基于椭圆曲线的有序多重数字签名方案

严 安 杨明福
(华东理工大学信息学院 上海 200237)

摘 要 提出了基于椭圆曲线的有序多重签名方案,允许任意个数的人按顺序对同一文档签名,并将它保密地发送给接受者。该方案可以检测出不诚实签名者,并防止他们的欺骗行为,有效避免了签名死锁,因而更具安全性和实用性。

关键词 多重数字签名 有序 椭圆曲线密码体制 离散对数难题

A SEQUENTIAL DIGITAL MULTI-SIGNATURE SCHEME BASED ON ELLIPTIC CURVE CRYPTOSYSTEM

Yan An Yang Mingfu
(*Information School, East China University of Science and Technology, Shanghai 200237, China*)

Abstract This paper proposes a scheme based on Elliptic Curve Cryptosystem. It allows any number of signatories to sign a document orderly and send it to the receiver secretly. It can prevent some dishonest participants from cheating others, and also efficiently avoid signature deadlock, therefore, it is more secure and practical.

Keywords Multi-signature Sequential Elliptic curve cryptosystem Discrete logarithm problem

0 引 言

数字签名是密码学的重要问题之一,它是传统文件手写签名的模拟,能够实现用户对电子消息的认证。一般的数字签名方案包括 3 个过程:系统的初始化过程、签名产生过程和签名验证过程。签名算法是数字签名体制的基石。目前能够有效进行作为数字签名算法的有 3 类:(1)建立在大整数因子分解难题基础上;(2)建立在有限域的离散对数问题(DLP)上;(3)建立在椭圆曲线离散对数难题(ECDLP)上的数字签名算法。1992 年 Scott Vanstone 首先提出椭圆曲线数字签名算法 ECDSA。ECDSA 于 1998 年被接受为 ISO 标准(ISO14888-3),1999 年成为 ANSI 标准(ANSI X9. 62),2000 年成为 IEEE 标准(IEEE P1363)以及 FIPS 标准(FIPS 186-2)。文献[4]系统描述了 ECDSA 的实现方法。

在现实世界里,很多情况下会用到多重数字签名。多重数字签名即多个签名者签署同一份文件。目前主要有有序多重签名和广播多重签名两种多重数字签名方案。对于有序多重数字签名问题,Harn 和 Kiesler 提出了基于 RSA 的多重数字签名模型,允许任意个数的人对同一文档签名并将它保密地发送给接受者,但是需要多次处理且认证效率依赖于签名者的个数。文献[2]提出了一种基于 Elgamal 的有序多重签名方案,但是消息采用的是明文方式发送,不能保证信息的机密性。文献[1,3]于 1994 年提出了一种将 RSA 与 Elgamal 结合起来的高强度加密和签名算法,但是不适合作多重签名。

由于文献[4]的 ECDSA 方案不适合进行多重数字签名,文献[5]中对 ECDSA 方案进行改进,提出了一种适合进行椭圆曲线多重数字签名的方案,并以改进的方案为基础设计了一种椭圆曲线广播多重数字签名方案。本文在文献[5]的基础上提出了一种新的椭圆曲线有序多重数字签名方案。采用椭圆曲线密码体制进行多重数字签名,不仅使多重数字签名建立在椭圆曲线离散对数难题上,具有了更高的安全性,而且可以充分利用椭圆曲线密码体制的各种优点,缩短密钥长度、提高执行速度、减少占用存储空间、提高签名效率并且占用的带宽减小。

1 改进的椭圆曲线数字签名方案

1.1 椭圆曲线数字签名的产生

输入:签名消息 m;主域参数 $D=(E,q,G,n,h)$,其中 E 是 F_q(F_q 表示含 q 个元素的有限域)上定义的椭圆曲线,G 是 $E(F_q)$ 中的 n 阶素数点(即 $nG=O$),h 为余因子;系统用户私钥 d,相应公钥是 $Q=dG$;一个单向散列函数 SHA-1。

输出:椭圆曲线数字签名(R,S)。

签名过程:①选取一个随机数 k,$1\leqslant k\leqslant n-1$;

②计算 $R'=kG=(x_1,y_1)$,如果 $x_1=0$,转①;

③计算 $e=SHA\text{-}1(m)$;

④计算 $R=kR'$;

⑤计算 $S=(deG+kR')\ modn$,如果 $S=O$,转①;

⑥实体 A 对消息的签名是(R,S),算法结束。

收稿日期:2004-09-15。严安,硕士生,主研领域:计算机网络及其应用。

1.2　椭圆曲线数字签名的验证

输入:A 的数字签名(R,S),签名消息 m,主域参数 $D=(E,q,G,n,h)$ 以及 A 的公钥 Q。

输出 接受或者拒绝签名。

验证过程:①验证 R 和 S 是 E 上的点;

②计算 $e=SHA\text{-}1(m)$;

③计算 $X=S-eQ$,记 X 的坐标为(x_1,y_1),若 $X=O$ 拒绝签名;

④设 $R=(x_2,y_2)$;

⑤当且仅当 $x_1=x_2$ 时,接受签名。算法结束。

证明如下:$X=S-eQ=deG+kR'-eQ=kR'=R$,所以当且仅当 $x_1=x_2$ 时,接受签名。

2　椭圆曲线有序多重数字签名方案

本文在文献[5]的基础上提出了一种新的椭圆曲线有序多重数字签名方案,该方案包含以下 3 个步骤:系统初始化、签名的产生过程和验证过程。方案参与者有签名者及可信的中间机构 CA 中心。

2.1　系统初始化

假设有 N 个签名者 $A_1,A_2,\cdots,A_N$ 签署同一份信息 m,主域参数为 $D=(E,q,G,n,h)$,单向散列函数 H。

①随机选取 n 个不同元素 $d,a_1,a_2,\cdots,a_{n-1}\in F_q-\{0\}$,做 $(n-1)$ 阶多项式 $f(x)=d+a_1x+a_2x^2+\cdots+a_{n-1}x^{n-1}$,$f(0)=d$ 为多重签名私钥,必须保密。

②每个签名者 $U_i(1\leqslant i\leqslant N)$ 随机选择数 $k_i,1\leqslant k_i\leqslant n-1$,计算 $X_i=k_iG$。设 X_i 的坐标为(x_i,y_i)。若 $x_i=0$,则另取值。此时,k_i 是签名者的私钥,必须保密,X_i 为签名者的公钥,可公开。

③由①②得,每个签名者的多重签名私钥为 $f(x_i)$,签名公钥为 $Q_i=f(x_i)G$,而多重签名公钥为 $Q=f(0)G=dG$。

2.2　签名的产生过程

在有序多重数字签名方案中,CA 中心预先设计一种签名顺序$(U_1,U_2,\cdots,U_N)$,并将签名顺序发送给每一位签名者。由 CA 中心确定发送的消息 m,计算 $X=H(m,T)$,然后将消息(m,X)发送给第一个签名者 U_1,并给每个签名者发送签名时间标志 T,要求签名者在时间 ΔT 内签名,这样可以防止签名重播。U_1 收到消息(m,X)后先验证,通过后进行如下签名操作:

(1) 计算 $X=\sum_{i=1}^{N}X_i$,并将 X 发送给后续每一个签名者和 CA 中心。

(2) 计算 $R_1=k_1X$,$S_1=f(x_1)e\prod_{j=1,j\neq i}^{N}\frac{-x_j}{x_i-x_j}G+k_1X\bmod n$,$e$ 为消息的散列值,$e=H(m,T)$。

(3) 将消息$(m,(S_1,R_1))$发送给下一个签名者 U_2。

每一位签名者 $U_i(i\geqslant 2)$ 收到$(m,(S_{i-1},R_{i-1}))$后,先进行签名验证,通过后进行如下签名操作:

①计算 $R_i=k_iX+R_{i-1}$,$S_i=S_{i-1}+f(x_i)e\prod_{j=1,j\neq i}^{N}\frac{-x_j}{x_i-x_j}G+k_iX\bmod n$;

②将消息$(m,(S_i,R_i))$发送给下一位签名者 U_{i+1}。

2.3　签名的验证过程

在上述签名过程中,要求签名者 $U_i(i=1,2,\cdots,N)$ 对 U_{i-1} 的签名进行验证。各个签名者及 CA 中心的验证过程如下:

2.3.1　签名者 U_1

首先计算 $T_1=\Delta T_1+T$,再进行如下操作:

(1) 若(m,X)在 T_1 时刻之前到达,验证 $x=H(m,T)$ 是否成立。若成立则签名有效,否则请求 CA 中心重发消息;

(2) 若真实消息(m,X)在 T_1 时刻还没有到达,U_1 向 CA 中心发送一个超时消息,表示对 m 的签名失败。

2.3.2　签名者 $U_i(i\geqslant 2)$

首先计算 $T_i=\Delta T_i+T$,再进行如下操作:

(1) 若 U_{i-1} 的签名消息$(m,(S_{i-1},R_{i-1}))$在时刻 T_i 之前到达,验证操作如下:

①验证 R_{i-1} 和 S_{i-1} 是 E 上的点;

②计算消息 m 的散列值 e,$e=H(m,T)$;

③计算 $X_i=S_{i-1}-eQ$　　(1)

X_i 的坐标为(x_i,y_i),如果 $X_i=O$,则拒绝签名;

④设 $R_{i-1}=(r_i,v_i)$;

⑤当且仅当 $x_i=r_i$ 时接受签名,并对 m 继续进行签名;否则请求 U_{i-1} 重发签名消息,同时,统计其重发次数 p,当 $p>P$ 时(P 为一常数,在系统初始化时设定),终止签名过程。

(2) 若 U_{i-1} 的真实签名消息在 T_i 时刻还没到达,U_i 向 CA 中心发送一个超时信息,表示对 m 的签名失败。

2.3.3　CA 中心

假设对 m 的签名要求在时间 ΔT_0 内完成(其中 $\Delta T_0>\Delta T_N$)。CA 中心先计算 $T_0=\Delta T_0+T$,然后进行如下操作:

(1) 若 U_N 的签名消息$(m,(S_N,R_N))$在 T_0 时刻之前到达,则验证过程如下:

①验证 R_N 和 S_N 是 E 上的点;

②计算消息 m 的散列值 e,$e=H(m,T)$;

③计算 $X_N=S_N-eQ$。X 的坐标为(x,y),如果 $X=O$,则拒绝签名;

④设 $R_N=(r_N,v_N)$;

⑤当且仅当 $x=r_N$ 时接受签名,否则签名无效,请求 U_N 重发签名消息。

(2) 若 U_N 的真实消息在 T_0 时刻还没有到达,则 CA 中心认为本次签名失败。

2.4　正确性证明

定理 1　在椭圆曲线多重数字签名方案中,如果等式 $x_l=r_l$ 成立,则多重数字签名(S_l,R_l)被验证。

证明

$$\because\ f(x)=d+a_1x+a_2x^2+\cdots+a_{n-1}x^{n-1}=\sum_{i=1}^{n}f(x_i)\prod_{j=1,j\neq i}^{n}\frac{x-x_j}{x_i-x_j}$$

$$\therefore\ S_l=\sum_{i=1}^{l}S_i=\sum_{i=1}^{l}\left(f(x_i)e\prod_{j=1,j\neq i}^{n}\frac{-x_j}{x_i-x_j}G+k_iX\bmod n\right)=ef(0)G+X\sum_{i=1}^{l}k_i\bmod n=edG+X\sum_{i=1}^{l}k_i\bmod r_i\ (l=1,2,\cdots,N)$$

$$\therefore\ S_l-eQ=edG+X\sum_{i=1}^{l}k_i-eQ\bmod n=\sum_{i=1}^{l}k_iX\bmod n=(x_l,y_l)$$

$$\because\ R_l=\sum_{i=1}^{l}k_lX=(r_l,v_l)$$

∴ 当且仅当 $x_l=r_l$ 时接受签名。

(下转第 183 页)

谁做的,何时发生的,有什么影响这样一些问题。各个项目组 PM,软件配置管理人员,日本客户等所希望看到的配置状态统计信息的内容可能会有不少的差异。项目组 PM 可能关心当前发布修复了哪些 BUG,分别是什么类型的,是不是都顺利交付了等等;而客户如果心中有杆秤,他可能会更注意实际的工作量等。总之,关注方面千人千面,因人而异。我们要善加利用这个有用的资源。

6　关于管理工具的应用

一些满足软件组织切合实际需求的管理工具可以促使软件配置管理更为合理有效,这是众所周知的;但是要想选择一款完全合适的,的确不是一件容易的事。如果是免费的,功能不很完备的,未必适合公司的管理配置的整个流程;如果是付费的,功能庞大的,可能经过裁剪会制定出合适的,但是购买的价格可想而知。对日软件外包企业现在在中国各大城市遍地开花,不象几年前奇货可居,所以如今,就按日方付给中方的元/人月(工资)来讲也日趋缩小,外包企业可能承接的外包量越来越多,性价比却不见得有所提高,企业盈利也不会有什么质的飞跃。针对这种情况,企业领导也不见得会付巨资去买一些高档管理工具,最好的方法应该就是先用免费工具去实现配置管理中的一些基本功能,然后可以根据实际需要自己去开发一些小工具,要知道对日软件外包企业中虽然很少有配置管理专家,但是却会有了解自身企业需要的管理和技术人才,两者相互补足,随着时间的推移,功能的逐步完善,到时就会通过自己企业的力量开发出一套适合自身流程的工具。这样,既节约了成本投入,又有了合适的工具,更重要的是在企业内部也会使得管理和技术人员得到很好的沟通,更利于开展工作,一举三得,何乐不为。

7　结　论

由于笔者现有的理解基础和思想的局限性,目前对软件配置管理尚不够深入,相信在这一行业的经验的不断累积,又会出现一些新的想法和观念,但是,无论如何,有一点我深信不疑,我们进行软件配置管理并不是为了 CMM 或 CMMI 的评估,也不是人云亦云,随波逐流。我们应该清醒地意识到,对于任何想要发展、壮大、以盈利为目的的企业,软件配置管理是非常有必要的,尽管在具体操作中比较繁琐,费力,但此时千万不能轻言放弃,因为这是必然的过程,到最后公司终将会从中得到不少的启示和具体的效益。

参　考　文　献

[1] 软件配置管理 IBM Rational 技术白皮书,版本 1.1.

[2] 计算机软件配置管理计划规范,GB/T 12505-90.

[3] Capability Maturity Model® Integration (CMMI℠),Version 1.1.

[4] Software Engineering 6 th edition.

[5] 齐治昌、谭庆平、宁洪著,软件工程第二版,2004 年 4 月.

(上接第 165 页)

3　性能分析

3.1　安全性分析

(1) 在本文的有序签名方案中,安全性是基于求解椭圆曲线离散对数问题的困难性。每一个签名者 U_i 有私钥 $k_i \in [1, n-1]$ 和公钥 $X_i = k_i G$。攻击者若想从每个签名者 U_i 的公钥 X_i 中求解私钥 k_i 是很难实现的。同样,攻击者若想从签名者的签名公钥 Q_i 中求解签名私钥 $f(x_i)$ 也是很难实现的。

(2) 在签名者中,若存在不诚实签名者 U_{i-1} 给接受者 U_i 发送伪签名 $(m,(S_{i-1},R_{i-1}))$,U_i 可以通过验证方程式(1)来发现其欺骗行为,并要求 U_{i-1} 重新签名;若不诚实签名者通过提供伪签名的方式来延误时间,企图阻止签名的正常进程,则 CA 中心可以通过签名接受者 U_i 提供的失败信息查找原因,及时处理;若签名者 U_i 在 $p > P$ 次收到的来自签名者 U_{i-1} 的不正确的签名信息,则 U_{i-1} 可能是不诚实签名者或攻击者,此时应终止签名过程。

(3) 在本方案中,由于加入了时间限制,可以抵抗签名重播之类攻击。即使攻击者获得原有的一份签名 $(m,T,(S_i,R_i))$,将时间 T 改为当前时间,但是 S_i 中的 $H(m,T)$ 是无法改掉的,因此这类攻击不会成功。

(4) 方案中,每一位签名者 U_i 只对上一个签名者 U_{i-1} 的签名进行验证,这样可以最大限度地避免签名死锁。所谓签名死锁是指在签名过程中,因为签名系统中总存在一个以上的错误操作者而造成的签名—失败—再签名—再失败的死循环。

(5) 在签名步骤 2.3.2 中,若验证方程式(1)不成立,则在请求 U_{i-1} 重发签名的同时,统计其重发次数 p 并判断,若 $p > P$,则终止签名过程。这样就有效地限制了攻击者的尝试次数,增强了系统的稳定性和安全性。

3.2　效率分析

此方案中,签名验证的任务是交给 N 位签名者分别执行的,这样就形成了一种层次验证结构,简化了签名者之间的验证过程,减少了冗余操作,提高了签名效率。同时,这个方案有效避免了签名死锁,特别是签名人数较多时方案的签名效率是较高的,其优越性随着签名人数的增加而愈发显现。

4　结束语

本文提出了一种适合多重数字签名的改进的椭圆曲线数字签名方案,然后在此基础上提出了椭圆曲线多重数字签名方案,该方案安全性较高,能有效避免签名死锁,减少了冗余操作,提高了签名效率。因而在电子商务、电子政务和军事指挥等领域具有实际的应用前景。

参　考　文　献

[1] Harn L. New digital signature based on discrete logarithm[J]. Electronics Letters 1994,30(5):396~398.

[2] 卢建朱、陈火炎、林飞,"Elgamal 型多重数字签名算法及其安全性[J]",《计算机研究与发展》,2000,37(11).

[3] Harn L. Public-Key Cryptosystem design based on factoring and discrete logarithms[J]. IEEE Proc Compute Digit Tech,1994,141(3):193~195.

[4] Johnson D.,Menezes A. The Elliptic Curve Digital Signature Algorithm (ECDSA). Technical Report CORR 99-34,Dept. of C&O,University of Waterloo,Canada,2000-08.

[5] 吕皖丽、钟诚,"一种安全的椭圆曲线多重数字签名方案",《计算机工程》,2004,30(5):126~128.

[6] 崔新春、张文龙、俞时权,"一种 Elgamal 型有序多重数字签名方案",《计算机工程》,2001,27(10):134~135,154.