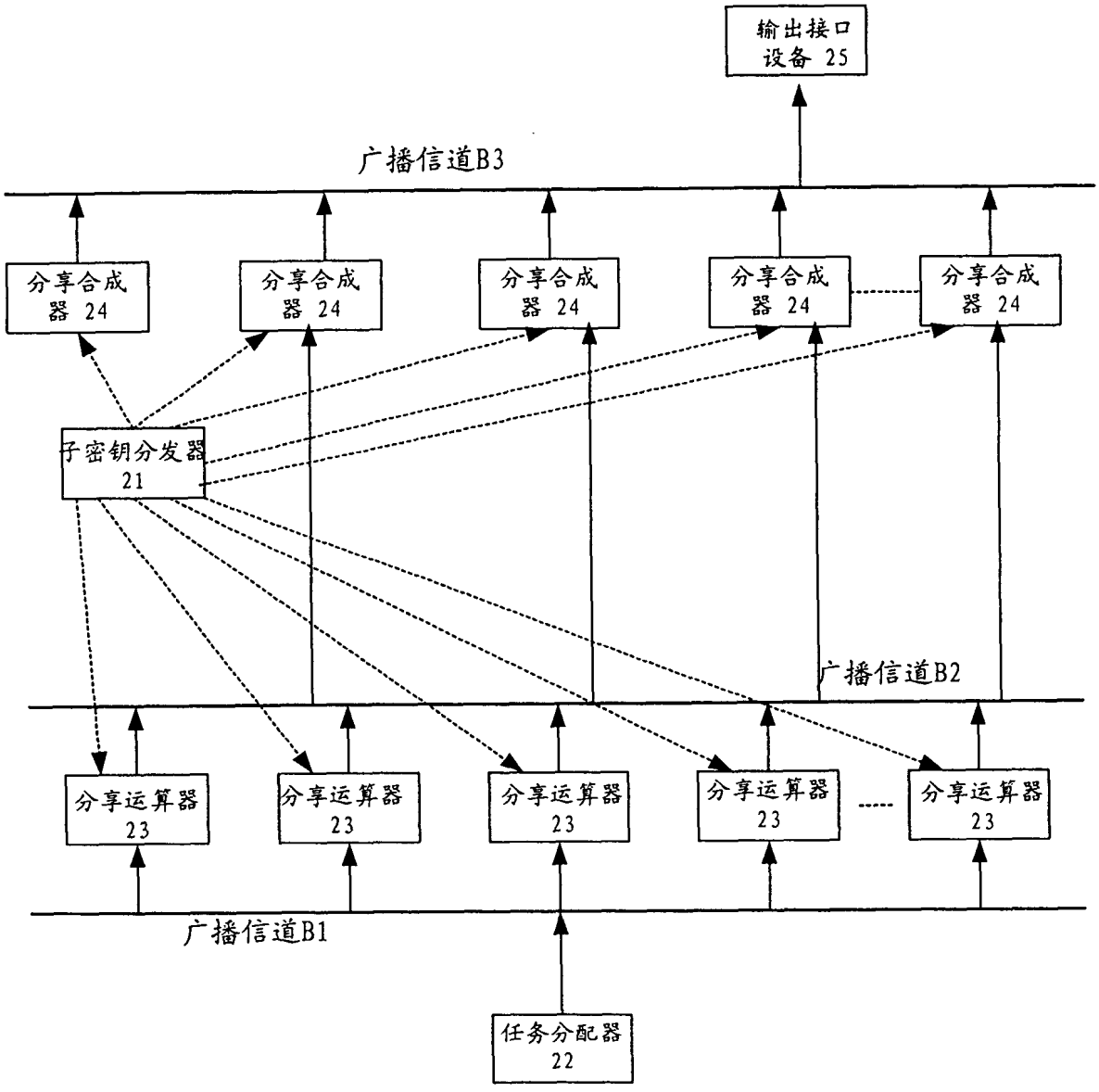

输出接口
设备 25
广播信道B3
分享合成
器 24
分享合成
器 24
分享合成
器 24
分享合成
器 24
分享合成
器 24
子密钥分发器
21
广播信道B2
分享运算器
23
分享运算器
23
分享运算器
23
分享运算器
23
分享运算器
23
广播信道B1
任务分配器
22

图 2

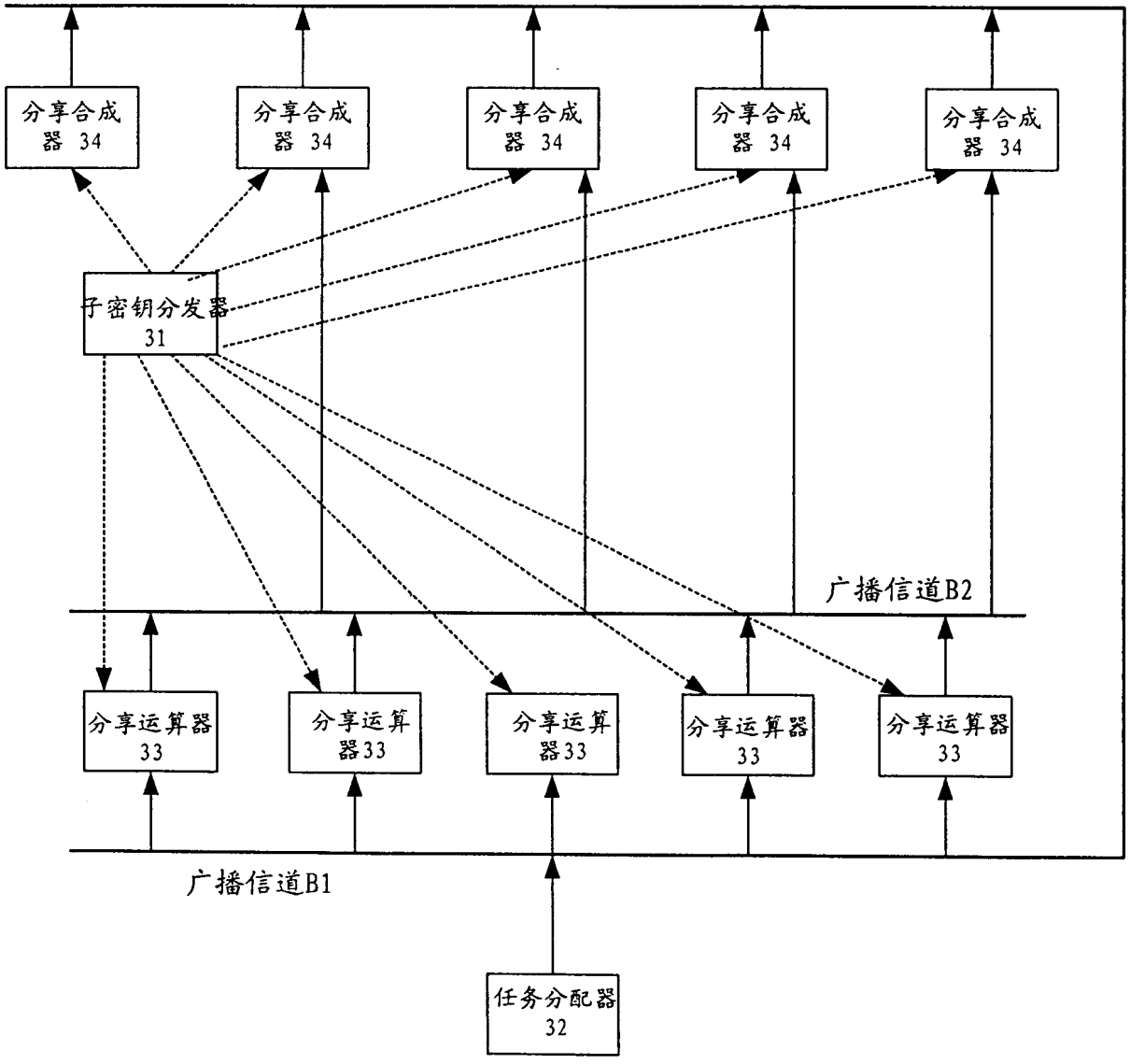

分享合成器 34
分享合成器 34
分享合成器 34
分享合成器 34
分享合成器 34
子密钥分发器 31
广播信道B2
分享运算器 33
分享运算器33
分享运算器33
分享运算器 33
分享运算器 33
广播信道B1
任务分配器 32

图 3

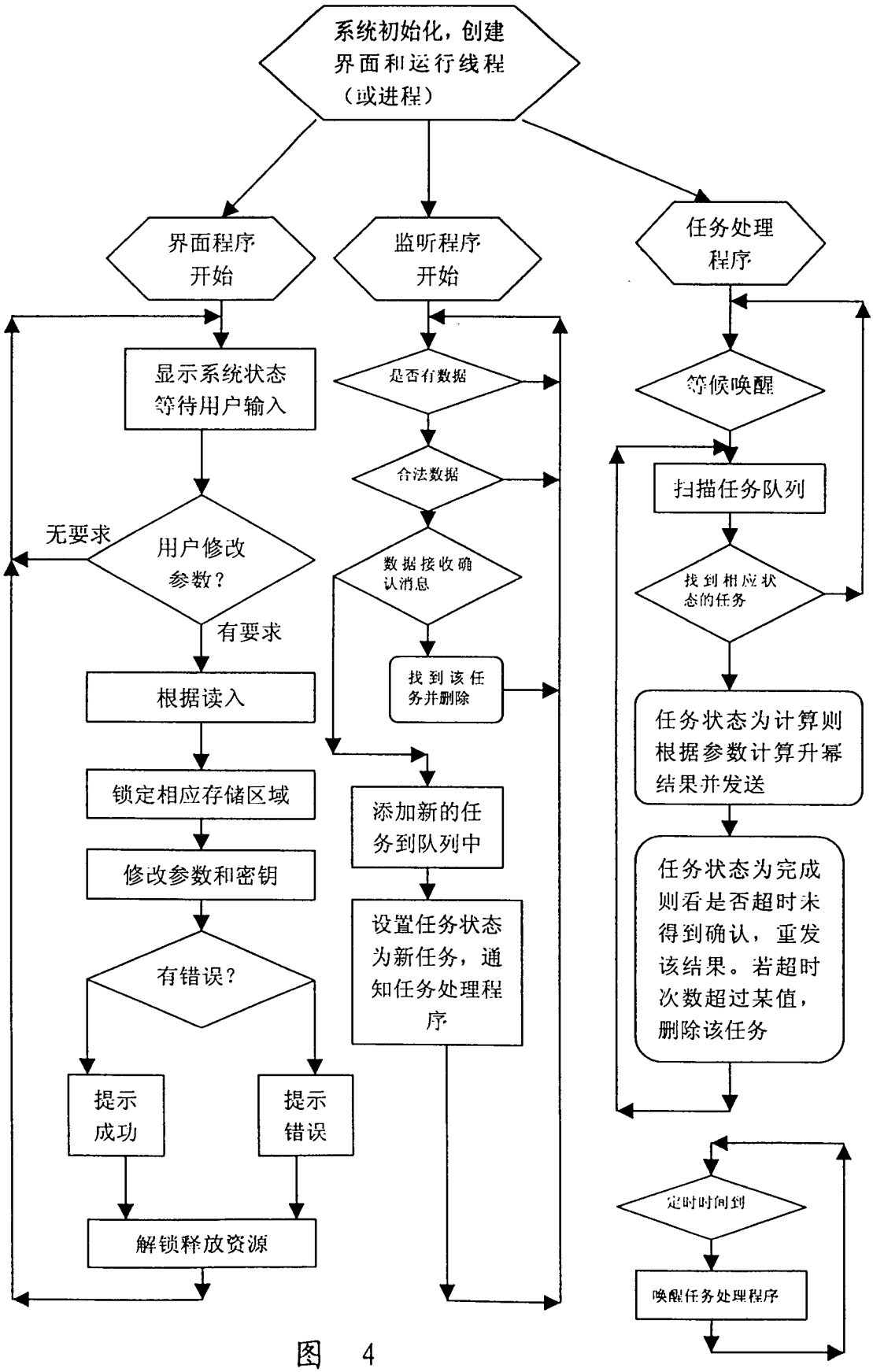

系统初始化，创建界面和运行线程（或进程）
界面程序开始
监听程序开始
任务处理程序
显示系统状态等待用户输入
是否有数据
等候唤醒
合法数据
扫描任务队列
无要求
用户修改参数？
数据接收确认消息
找到相应状态的任务
有要求
找到该任务并删除
根据读入
任务状态为计算则根据参数计算升幂结果并发送
锁定相应存储区域
添加新的任务到队列中
任务状态为完成则看是否超时未得到确认，重发该结果。若超时次数超过某值，删除该任务
修改参数和密钥
设置任务状态为新任务，通知任务处理程序
有错误？
提示成功
提示错误
定时时间到
解锁释放资源
唤醒任务处理程序

图 4

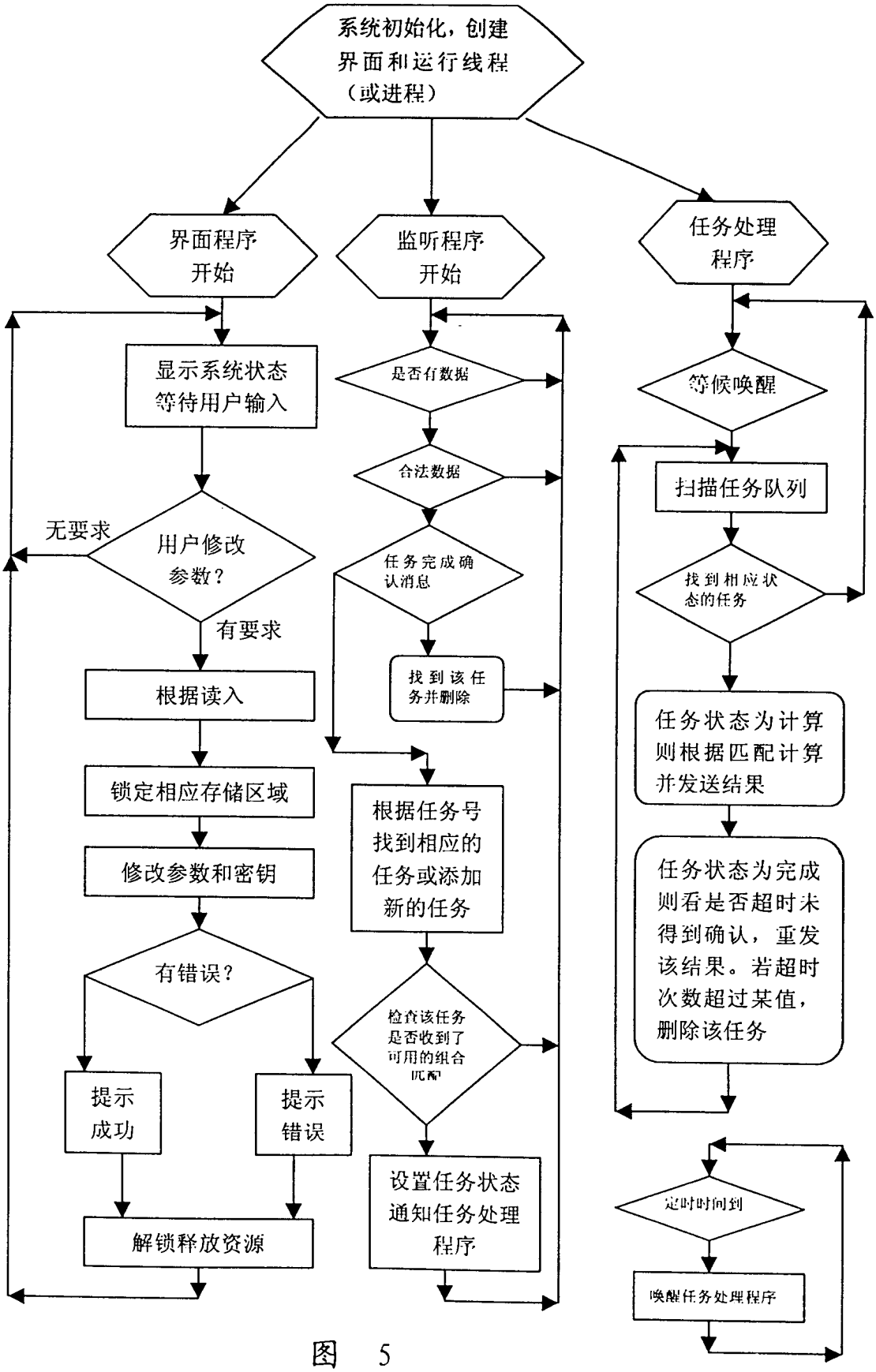

系统初始化，创建界面和运行线程（或进程）
界面程序开始
监听程序开始
任务处理程序
显示系统状态等待用户输入
用户修改参数？
无要求
有要求
根据读入
锁定相应存储区域
修改参数和密钥
有错误？
提示成功
提示错误
解锁释放资源
是否有数据
合法数据
任务完成确认消息
找到该任务并删除
根据任务号找到相应的任务或添加新的任务
检查该任务是否收到了可用的组合匹配
设置任务状态通知任务处理程序
等候唤醒
扫描任务队列
找到相应状态的任务
任务状态为计算则根据匹配计算并发送结果
任务状态为完成则看是否超时未得到确认，重发该结果。若超时次数超过某值，删除该任务
定时时间到
唤醒任务处理程序

图 5