第32卷 第19期 计 算 机 工 程 2006年10月
Vol.32 № 19 Computer Engineering October 2006
·博士论文· 文章编号：1000—3428(2006)19—0001—02 文献标识码：A 中图分类号：TP309.3

基于DSA的扩展自证明签名方案

侍伟敏，钮心忻，杨义先，高海英
(北京邮电大学信息安全中心，北京 100876)

摘 要：自证明签名对验证者来说一次仅验证了两个签名，而在PMI系统中，验证者除了要认证用户身份，其中包括两个验证：一个是验证用户的签名，另一个是验证CA颁发的公钥证书，还需要验证AA颁发的属性证书。针对此问题，该文对自证明签名做了一定的扩展，提出了扩展自证明签名ESCS方案，ESCS由验证两个签名扩展到可同时验证3个签名，此后又对ESCS方案做了进一步的扩展，扩展后的ESCS方案可以同时验证多个签名。

关键词：扩展自证明签名；DSA；PMI；PKI

Extend Self-certification Signature Scheme Based on DSA

SHI Weimin, NIU Xinxin, YANG Yixian, GAO Haiying
(Information Security Center, Beijing University of Posts and Telecommunications, Beijing 100876)

【Abstract】 A verifier verifies only both signer's signature using self-certified signatures. In PMI system, verifier sides need verify AC issued by AA besides PKC issued by CA which includes two verificationsm, namely user sign and PKC. To the above problem, this paper proposes a new digital signature scheme called multi-certification signature which extents self-certified signatures and verifies two extented to three certifications once. Then, it extents more multi-certification signature which can verfy multi-signatures once.

【Key words】 Extend self-certification signature; DSA; PMI; PKI

数字签名技术[1]保证了所签消息的完整性、认证性和不可拒认性。签名机制的本质特征是该签名只有通过签名者的私有密钥才能产生，验证方利用签名者的公钥验证签名。如果数字签名是来自不同的实体，而对于同一个验证者来说则需要分别验证，比如在PKI[2]中，对于验证方来说首先要利用签名人公钥证书中的公钥验证对消息的签名，接下来还必须验证CA对签名人公钥证书的签名，所以验证的过程实际上包括两次签名验证。

文献[3]指出：“站在验证人的角度来看，验证两个独立主体的数字签名是一种负担”。他们提出了自证明签名(self-certified signatures，SCS)方案。自证明签名[4,5]对验证者来说仅验证了两个签名，这在实际的应用中仍然不能满足需求，比如在PMI[6]系统中，通常来说，除了要认证用户身份，其中包括两个验证：一个是验证用户的签名，另一个是验证CA颁发的公钥证书，还需要验证AA颁发的属性证书AC。针对此问题，本文对自证明签名做了一定的扩展，提出了扩展自证明签名ESCS方案，ESCS由验证两个签名扩展到可同时验证3个签名，此后又对ESCS方案做了进一步的扩展，扩展后的ESCS方案可以同时验证多个签名。

1 扩展自证明签名(ESCS)方案

SCS是基于Schnorr[7]签名方案， 本文的ESCS则是基于DSA[8,9]签名方案。DSA算法的安全性是基于解离散对数的困难性，具有较大的兼容性和适用性。

1.1 ESCS方案的定义

签名者有一个长期的公私钥对(x_0, y_0) (其中x_0为私钥，y_0为公钥)，还有CA颁发的公钥证书$Cert_S$和AA颁发的属性证书$Cert_A$。ESCS由以下3个算法组成：

(1)密钥生成算法：由签名者的长期公私钥对(x_0, y_0)、公钥证书$Cert_S$和属性证书$Cert_A$产生一个临时的公私钥对(x,y)：$x = f(x_0, Cert_S, Cert_A)$，$y = g(y_0, Cert_S, Cert_A)$，其中$f$，$g$是公开算法。

(2)签名算法：由消息m、$Cert_S$、$Cert_A$和临时的签名私钥x产生ESCS签名$\delta = S_x(m, Cert_S \| Cert_A)$。

(3)验证算法：由消息m、$Cert_S$、$Cert_A$和长期的公钥y_0作为输入参数，输出一个0(无效)或1(有效)值来验证签名的有效性，则需要分以下3个步骤：

1) 计算临时的公钥值$y = g(y_0, Cert_S, Cert_A)$；

2)用临时的公钥y验证多证明签名$V_y(\delta, m, Cert_S \| Cert_A) \stackrel{?}{=} 1$；

3) 检查长期公钥y_0是否与$Cert_S$中的一致，并校验$Cert_A$内容是否正确。

1.2 ESCS方案的描述

假设签名人S有一个长期的公私钥对(x_0, y_0)，其中$y_0 = g^{x_0} \bmod p$。签名人还有由CA颁发的公钥证书PKC和AA颁发的属性证书AC。CA的公私钥对、AA的公私钥对和签名人S的公私钥对使用同样的DSA公钥系统，即参数p，q和g是一样的。假设CA的公私钥对是(x_{CA}, y_{CA})，其中$y_{CA} = g^{x_{CA}} \bmod p$。由CA准备的用户证书信息是$CI_S$，$CI_S$由证书序列号、用户长期公钥$y_0$、用户身份、CA的身份等信息组成。当颁发公钥证书时，CA产生一个小于q的随机数k_c，

基金项目：国家自然科学基金资助项目(60372094)
作者简介：侍伟敏(1978－)，女，博士生，主研方向：网络与信息安全；钮心忻，副教授；杨义先，博士、教授、博导、长江学者特聘教授；高海英，博士
收稿日期：2005-10-27 **E-mail：**jsj96@163.com

则用户 S(签名人)的公钥证书就是：

$Cert_S=\{CI_S,(r_c,s_s)\}=\{CI_S,((g^{k_c} \bmod p) \bmod q,(k_c^{-1}(H(CI_S)+$
$+x_{CA}r_c)) \bmod q)\}$，S通过计算下式来验证其公钥证书的有效性：$w_c=s_c^{-1} \bmod q, u_{c1}=(H(CI_S)\times w_c) \bmod q, u_{c2}=(r_c w_c) \bmod q$，$v_c=((g^{u_{c1}}\times y_{CA}^{u_{c2}}) \bmod p) \bmod q$，如果 $v_c=r_c$，则签名有效。另外假设 AA 的公私钥对是 (x_{AA},y_{AA})，其中 $y_{AA}=g^{x_{AA}} \bmod p$。由AA准备的属性证书信息是 CI_A，CI_A 由证书序列号、用户身份，角色、权限、AA的身份等信息组成。当颁发属性证书时，AA产生一个小于q的随机数 k_A，则用户 S(签名人)的属性证书就是：

$Cert_A=\{CI_A,(r_A,s_A)\}=\{CI_A,((g^{k_A} \bmod p) \bmod q,(k_A^{-1}(H(CI_A)+$
$+x_{AA}r_A)) \bmod q)\}$，S 通过计算下式来验证其属性证书的有效性：$w_A=s_A^{-1} \bmod q$，$u_{A1}=(H(CI_A)\times w_A) \bmod q$，$u_{A2}=(r_A w_A) \bmod q$，$v_A=((g^{u_{A1}}\times y_{AA}^{u_{A2}}) \bmod p) \bmod q$，如果 $v_A=r_A$，则签名有效。属性证书 AC 没有独立的公私钥对，但是 AC 是针对某个 PKC 发布的，它们之间有着必然的联系，因此 PKC 和 AC 可以共享同一个 PKC 的公私钥对。于是一个签名者则会签名一个消息，既包括 PKC 又包括 AC。

以下分密钥生成、签名和验证 3 部分来描述对消息 m 的扩展自证明签名，在此期间签名人利用了他的公钥证书信息和属性证书信息。

(1)密钥生成：签名人 S 利用他的长期公私钥对 (x_0,y_0)、CA 颁发的公钥证书和 AA 颁发的属性证书计算出他的 ESCS 密钥对 (x, y)，即 $x=x_0(u_{c2}+u_{A2}) \bmod q$，$y=y_0^{(u_{c2}+u_{A2})}g^{u_{c1}}y_{CA}^{u_{c2}}r_c^{-1}g^{u_{A1}}y_{AA}^{u_{A2}}r_A^{-1} \bmod p$，其中 $w_c=s_c^{-1} \bmod q$，$u_{c1}=(H(CI_S)\times w_c) \bmod q$，$u_{c2}=(r_c w_c) \bmod q$；$w_A=s_A^{-1} \bmod q$，$u_{A1}=(H(CI_A)\times w_A) \bmod q$，$u_{A2}=(r_A w_A) \bmod q$。

(2)签名：使用 ESCS 密钥 x，S 计算出他对消息 m(其中 m 是 $m\|CI_s\|CI_A$ 消息的合并)的 ESCS 签名 $\delta=(r,s)$，方法如下：S 产生一个小于 q 的随机数 k，并计算：$r=(g^k \bmod p) \bmod q$ 和 $s=(k^{-1}(H(m\|CI_S\|CI_A)+xr)) \bmod q$，则 $\{\{r,s),CI_S,CI_A,r_A,r_c,s_c,s_A\}$ 就是验证人收到的 S 关于消息 m 的签名。

(3)验证：验证人通过以下 3 步验证扩展自证明签名 $\{\{r,s),CI_S,CI_A,r_A,r_c,s_c,s_A\}$ 的有效性：

1) 计算出签名人 S 的 ESCS 公钥，$y=y_0^{(u_{c2}+u_{A2})}g^{u_{c1}}y_{CA}^{u_{c2}}r_c^{-1}g^{u_{A1}}y_{AA}^{u_{A2}}r_A^{-1} \bmod p$，其中 $w_c=s_c^{-1} \bmod q$，$u_{c1}=(H(CI_S)\times w_C) \bmod q$，$u_{c2}=(r_c w_c) \bmod q$，$w_A=s_A^{-1} \bmod q$，$u_{A1}=(H(CI_A)\times w_A) \bmod q$，$u_{A2}=(r_A w_A) \bmod q$。

2)用 ESCS 的公钥 y 计算下式来验证签名(r,s)，即 $w=s^{-1} \bmod q$，$u_1=(H(m\|CI_S\|CI_A)\times w) \bmod q$，$u_2=(rw) \bmod q$，$v=((g^{u_1}\times y^{u_2}) \bmod p) \bmod q$，如果 v=r，则签名有效。

3)检查长期公钥 y_0 是否与 $Cert_S$ 中的一致，并核对 $Cert_A$ 内容是否正确。

2 ESCS 的进一步扩展

在 PMI 的环境中，一个用户可能拥有多个属性证书，对于验证方来说，除了认证用户身份，还需要同时验证这多个属性证书才能确认此用户的访问权限是否是合法的，为了解决这个问题，本文又对 ESCS 做进一步扩展，扩展后的 ESCS 便可以同时验证多个证书。

2.1 定义上的扩展

签名者有一个长期的公私钥对 (x_0,y_0)，$(c_1,c_2,\ldots,c_n)$ 为 n 个与公钥相连证书信息。比如 PKC_S、CRL_S、AC_S 等。扩展后的多证明签名(MCS)由以下 3 个算法组成。

(1)密钥生成算法：由签名者的长期公私钥对 (x_0,y_0) 和 n 个 $(c_1,c_2,\ldots,c_n)$ 证书信息产生一个临时的公私钥对 (x,y)：$x=f(x_0,c_1,c_2,\ldots,c_n)$，$y=g(y_0,c_1,c_2,\ldots,c_n)$，其中 f，g 是公开算法。

(2)签名算法：由消息 m、n 个 $(c_1,c_2,\ldots,c_n)$ 证书信息和临时的签名私钥 x 产生多证明签名 $\delta=S_x(m,c_1,c_2,\ldots,c_n)$。

(3)验证算法：由消息 m、n 个 $(c_1,c_2,\ldots,c_n)$ 证书信息和长期的公钥 y_0 作为输入参数，输出一个 0(无效)或 1(有效)值来验证签名的有效性，则需要分以下 3 个步骤：

1)计算临时的公钥值 $y=g(y_0,c_1,c_2,\ldots,c_n)$。

2)用临时的公钥 y 验证多证明签名 $V_y(\delta,m,c_1,c_2,\ldots,c_n)\overset{?}{=}1$。

3)核对 n 个 $(c_1,c_2,\ldots,c_n)$ 证书信息的内容是否正确。

2.2 扩展后的实现

以下分密钥生成、签名和验证 3 部分来描述进一步扩展后的 ESCS 方案：

(1) 密钥生成：签名人 S 利用他的长期公私钥对 (x_0,y_0)、CA 颁发的公钥证书和 AA 颁发的属性证书计算出扩展后的 ESCS 密钥对 (x,y)，即 $x=x_0(u_{12}+u_{22}+\cdots+u_{n2}) \bmod q$，$y=y_0^{(u_{12}+u_{22}+\cdots+u_{n2})}g^{u_{11}}y_1^{u_{12}}r_1^{-1}g^{u_{21}}y_2^{u_{22}}r_2^{-1}\ldots g^{u_{n1}}y_n^{u_{22}}r_n^{-1} \bmod p$，其中 $w_1=s_1^{-1} \bmod q, u_{11}=(H(CI_1)\times w_1) \bmod q, u_{12}=(r_1 w_1) \bmod q$，$w_2=s_2^{-1} \bmod q, u_{21}=(H(CI_2)\times w_2) \bmod q, u_{22}=(r_2 w_2) \bmod q$，…$w_n=s_n^{-1} \bmod q$，$u_{n1}=(H(CI_n)\times w_n) \bmod q$，$u_{n2}=(r_n w_n) \bmod q$。

(2) 签名：用扩展后的 ESCS 密钥 x，S 计算出他对消息 m(m 为 $m\|CI_1\|CI_2\|\ldots\|CI_n$ 消息的合并)的签名 $\delta=(r,s)$，方法如下：S 产生一个小于 q 的随机数 k，并计算：$r=(g^k \bmod p) \bmod q$ 和 $s=(k^{-1}(H(m\|CI_1\|CI_2\|\ldots\|CI_n)+xr)) \bmod q$，则 $\{\{r,s),CI_1,CI_2,\ldots CI_n,s_1,s_2\ldots s_n,r_1,r_2\ldots r_n\}$ 就是验证人收到的 S 对消息 m 签名。

(3) 验证：验证人通过以下 3 步验证其签名 $\{\{r,s),CI_1,CI_2,\ldots,CI_n,s_1,s_2\ldots s_n,r_1,r_2\ldots r_n\}$ 的有效性：

1) 计算出签名人 S 扩展后的 ESCS 公钥，$y=y_0^{(u_{12}+u_{22}+\cdots+u_{n2})}g^{u_{11}}y_1^{u_{12}}r_1^{-1}g^{u_{21}}y_2^{u_{22}}r_2^{-1}\cdots g^{u_{n1}}y_n^{u_{22}}r_n^{-1} \bmod p$，其中 $w_c=s_c^{-1} \bmod q, w_1=s_1^{-1} \bmod q, u_{11}=(H(CI_1)\times w_1) \bmod q$，$u_{12}=(r_1 w_1) \bmod q$，$w_2=s_2^{-1} \bmod q$，$u_{21}=(H(CI_2)\times w_2) \bmod q$，$u_{22}=(r_2 w_2) \bmod q$，… $w_n=s_n^{-1} \bmod q$，$u_{n1}=(H(CI_n)\times w_n) \bmod q$，$u_{n2}=(r_n w_n) \bmod q$。

2)用扩展后的 ESCS 公钥 y 计算下式来验证签名 (r,s)，即 $w=s^{-1} \bmod q$，$u_1=(H(m\|CI_1\|CI_2\|\ldots\|CI_n)\times w) \bmod q$，$u_2=(rw) \bmod q$，$v=((g^{u_1}\times y^{u_2}) \bmod p) \bmod q$，如果 $v=r$，则签名有效。

3)核对 n 个 $(c_1,c_2,\ldots,c_n)$ 证书信息的内容是否正确。

3 结论

扩展后的自证明签名方案大大减轻了验证方的负担，特别是需要验证多个签名时优势更加明显。尽管签名人的负担

(下转第 5 页)