

[19] 中华人民共和国国家知识产权局



## [12] 发明专利申请公布说明书

[21] 申请号 200810046535.7

[51] Int. Cl.

H04L 9/32 (2006.01)

H04L 9/30 (2006.01)

[43] 公开日 2009 年 5 月 6 日

[11] 公开号 CN 101425902A

[22] 申请日 2008.11.12

[21] 申请号 200810046535.7

[71] 申请人 电子科技大学

地址 610054 四川省成都市建设北路二段 4  
号

[72] 发明人 许春香 张 辉

权利要求书 3 页 说明书 10 页

### [54] 发明名称

一个具有前向安全的门限数字签名方法与系  
统

### [57] 摘要

本发明属于信息安全技术领域，涉及对数字信息进行签名问题，更确切地说是涉及一种能够增加敌手窃取签名密钥难度并且能够减轻签名密钥泄露影响的数字签名方法与系统。该签名方法通过应用 Shamir 秘密共享技术以及多方安全计算技术为一个经典的基础数字签名方法添加了门限机制与子秘密更新机制。门限机制增强了签名密钥的安全性，并可以起到权利分散的作用；子秘密更新机制实现了签名密钥的前向安全，即：即使敌手获得当前时间段的签名密钥，敌手也不能够通过该密钥伪造出一个属于前一时间段的合法签名，保护了以前的签名的有效性，降低了密钥泄露的损失。另外，该签名方法还包括一个成员加入机制，加强了方案的安全性和适用范围。

1、一种数字签名方法与签名系统，该签名方法具有门限机制、子秘密更新机制以及成员加入机制。其特征是，以 Shamir-SS 协议、Mult-SS 协议、Joint-Shamir-RSS 协议为基础模块，设计出的前向安全的门限签名方案(FST-SIG 签名方案)。整个数字签名方案包括五个部分：密钥生成协议、密钥进化协议、签名协议、签名验证算法、新成员加入协议。方案中涉及到的协议及算法的核心内容如下：

**Protocol FST-SIG keygen (k, T) // 密钥生成协议**

- (1). 分发者挑选两个随机的大素数  $p$  和  $q$ ，并且满足  $p \equiv q \equiv 3 \pmod{4}$ ， $p, q$  需要保密；
- (2). 分发者计算  $N$ ，使  $N = pq$ ；
- (3). 分发者在  $Z_N^*$  中随机选取  $S_0$  并计算  $U$ ， $U = (S_0^{2^{l(T+1)}})^{-1}$ ；
- (4). 分发者利用 Shamir-SS，在  $Z_n$  上计算  $S_0$  的子秘密： $S_0^{(1)}, S_0^{(2)}, \dots, S_0^{(n)}$ ；
- (5). 令  $SK_0^{(\rho)} = (N, T, 0, S_0^{(\rho)})$  ( $\rho = 1, 2, \dots, n$ )， $PK = (N, T, U)$ ；
- (6). 分发者通过保密的信道将  $SK_0^{(\rho)}$  发送给第  $\rho$  个参与者并发布签名验证密钥  $PK$ 。

**Protocol FST-SIGsign (m, j) // 签名协议**

- (1). 参与者利用 Joint-Shamir-RSS 共同生成随机数  $R$  ( $R \in Z_N$ )，每个参与者拥有  $R$  的子秘密  $R^{(\rho)}$ ；
- (2). 参与者根据  $R^{(\rho)}$  利用 Mult-SS 计算  $Y$ ，使  $Y = R^{2^{l(T+1-j)}}$ ；
- (3). 每一个参与者  $\rho$  计算  $\sigma = H(j, Y, m)$ ；
- (4). 利用 Mult-SS 参与者联合计算  $Z = RS_j^\sigma$ ；
- (5). 公布消息  $m$  的签名  $< j, (Z, \sigma) >$ 。

**Algorithm FST-SIG.verify (m, PK, sign) // 签名验证算法**

假设：PK 是  $(N, U, T)$ ；sign 是  $< j, (Z, \sigma) >$ ；

if  $Z \equiv 0 \pmod{N}$

```
return(0);
```

```
else  $Y' = Z^{2^{\lceil T+1-j \rceil}} U^\sigma \bmod N;$ 
```

```
if  $\sigma = H(j, Y', m)$ 
```

```
then return(1);
```

```
else return(0);
```

**Protocol FST-SIGupdate(j) // 密钥进化协议**

(1). 如果  $j=T$ , 则返回空串; 否则, 执行:

(2). 参与者根据各自的子秘密  $S_{j-1}^{(\rho)}$  利用 Mult-SS, 计算  $S_{j-1}$  的  $2^t$  次方  $S_j$  的子秘密  $S_j^{(\rho)}$ ;

(3). 每个参与者  $\rho$  删除  $S_{j-1}^{(\rho)}$ 。

**Protocol FST-SIGjion(j,n+1) // 成员加入协议**

(1). 每一个参与者  $P_i, i \in \{1 \dots n\}$  在  $Z_q$  上选取一个随机  $t$  次多项式  $\delta_i(x)$ , 满足  $\delta_i(n+1) = 0$ 。

(可以这样选取: 在  $Z_q$  上选取随机数  $\{\delta_{ij}\}_{j \in \{1 \dots t\}}$  然后计算  $\delta_{i0} = -\sum_{j \in \{1 \dots t\}} \delta_{ij} (n+1)^j \pmod q$ 。)

(2). 每个参与者  $P_i$  使用其他参与者  $P_j$  的公钥加密  $\delta_i(j)$  ( $j \in \{1 \dots n\}, j \neq i$ ) 得到

$\{ENC_j(\delta_i(j))\}$  并广播。

(3). 每个参与者  $P_i$  计算  $S_j^{(i)} = S_j^{(i)} + \sum_{P_j \in D} \delta_j(i)$  并将  $S_j^{(i)}$  保密传送给  $P_{n+1}$ 。

(4). 新加入者  $P_{n+1}$  获得所有的  $S_j^{(i)}$  用拉格朗日插值法恢复出属于他的子秘密  $S_j^{(n+1)}$ , 进而

获得签名子密钥  $SK_j^{(n+1)} = (N, T, j, S_j^{(n+1)})$ 。

### Shamir-SS 算法

算法参数:  $Z, s, n, t$

(1). 执行者在集合  $Z$  中选择  $t$  个随机数  $a_1, a_2, \dots, a_t$  作为系数, 以秘密  $s$  作为常数项构造  $t$

次多项式  $f(x) = s + a_1x + a_2x^2 + \dots + a_tx^t$ ;

(2). 执行者为多项式赋值，得到关于秘密  $s$  的子秘密  $s_1 = f(1), s_2 = f(2), \dots, s_n = f(n)$ 。

### Mult-SS 协议

参与者  $P_i$  的输入：  $f_\alpha(i)$  和  $f_\beta(i)$  的值

(1). 参与者选取一个随机  $t$  次多项式  $h_i(x)$ ，满足  $h_i(0) = f_\alpha(i)f_\beta(i)$ ，用各个参与者对应的公钥加密属于他们的值  $h_i(j)$  得到  $E_{PK_j}(h_i(j))$ ， $1 \leq j \leq 2t+1$  并以广播的形式将这些加密后的子秘密公布出去。

(2). 每一个参与者  $P_j$  接收  $E_{PK_j}(h_i(j))$  解密出  $h_i(j)$ ，然后计算属于他的  $\alpha\beta$  的子秘密：

$$H(j) = \sum_{i=1}^{2t+1} \lambda_i h_i(j).$$

### Joint-Shamir-RSS 协议

(1). 参与者  $P_i$  选取一个随机数  $s_i$  作为秘密，并选取  $t$  个随机数  $a_{i1}, a_{i2}, \dots, a_{it}$  作为系数构造  $t$  次多项式  $f_i(x) = s_i + a_{i1}x + a_{i2}x^2 + \dots + a_{it}x^t$ ；

(2). 参与者  $P_i$  计算属于每一个参与者的子秘密： $S_i^{(1)}, S_i^{(2)}, \dots, S_i^{(n)}$ ，用各个参与者对应的公钥加密属于他们的子秘密得到  $E_{PK_1}(S_i^{(1)}), E_{PK_2}(S_i^{(2)}), \dots, E_{PK_n}(S_i^{(n)})$ ，并以广播的形式将这些加密后的子秘密公布出去；

(3). 每一个参与者  $P_j$  解密所有接收到的  $E_{PK_i}(S_i^{(j)})$  得  $S_i^{(j)}$ ，而参与者  $P_j$  掌握的对应的联合生成的随机数的子秘密为  $S_1^{(j)} + S_2^{(j)} + \dots + S_n^{(j)}$ 。

## 一个具有前向安全的门限数字签名方法与系统

### 技术领域

本发明属于信息安全技术领域，涉及对数字信息进行签名问题，更确切地说是涉及一种能够增加敌手窃取签名密钥难度并且能够减轻签名密钥泄露影响的数字签名方法与系统。

### 背景技术

前向安全签名的概念在 1997 年由 Anderson 引入，解决了通常数字签名的一些缺陷：一旦秘密密钥丢失(或被窃取)，以前由这个密钥生成的所有签名都变得无效。为了减少这样的损失，Anderson 提出把密钥的有效期分成时段，在每个时段的最后，签名者以一个单向的模式从当前时段的秘密密钥得到一个新的下一个时段秘密密钥，并且安全的删除不再使用的秘密密钥。而在整个密钥的生命周期里公钥不变，这个方法确保了泄露密钥的时段以前的所有签名的有效性。在最近这些年，具备前向安全的数字签名方案得到研究和发展。Bellare 和 Miner 通过对通常签名方案安全定义的扩展，给出了前向安全正式的定义，同时提出了两个方案：一个是在普通签名基础上使用树型结构的证书链构造的方案；另一个是修改通常的签名方案(Fiat-Shamir 签名方案)。2001 年，Abdalla 和 Miner 给出一个前向安全的门限数字签名，但是该方案的签名密钥和验证密钥都比较长。

我们这里要研究的前向安全的门限签名方案应用门限机制对消息进行签名，即将签名密钥分割成多份，分别有多人保管，须由一定数目的签名子密钥拥有者联合才能够产生一个有效的签名，该签名等同于由签名密钥直接产生并可使用对应的公开密钥验证其有效性。方案还采用密钥进化机制，把整个周期（一般是声明的公钥有效期）分成更小的一个个时段，在整个周期内，公钥是固定不变的，而私钥是随时段不断进化。

一个标准的数字签名包括密钥生成协议、签名协议和签名验证算法，而在前向安全的数字签名方案中，还包括一个密钥进化协议，这个协议是用来说明在整个周期中，密钥是怎样进化的。另外，一个不可忽视的问题是在门限签名系统中，成员签名子密钥会出现损坏或丢失，当子密钥损失数目超过一定值后，整个系统将不可用，另外针对某具体应用，可能会有向系统中加入新成员的应用需求，因此一个完善的前向安全的门限签名方案还应该包括一个成员加入协议。

一个  $A(t, k_s, k_u, k_j, n)$  前向安全的门限数字签名是指该方案最多可以在  $t$  个参与者被攻陷后，仍能保证其签名密钥的安全性，它的工作过程如下：

在密钥生成阶段，分发者生成公、私钥并将私钥分割后分发给  $n$  个参与者。

在每一个时间段的开始时刻，由  $n$  个参与者中的  $k_u$  个没被攻陷的参与者执行密钥进化协议，协议执行后，这个时段的私钥将改变，并且每一个参与者，无论是否参与密钥进化协议，都会得到这个时段属于自己的新的子密钥。

要对一个消息  $m$  进行签名，需  $k_s$  个参与者共同执行签名协议，进而生成一个使用当前时段签名密钥签署的消息签名。这个签名中将包括本时段的序号并可以由公钥验证通过。

当有新成员要加入系统时，需要  $k_f$  个参与者共同执行成员加入协议，为新加入成员生成一个这一时段属于他的签名子密钥。该过程不泄露任何原有成员的秘密信息，并且新成员加入后，与原有成员拥有相同的地位。

验证过程同普通数字签名验证过程相同。任何持有公钥的一方都可以执行该过程，验证结果将是“接受”或“拒绝”，以告知验证者被验证的签名是否有效。我们假设前向安全的门限数字签名方案产生的签名形式是  $\langle j, tag \rangle$ ，其中  $j$  为产生该签名的时段序号，对于一个前向安全的门限签名方案，在时段  $l$ ，即使敌手攻陷  $t$  个以上参与者，他也不能成功伪造一个签名  $\langle j, tag \rangle$  满足  $verify_{PK}(m, \langle j, tag \rangle) = 1$  并且  $j < l$ 。这里， $verify()$  是签名验证算法，也就是说，敌手即使在第  $l$  时段获得签名密钥，他也不能伪造出  $l$  时段之前的有效签名。前向安全的方案都要求使用者删除前一时段使用的密钥，并将这一步骤作为密钥进化协议的一部分，这很关键，否则，敌手攻陷系统后将根据以前时段的子秘密信息获得以前时段的签名密钥来生成以前时段的有效签名。

在我们的发明中，应用了下面的数论知识：

设  $k$  和  $l$  是两个安全参数， $p_1 \equiv p_2 \equiv 3 \pmod{4}$  是两个大素数。 $N = p_1 p_2$  是一个  $k$  位的整数（即  $N$  是一个 Blum 整数）。为了简化计算，我们可以合理地假设  $N > 2k - 1$ ，并且  $|Z_N^*| = N \cdot p_1 \cdot p_2 + 1 \geq 2^{k-1}$ 。记  $Q$  是模  $N$  的二次剩余集合。由数论中的定理可知  $|Q| \geq 2^{k-3}$ ，并且对于集合  $Q$  中任意元素  $x$ ， $x$  的四个平方根中有且仅有一个属于  $Q$ ，因此，平方在  $Q$  上是一个置换。从现在起，当我们说  $x$  的平方根时，我们是指属于  $Q$  的那个平方根。

令  $U \in Q$ ，定义： $F_0(Z) = Z^2 \pmod{N}$ ， $F_1 = UZ^2 \pmod{N}$ 。对于  $l$  位二进制串  $\sigma = \sigma_1 \cdots \sigma_l$ ，定义  $F_\sigma : Q \rightarrow Q$  为： $F_\sigma(Z) = F_{\sigma_l}(\cdots(F_{\sigma_2}(F_{\sigma_1}(Z)))\cdots) = Z^{2^l} U^\sigma \pmod{N}$ 。（注意：这里的  $U^\sigma$  并非通常意义上的  $U$  的  $\sigma$  次幂， $\sigma$  在这里是一个二进制串，而非代表一个整数）因为平方是在  $Q$  上的置换，而  $U \in Q$ ，故  $F_\sigma$  在  $Q$  上也是一个置换。

在知道  $U$  和  $N$  的前提下,  $F_\sigma(Z)$  可以快速计算, 同时, 如果知道  $p_1$  和  $p_2$ , 则对于给定的  $Y$ , 可以快速计算出  $Z = F_\sigma^{-1}(Y)$ 。(通过计算  $S = \sqrt[U^2]{Y} \bmod N$ ,  $Z = Y^{2^{-t}} S^\sigma \bmod N$  可得。这些计算可以先分别计算  $\bmod p_1$  和  $\bmod p_2$  的结果, 然后用中国剩余定理合并。) 然而, 如果不知道  $U$  的平方根, 那么  $F_\sigma^{-1}$  是难于计算的。下面我们给出证明:

引理: 给定  $Y \in Q$  和两个不同的等长的串  $\sigma$  和  $\tau$ ,  $Z_1 = F_\sigma^{-1}(Y)$ ,  $Z_2 = F_\tau^{-1}(Y)$ , 能够计算出  $V \in Q$  且  $V^2 \equiv U \bmod N$

证明: 如果  $|\sigma|=|\tau|=1$ , (无损于一般性地), 令  $\sigma=0, \tau=1$ , 则  $F_0(Z_1)=F_1(Z_2)=Y$ , 则  $Z_1^2 \equiv UZ_2^2 \bmod N$ , 于是得到  $V = Z_1 / Z_2 \bmod N$ 。在归纳证明中, 令  $\sigma$  和  $\tau$  为两个长  $m+1$  位的串,  $\sigma'$  和  $\tau'$  为其对应的前  $m$  位。如果  $F_{\sigma'}(Z_1)=F_{\tau'}(Z_2)$ , 则归纳假设完成, 否则  $\sigma$  和  $\tau$  最后一位应该是不同的, 于是 (不失一般性), 假设  $\sigma$  最后一位为 0,  $\tau$  最后一位为 1, 则  $F_0(F_{\sigma'}(Z_1))=F_1(F_{\tau'}(Z_2))$ , 同上可以得证。

根据上面提到的单向函数可以构造下面的数字签名方案。本文提出的前向安全的门限签名方案就是以该方案为基础, 加入门限机制和前向安全机制得到的。

签名者生成大模数  $N$  和一个随机数  $S \in Q$ ,  $S$  作为签名密钥, 需安全保管。计算  $U=(S^2)^{-1}$  并公开  $(N, U)$  作为公钥。 $H(\cdot)$  为一个输出位数为  $l$  的哈希函数。

要对消息  $M$  签名时, 首先生成随机数  $R \in Q$ , 计算  $Y=R^{2^l}$ ,  $\sigma = H(Y, M)$ ,  $Z=F_\sigma^{-1}(Y)=RS^\sigma \bmod N$ 。输出  $(Z, \sigma)$  作为消息  $M$  的签名。

验证者验证签名时, 首先验证  $Z \neq 0 \bmod N$ , 而后计算  $Y'=F_\sigma(Z)=Z^{2^l} U^\sigma \bmod N$ , 最后检验  $\sigma = H(Y', M)$  是否成立, 成立则证明该签名是  $M$  的合法签名。

该数字签名算法是一个经典算法, 其正确性和安全性已经由许多前辈学者证明过并经历了长时间的实践检验。

## 发明内容

本发明的目的是设计一种数字签名方法与签名系统, 该签名方法具有门限机制与子秘密更新机制。门限机制增强了签名密钥的安全性, 并可以起到权利分散的作用; 子秘密更新机制实现了签名密钥的前向安全, 即: 即使敌手获得当前时间段的签名密钥, 敌手也不能够通

过该密钥伪造出一个属于前一时间段的合法签名，保护了以前的签名的有效性，降低了密钥泄露的损失。另外，该签名方法还拥有成员加入机制，加强了方案的安全性和适用范围。

方法的特点：

1. 本签名方法是一个 $(t, 2t+1, 2t+1, t+1, n)$ 的签名方案，其中  $n \geq 3t+1$ ，
2. 密钥长度短：签名密钥（私钥）与验证密钥（公钥）的长度是常数，与总时段数无关，并且与其他不具前向安全性的签名方案的密钥长度相当。
3. 在门限数字签名方案中加入前向安全机制的方法具有不额外增加计算量的特点，即该前向安全门限签名方案与普通门限数字签名方案具有相同的效率。
4. 完善的成员加入机制，使系统更安全更灵活。

本发明的技术方案是这样的：

整个方案包括五个部分：密钥生成协议、密钥进化协议、签名协议、签名验证算法、新成员加入协议。另外，方案中应用到一些关于秘密共享以及安全多方计算的知识，我们将这些知识作为基础模块应用在我们的方案中。下面我们首先介绍基于这些基础知识协议模块，然后给出我们发明的前向安全的门限数字签名方案的完整描述。

### 一、Shamir 秘密共享方案（Shamir-SS）

1979 年，Shamir 提出秘密共享的思想，并给出了一个有限域上的秘密共享方案。具体方案如下：设  $GF(q)$  是一有限域，其中  $q$  是一大素数，满足  $q \geq n+1$ ，秘密  $a_0$  是在  $GF(q) \setminus \{0\}$  上均匀选取的一个随机数， $k-1$  个系数  $a_1, a_2, \dots, a_{k-1}$  的选取也满足  $a_i \in_R GF(q) \setminus \{0\}$ 。在  $GF(q)$  上构造一个  $t$  次多项式  $f(x) = a_0 + a_1x + a_2x^2 + \dots + a_tx^t$ 。

$n$  个参与者记为  $P_1, P_2 \dots P_n$ ， $P_i$  分配到的子密钥为  $f(i)$ 。如果任意  $t+1$  个参与者  $P_{i_1}, \dots, P_{i_{t+1}}$  ( $1 \leq i_1 < i_2 < \dots < i_{t+1} \leq n$ ) 要想得到秘密  $a_0$ ，可使用  $\{(i_j, f(i_j)) \mid j = 1, 2, \dots, t+1\}$  构造如下的线性方程组：

$$\begin{cases} a_0 + a_1(i_1) + \dots + a_t(i_1)^t = f(i_1) \\ a_0 + a_1(i_2) + \dots + a_t(i_2)^t = f(i_2) \\ \dots \\ a_0 + a_1(i_{t+1}) + \dots + a_t(i_{t+1})^t = f(i_{t+1}) \end{cases}$$

因为  $i_l (1 \leq l \leq t+1)$  均不相同，所以可由 Lagrange 插值公式构造如下的多项式：

$$f(x) = \sum_{j=1}^{t+1} f(i_j) \prod_{\substack{l=1 \\ l \neq j}}^{t+1} \frac{(x - i_l)}{(i_j - i_l)} \pmod{q}$$

从而可得秘密  $a_0 = f(0)$ 。

由上面知识我们构造下面的秘密共享算法：

### Shamir-SS 算法

算法参数：  $Z, s, n, t$

1. 执行者在集合  $Z$  中选择  $t$  个随机数  $a_1, a_2, \dots, a_t$  作为系数，以秘密  $s$  作为常数项构造  $t$  次多项式  $f(x) = s + a_1x + a_2x^2 + \dots + a_tx^t$ ；
2. 执行者为多项式赋值，得到关于秘密  $s$  的子秘密  $s_1 = f(1), s_2 = f(2), \dots, s_n = f(n)$ 。

算法说明：Shamir-SS 实现将秘密  $s$  利用 Shamir 秘密共享方案分割成  $n$  份，得到子秘密  $s_1, s_2, \dots, s_n$  并且其中的  $t+1$  份联合可以恢复出共享秘密  $s$ 。

## 三、安全多方计算

在我们的协议的子秘密更新的部分中，需要解决参与者联合计算生成原共享秘密平方的子秘密的问题，这个问题可归类于计算生成两个共享秘密的乘积的子秘密的问题，即：假设  $n$  个参与者，通过  $t$  次多项式  $f_\alpha(x)$  和  $f_\beta(x)$  共享秘密  $\alpha$  和  $\beta$ ，即  $f_\alpha(0) = \alpha, f_\beta(0) = \beta$ 。现在，参与者要联合计算获得  $\alpha\beta$  的子秘密。对于这个问题，有下面的解决方案。

令  $P_i$  所拥有的关于  $\alpha, \beta$  的子秘密记为  $f_\alpha(i), f_\beta(i)$ ， $f_\alpha(x)$  与  $f_\beta(x)$  的乘积为  $f_\alpha(x)f_\beta(x) = \gamma_{2t}x^{2t} + \dots + \gamma_1x + \alpha\beta = f_{\alpha\beta}(x)$ ，对于  $1 \leq i \leq 2t+1$ ，有  $f_{\alpha\beta}(i) = f_\alpha(i)f_\beta(i)$ ，所以可以写成：

$$A \begin{bmatrix} \alpha\beta \\ \gamma_1 \\ \vdots \\ \gamma_{2t} \end{bmatrix} = \begin{bmatrix} f_{\alpha\beta}(1) \\ f_{\alpha\beta}(2) \\ \vdots \\ f_{\alpha\beta}(2t+1) \end{bmatrix}$$

这里矩阵  $A = (a_{ij})$  是  $(2t+1) \times (2t+1)$  的范德蒙矩阵， $a_{ij} = i^{j-1}$ 。显然， $A$  是可逆矩阵，设  $A$  的逆矩阵为  $A^{-1}$ ， $A^{-1}$  的第一行记为  $(\lambda_1 \dots \lambda_{2t+1})$ ，当  $t$  确定的情况下， $\lambda_1 \dots \lambda_{2t+1}$  为确定的常数， $\alpha\beta = \lambda_1 f_{\alpha\beta}(1) + \dots + \lambda_{2t+1} f_{\alpha\beta}(2t+1)$ 。如果给定  $t$  次多项式  $h_1(x), \dots, h_{2t+1}(x)$  满足  $h_i(0) = f_{\alpha\beta}(i)$ ， $(1 \leq i \leq 2t+1)$ ，定义：

$$H(x) = \sum_{i=1}^{2t+1} \lambda_i h_i(x)$$

则有  $H(0) = \lambda_1 f_{\alpha\beta}(1) + \dots + \lambda_{2t+1} f_{\alpha\beta}(2t+1) = \alpha\beta$ 。因此，如果每一个参与者用一个多项式  $h_i(x)$  共享他的子秘密，那么  $H(x)$  就是可以将  $\alpha\beta$  秘密共享的那个  $t$  次多项式。

根据上面的知识，我们设计了下面的 Mult-SS 协议：

#### Mult-SS 协议

参与者  $P_i$  的输入： $f_\alpha(i)$  和  $f_\beta(i)$  的值

1. 参与者选取一个随机  $t$  次多项式  $h_i(x)$ ，满足  $h_i(0) = f_\alpha(i)f_\beta(i)$ ，用各个参与者对应的公钥加密属于他们的值  $h_i(j)$  得到  $E_{PK_j}(h_i(j))$ ， $1 \leq j \leq 2t+1$  并以广播的形式将这些加密后的子秘密公布出去。
2. 每一个参与者  $P_j$  接收  $E_{PK_j}(h_i(j))$  解密出  $h_i(j)$ ，然后计算属于他的  $\alpha\beta$  的子秘密：

$$H(j) = \sum_{i=1}^{2t+1} \lambda_i h_i(j)。$$

协议说明：协议利用安全多方计算实现共享两个共享秘密乘积，参与者利用该协议可以利用自己当前拥有的两个共享秘密  $\alpha$  和  $\beta$  的子秘密  $f_\alpha(i)$  和  $f_\beta(i)$  得到属于他的  $\alpha\beta$  的子秘密。

### 三、联合生成并秘密共享随机数(Joint-Shamir-RSS)

在门限数字签名方案中，要涉及参与者联合生成并秘密共享随机数的问题，这个问题可以用下面的方法解决。每一个参与者  $P_i$  选取一个随机数并将它作为秘密，用 Shamir 秘密共享方案，计算属于每一个参与者的子秘密： $S_i^{(1)}, S_i^{(2)} \dots$ ，用各个参与者对应的公钥加密属于他

们的子秘密得到  $E_{PK_1}(S_i^{(1)}), E_{PK_2}(S_i^{(2)}) \dots$ , 并以广播的形式将这些加密后的子秘密公布出去。

这样, 联合生成的随机数就是各个参与者选取的随机数之和, 而每一个参与者  $P_j$  掌握的对应的子秘密为  $S_1^{(j)} + S_2^{(j)} + \dots$ 。

### Joint-Shamir-RSS 协议

1. 参与者  $P_i$  选取一个随机数  $s_i$  作为秘密, 并选取  $t$  个随机数  $a_{i1}, a_{i2}, \dots, a_{it}$  作为系数构造  $t$  次多项式  $f_i(x) = s_i + a_{i1}x + a_{i2}x^2 + \dots + a_{it}x^t$ ;
2. 参与者  $P_i$  计算属于每一个参与者的子秘密:  $S_i^{(1)}, S_i^{(2)}, \dots, S_i^{(n)}$ , 用各个参与者对应的公钥加密属于他们的子秘密得到  $E_{PK_1}(S_i^{(1)}), E_{PK_2}(S_i^{(2)}), \dots, E_{PK_n}(S_i^{(n)})$ , 并以广播的形式将这些加密后的子秘密公布出去。
3. 每一个参与者  $P_j$  解密所有接收到的  $E_{PK_j}(S_i^{(j)})$  得  $S_i^{(j)}$ , 而参与者  $P_j$  掌握的对应的联合生成的随机数的子秘密为  $S_1^{(j)} + S_2^{(j)} + \dots + S_n^{(j)}$

协议说明: 协议通过每一个参与者选取一个随机数并执行一个类似于 Shamir-SS 协议的过程, 使所有参与者共同生成一个随机数, 而每个参与者掌握这个随机数对应的子秘密。

下面我给出 FST-SIG 数字签名方案的完整描述, 其中所有计算都是  $\bmod N$  的。

#### Protocol FST-SIG keygen ( $k, T$ )

1. 分发者挑选两个随机的大素数  $p$  和  $q$ , 并且满足  $p \equiv q \equiv 3 \pmod{4}$ ,  $p, q$  需要保密;
2. 分发者计算  $N$ , 使  $N = pq$ ;
3. 分发者在  $Z_N^*$  中随机选取  $S_0$  并计算  $U$ ,  $U = (S_0^{2^{T+1}})^{-1}$ ;
4. 分发者利用 Shamir-SS, 在  $Z_n$  上计算  $S_0$  的子秘密:  $S_0^{(1)}, S_0^{(2)}, \dots, S_0^{(n)}$ ;
5. 令  $SK_0^{(\rho)} = (N, T, 0, S_0^{(\rho)})$  ( $\rho = 1, 2, \dots, n$ ),  $PK = (N, T, U)$ ;
6. 分发者通过保密的信道将  $SK_0^{(\rho)}$  发送给第  $\rho$  个参与者并发布签名验证密钥  $PK$ 。

#### Protocol FST-SIG.sign ( $m, j$ )

1. 参与者利用 Joint-Shamir-RSS 共同生成随机数  $R (R \in Z_N)$ , 每个参与者拥有  $R$  的子秘密  $R^{(\rho)}$ ;
2. 参与者根据  $R^{(\rho)}$  利用 Mult-SS 计算  $Y$ , 使  $Y = R^{2^l(r+1-j)}$ ;
3. 每一个参与者  $\rho$  计算  $\sigma = H(j, Y, m)$ ;
4. 利用 Mult-SS 参与者联合计算  $Z = RS_j^\sigma$ ;
5. 公布消息  $m$  的签名  $< j, (Z, \sigma) >$ 。

**Algorithm FST-SIGverify (m, PK, sign)**

假设: PK 是  $(N, U, T)$ ; sign 是  $< j, (Z, \sigma) >$ ;

```

if  $Z \equiv 0 \pmod{N}$ 
    return(0);
else  $Y' = Z^{2^l(T+1-j)} U^\sigma \pmod{N}$ ;
    if  $\sigma = H(j, Y', m)$ 
        then return(1);
    else return(0);

```

**Protocol FST-SIGupdate(j)**

1. 如果  $j=T$ , 则返回空串; 否则, 执行:
2. 参与者根据各自的子秘密  $S_{j-1}^{(\rho)}$  利用 Mult-SS, 计算  $S_{j-1}$  的  $2^l$  次方  $S_j$  的子秘密  $S_j^{(\rho)}$ ;
3. 每个参与者  $\rho$  删除  $S_{j-1}^{(\rho)}$ ;

**Protocol FST-SIGjoin(j,n+1)**

1. 每一个参与者  $P_i, i \in \{1 \cdots n\}$  在  $Z_q$  上选取一个随机  $t$  次多项式  $\delta_i(x)$ , 满足  $\delta_i(n+1) = 0$ 。

(可以这样选取：在  $Z_q$  上选取随机数  $\{\delta_j\}_{j \in \{1 \dots t\}}$  然后计算  $\delta_{i0} = -\sum_{j \in \{1 \dots t\}} \delta_j (n+1)^j \pmod{q}$ 。)

2. 每个参与者  $P_i$  使用其他参与者  $P_j$  的公钥加密  $\delta_i(j) (j \in \{1 \dots n\}, j \neq i)$  得到  $\{ENC_j(\delta_i(j))\}$  并广播。

3. 每个参与者  $P_i$  计算  $S_j^{(i)} = S_j^{(i)} + \sum_{P_j \in D} \delta_j(i)$  并将  $S_j^{(i)}$  保密传送给  $P_{n+1}$ 。

4. 新加入者  $P_{n+1}$  获得所有的  $S_j^{(i)}$  用拉格朗日插值法恢复出属于他的子秘密  $S_j^{(n+1)}$ ，进而获得签名子密钥  $SK_j^{(n+1)} = (N, T, j, S_j^{(n+1)})$ 。

下面分别对上述协议、算法进行说明：

Protocol FST-SIG.keygen ( $k, T$ ) 是密钥生成协议，由可信的分发者执行。分发者选择好密钥后，用我们上面提到过的 Shamir 秘密共享方案将秘密密钥共享生成子密钥  $SK_0^{(\rho)} = (N, T, 0, S_0^{(\rho)})$  并通过安全信道传送给每一个参与者。这里下标 0 代表该应用密钥的时段序号，上标  $(\rho)$  表示该密钥为参与者  $P_\rho$  拥有。我们需要强调一点是：在签名方案中应用 Shamir 秘密共享时，我们的计算是在  $Z_N$  上的，这里  $Z_N$  显然不是域，但是，我们仍然可以证明系统是可以正确运行的。首先，我们要求所有参与者的数目  $n$  要小于系统参与  $p, q$ ，其次我们要求分发者分配给参与者  $P_i$  的子秘密为  $f(i)$ 。这样，所有用于重构秘密的子秘密将都不含有因子  $p$  或  $q$ 。这样，在秘密恢复时，我们构造的  $(t+1) \times (t+1)$  范德蒙矩阵中的所有元素都含有因子  $p$  或  $q$ ，这样，因为矩阵的行列式为  $\prod_{1 \leq j < k \leq t} (x_{i_k} - x_{i_j}) \pmod{N}$ ，因此可以保证行列式值是与  $N$  互素的。因此，在上面提出的两个前提下，在  $Z_N$  上的 Shamir 秘密共享方案仍然是正确的。

Protocol FST-SIGsign ( $m, j$ ) 是签名协议，该协议需要  $2t+1$  个参与者参与完成。在基础的数字签名算法中，随机数  $R$  应该是取自于  $Z_N^*$ ，而在这里，为了加入门限机制，随机数需要由多方共同选取的，这样  $R$  是  $Z_N$  中的一个随机数。但是，一个数属于  $Z_N$  但不属于  $Z_N^*$  的概率是十分小的，大约是  $\frac{1}{p} + \frac{1}{q}$ ，是一个可以忽略的小概率。所以我们可以认为由这个签名协议生

成的签名仍然是有效的。

**Algorithm FST-SIGverify<sup>PK</sup>(m, PK, sign)**是签名验证算法，与普通数字签名方案的验证算法相同，该算法由任何拥有对应公钥的一方执行。

**Protocol FST-SIGupdate(j)**是密钥进化协议。在每个时段的开始，由 $2t+1$ 个参与者参与，执行密钥进化协议，完成密钥进化，获得该时段所使用的签名密钥。在时段 $j$ 的开始时刻，成功进行上次密钥进化的参与者拥有 $j-1$ 时段签名密钥 $SK_{j-1}$ 的子密钥 $SK_{j-1}^{(\rho)}$ ，利用 Mult-SS，参与者可计算 $S_{j-1}$ 的 $2^l$ 次方 $S_j$ 的子秘密 $S_j^{(\rho)}$ ，然后立即删除 $SK_{j-1}^{(\rho)}$ ，此时 $SK_j^{(\rho)} = (N, T, j, S_j^{(\rho)})$ 。要强调的是，除了参与密钥进化计算的 $2t+1$ 个参与者外，当前所有的没有被敌手阻断攻陷的参与者（包括在前一时段被阻断，现在恢复的参与者）都可以从这 $2t+1$ 个参与者获得足够信息，计算出属于自己这一时段的子秘密。

**Protocol FST-SIGjoin(j,n+1)**是新成员加入协议。该协议可以在任何时刻添加入新成员，例如在 $j$ 时段时，首先当前成员之间先依据申请加入成员的序列号 $n+1$ 共同生成一个 $t$ 次多项式，可以记为 $\delta_j(x)$ 满足 $\delta_j(n+1) = 0$ ，而后当前每个成员 $P_i$ 分别计算 $S_j^{(i)} = S_j^{(i)} + \delta_j(i)$ 并将 $S_j^{(i)}$ 保密传送给 $P_{n+1}$ ，新加入成员 $P_{n+1}$ 可以更加接收到得 $S_j^{(i)}$ 恢复出一个多项式，带入 $n+1$ 即可得 $S_j^{(n+1)}$ ，继而得到属于自己 $j$ 时段的子密钥 $SK_j^{(n+1)} = (N, T, j, S_j^{(n+1)})$ 。

### 具体实施方式

在我们前向安全的门限数字签名系统中，所有的参与方包括 $n$ 个签名服务器（即方案中的参与者）用 $P_i$ 表示 $1 \leq i \leq n$ ，他们都位于一个广播网络上，并且他们两两之间存在可进行保密通信的安全信道，这样的信道可以在广播信道上使用加密技术实现，使用的加密算法应该是前向安全的公钥加密算法。另外系统还包括一个可信的分发者，签名服务器与分发者之间可以进行广播通信也可以进行点对点的加密通信。其中任意一台签名服务器都可以接受签名请求，发起并组织一次签名，并最后将生成的签名发送给签名请求者。最后我们要求系统具有同步性，他们在协议的某一阶段可以同步地发送他们的信息。

本发明的发明内容部分对实施已经做出了详细说明，在此不再重复描述。但需要说明的是：针对不同的应用需求，不同的安全性等级要求，可以采用不同规模的参数： $N$ ， $l$ 等。因此本发明可以具有很多种具体的实施方式。