JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 22, 965-973 (2006)

Short Paper

Design of Proxy Signature in the Digital Signature Algorithm (DSA)*

I-TE CHEN, MING-HSIN CHANG+ AND YI-SHIUNG YEH
Department of Computer Science and Information Engineering
National Chiao Tung University
Hsinchu, 300 Taiwan
+*Chunghwa Telecom Laboratories*
Taoyuan, 326 Taiwan

The use of proxy signature has attracted considerable attention and has been deployed in many applications. Unfortunately, most of the proxy signature schemes, which have been proposed up to now, are not based on standard signature such as Digital Signature Algorithm (DSA). Consequently, those proposals have inevitably been considered infeasible because of obvious security weaknesses so they suffer from new attacks. In fact, with the application of standard signature like DSA, which has been fully discussed, a proxy signature scheme could be well accepted for the deployment in many applications. Therefore, we propose a new proxy signature scheme which combines the DSA properties with proxy signatures. In other words, our scheme keeps not only the properties of the DSA but also fulfills the requirements of proxy signatures. We hope our new proxy signature based on DSA combining Public-key infrastructure (PKI) mechanism could be used in practice as well as the conventional DSA.

Keywords: proxy signature, DSA, PKI, cryptography, indistinguishable

1. INTRODUCTION

Digital signatures are pervasive in the electronic processes in order to replace handwritten signature in modern world. The Digital Signature Algorithm (DSA) based on ElGamal [1] and Schnorr's [8] signature schemes is a useful digital signature scheme and has become a U.S. Federal Information Process Standard (FIPS 186) in August of 1991; called as the Digital Signature Standard (DSS) [6]. Furthermore, the Elliptic Curve Digital Signature Algorithm (ECDSA), which was also accepted as FIPS standard (FIPS 186-2) in 2000, is based on DSA and the Elliptic curve cryptosystems (ECC) proposed in 1985.

In recent years, the proxy signature scheme, which was proposed in 1996 [4] by Mambo, Usuda and Okamoto, is variant digital signature scheme and has been widely

Received May 19, 2004; revised February 17 & May 10, 2005; accepted June 20, 2005.
Communicated by Ja-Ling Wu.
* This paper was partially supported by the National Science Council of Taiwan, R.O.C., under grant No. NSC 93-2213-E-009-033.

discussed [2-4, 9]. In proxy signature scheme, an original signer delegates its signing capability to a proxy signer, and the proxy signer creates a digital signature on behalf of the original signer. Actually, most of the proposed proxy signature schemes are not feasible in practice [3, 4, 9] because the security of those schemes cannot be really proved without adopting standard signature like DSA. Most of them are not strong, secure, and unbreakable sufficiently in order to against some unknown intentional attacks; in addition, they are not base on standard signature. To conquer those disadvantages, therefore, we are the first one that propose proxy-protected signature scheme [4] combining standard signature DSA [6] which is pretty well-known by their security properties to reinforce the proxy signature. Combining DSA, proxy signature and PKI mechanism, this work could be used in practice.

In addition to the requirements of digital signatures, our proposed scheme conforms to at least the following requirements [3, 4, 10]:

(i) **Unforgeability:** Only the designated proxy signer can create a valid proxy signature on behalf of the original signer.
(ii) **Verifiability:** From a proxy signature, a verifier can be convinced that the original signer agrees on signing the message.

2. PRELIMINARIES

The proposed scheme is related to Manbo's proxy signature scheme [4]. Hence, we will briefly describe the basic protocol of the Mambo's scheme at first. The participants are an original signer and a proxy signer. Let p be a prime number and q be a prime division of $p - 1$. g is an element of order q in Z_p^*. The tuple (p, q, g) is public and the basic protocol of the Mambo's scheme uses the following algorithms:

Key generation The original signer selects a random number $x \in Z_q^*$ as the private key and the corresponding public key is $y = g^x \bmod p$. Then, the original signer publishes (p, q, g, y).

Proxy key generation The original signer should do following steps:

1. Select a random number $k_A \in Z_q^*$.
2. Compute $r_A = g^{k_A} \bmod p$, and sets $s_A = (x + k_A r_A) \bmod q$.
3. Forward (r_A, s_A) to the proxy signer.

On receiving the (r_A, s_A), the proxy signer should verify the validity by checking equation $g^{s_A} = y r_A^{r_A} \bmod p$. The proxy signer accepts s_A, if the equation holds, and continues following steps. Moreover, the proxy key is s_A.

Proxy signature The proxy signer can sign a message m on behalf of the original signer to create a signature $S(s_A, m)$ using the proxy key s_A.

Proxy signature verification The verification of proxy signatures is carried out by

using the implicit public key $g^{s_A} = yr_A^{r_A} \bmod p$ to replace public key in the verification process. The verification is to check if $V(yr_A^{r_A}, S(s_A, m), m) \stackrel{?}{=} True$.

The original signer creates a proxy key S_A alone in Mambo's scheme. However, this scheme is a proxy-unprotected proxy signature scheme [4] in which the original signer knows and can derive the proxy key on his/her own. On the contrary, in the proxy-protected proxy signature scheme [4], the original signer and proxy signer creates the proxy key interactively so that the original signer cannot derive the proxy key on his/her own. In addition, our scheme is also a kind of proxy-protected proxy signature schemes.

3. THE PROPOSED SCHEME

The parameter (p, q, g) is defined as the above section. The Secure Hash Algorithm (SHA-serials) [7] and Public-key infrastructure (PKI) mechanism [10, 11] are used in our scheme; the length l of p is a multiple of 64 and $512 \le l \le 1024$, and q is a 160-bit prime. Moreover, the participants of the proposed scheme include an original signer 'Alice', a proxy signer 'Bob', and a verifier. PKI provides an authentication technology to identify parties. Suppose that Alice is certified by a Certificate Authority (CA). And Bob enrolls proxy key into the PKI when a proxy key is created with the original signer interactively.

Alice has a private key x and public key (p, q, g, y) that is the same as the key generation in above section. At initialization step, the CA or Registration Authority (RA) verifies the relationship of the delegation. Besides, we use X.509v3 certificate extension to indicate the relationship between an original signer and the proxy signer by proxy parameters. The PKI mechanism can avoid man-in-middle attack [5].

3.1 Proxy Key Generation (Executed by Alice and Bob)

1. Bob selects a random $\sigma \in Z_q^*$, where gcd (σ, $p - 1$) = 1 and computes $g' = g^{\sigma} \bmod p$. Then, Bob sends g' to Alice.
2. After receiving g', Alice selects a random $k_A \in Z_q^*$, computes $r_A = g^{k_A} \bmod p$, and sets $e = h(g'^{k_A}) \bmod p$ and $s_A = (xe + k_A) \bmod q$. Then, Alice then sends (r_A, s_A) to Bob. The pair (r_A, s_A) is a delegation proxy certification for proving that Alice delegates her signing capacity to Bob.
3. On receiving (r_A, s_A), Bob computes $e' = h(r_A^{\sigma}) \bmod p$ and verifies the validity by checking if $r_A = g^{s_A} y^{-e'} \bmod p$.
4. If the equation $r_A = g^{s_A} y^{-e'} \bmod p$ holds, Bob sets $s_B = s_A \sigma^{-1} \bmod q$ as a proxy key, sets (s_B, $g^{s_B} \bmod p$) as public key pairs and sends the certificate request [12] to the RA.
5. According to certificate policy, RA identifies Bob and then forwards the certificate request to the CA for signing proxy certificate. The process of proxy certificate generating is shown in Fig. 1.

3.2 Proxy Signature (Executed by Bob)

To sign a message m, Bob should do the following steps:

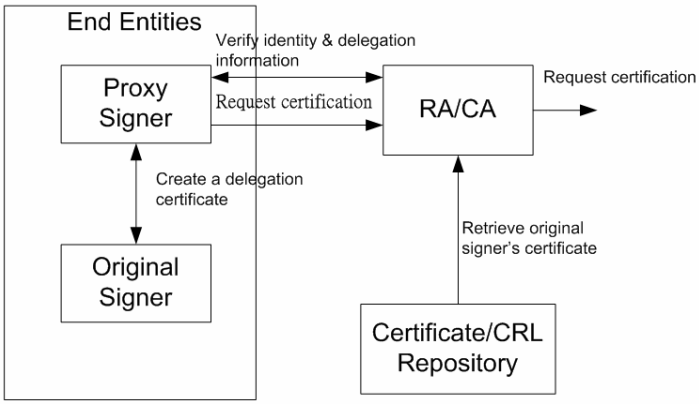

Fig. 1. Proxy signer initialization in PKI.

1. Select a random $k \in Z_q^*$.
2. Compute $r = (g'^k \bmod p) \bmod q$.
3. Set $s = k^{-1} (h(m) + s_B r) \bmod q$.

The proxy signature is the tuple (g', r_A, e', r, s).

3.3 Proxy Signature Verification

To verify the proxy signature (g', r_A, e', r, s) on message m, a verifier should:

1. Query repository and check if the certificate of proxy key is valid.
2. Verify that $1 \le r \le q$ and $1 \le s \le q$; if not holds, reject the signature.
3. Compute $w = s^{-1} \bmod q$.
4. Compute $u_1 = w \cdot h(m) \bmod q$, $u_2 = rw \bmod q$, and $u_3 = e'u_2 \bmod q$.
5. Compute $v = (g'^{u_1} r_A^{u_2} y^{u_3} \bmod p) \bmod q$.

Accept the signature if $v = r$.

3.4 Compatible to Conventional DSA

We consider that the proposed scheme can be deployed in both the DSA with proxy signature capability and the conventional DSA. The proposed scheme can be a conventional DSA if taking the parameters $g' = g$, $r_A = 1$ and $e' = 1$. Therefore, the proposed scheme is generalized DSA and can also be used in conventional DSA.

3.5 Correctness of Proposed Scheme

In this section, we will prove the correctness of the proposed scheme and also prove that the proposed scheme can fulfill the requirements of the proxy signature schemes. Firstly, if the delegation certification (r_A, s_A) is valid, it will pass the verification $r_A = g^{s_A} y^{-e'} \bmod p$.

Proof:

$s_A = (xe + k_A) \bmod q$;

where $e = h(g'^{k_A}) = h((g^{\sigma})^{k_A}) = h((g^{k_A})^{\sigma}) = h(r_A^{\sigma}) = e' \bmod p$.

Substitute e' for e, then we obtain

$s_A = (xe' + k_A) \bmod q$.

Rearrange the above equation as

$k_A = (s_A - xe') \bmod q$.

Raise both sides by g

$g^{k_A} = g^{(s_A - xe')} \bmod p$,
$r_A = (g^{s_A} \cdot g^{-xe'}) \bmod p$ ($\because r_A = g^{k_A} \bmod p$)
$r_A = (g^{s_A} \cdot y^{-e'}) \bmod p$ ($\because y = g^x \bmod p$)

Thus, $r_A = (g^{s_A} \cdot y^{-e'}) \bmod p$ as required. ❑

Secondly, if the proxy signature is generated by the proxy signer correctly, it will pass the proxy signature verification.

Proof:

We have a valid proxy signature $s = k^{-1}(h(m) + s_B r) \bmod q$.

Rearrange the signature as

$k = s^{-1}(h(m) + s_B r) \bmod q$
$k = s^{-1}(h(m) + s_A \sigma^{-1} r) \bmod q$. ($\because s_B = s_A \sigma^{-1} \bmod q$)
$k = s^{-1}[h(m) + (xe + k_A)\sigma^{-1} r] \bmod q$. ($\because s_A = (xe + k_A) \bmod q$)

Raise both sides by g'

$g'^k = (g'^{s^{-1}h(m)} g'^{k_A \sigma^{-1} r s^{-1}} g'^{xe\sigma^{-1} r s^{-1}} \bmod p) \bmod q$.

Substitute following notations respectively:

$g'^k = r$, $g'^{k_A \sigma^{-1}} = g^{k_A} = r_A$ and $g'^{x\sigma^{-1}} = g^x = y$ ($\because g' = g^{\sigma} \bmod p$)
$r = (g'^{s^{-1}h(m)} r_A^{rs^{-1}} y^{ers^{-1}} \bmod p) \bmod q$.

Let $w = s^{-1} \bmod q$, $u_1 = w \cdot h(m) \bmod q$, $u_2 = rw \bmod q$, and $u_3 = e' u_2 \bmod q$.
We yield the equation:

$r = (g'^{u_1} r_A^{u_2} y^{u_3} \bmod p) \bmod q$ as required. ❑

A verifier has to use both the original signer's public key and proxy key certificate to verify the proxy signature. Since the proxy key is created interactively between original signer and proxy signer, a verifier can be aware of the agreement upon signing the message from the original signer. This property obeys the definition of *verifiability*.

4. SECURITY ANALYSIS AND COMPARISONS

The security of the proposed scheme is based on the difficulty of breaking the one-way hash function as well as the hardness of two discrete logarithm problems. One of the discrete logarithms is in Z_q^* where the powerful index-calculus methods apply; the other is in the cyclic subgroup of order q [5]. In this section, we will explain how the proposed scheme resists some possible attacks.

Attack 1: An attacker might forge the proxy signature on the message m by selecting a random k and computing $r = g'^k \bmod p$.

Analysis of Attack 1: The attacker needs proxy key $s_B = \sigma^{-1}(xe + k_A) \bmod q$, k to forge signature $s = k^{-1}(h(m) + s_B r) \bmod q$. It is computationally infeasible to determine s without s_B and correct k under the assumption of the discrete logarithm problem [5]. In addition, the probability of successfully guesses s_B and correct k is $1/q$ which is negligible when q is large enough. Furthermore, the attacker does not either have proxy certificate to pass verification.

Attack2: Another malicious signer impersonates the authorized proxy signer to create a proxy key with an original signer interactively (man-in middle attack).

Analysis of attack 2: The malicious signer may select a random σ and creates a proxy key with an original signer interactively. But, this proxy key must be enrolled into PKI mechanism that RA will reject licensing certificate on this proxy key because the proxy key is not created by the authorized proxy signer. Without this proxy certificate, the proxy signature signed by any malicious signer cannot pass the verification phase naturally.

Attack 3: A dishonest original signer attempt to forge the proxy key.

Analysis of attack 3: The proxy signer uses a blind factor σ to blind $g' = g^{\sigma} \bmod p$ so that the original signer needs to solve σ from $g' = g^{\sigma} \bmod p$. It is difficult to determine σ according to the hardness of the discrete logarithm problem [5]. For the security reason, so-called '*proxy-protected*' property, an original signer should not be able to derive the authorized proxy signer's proxy key; otherwise a verifier could not distinguish exactly whether the original signer or the proxy signer creates the proxy signature.

Attack 4: A malicious proxy signer attempts to impersonate an original signer to create a delegation certification.

Analysis of attack 4: The malicious proxy signer selects a random k_A and computes $r_A = g'^{k_A} \bmod p$, and $e' = h(g'^{k_A}) \bmod p$. However, the malicious proxy signer does not know what the original signer's private key x is to create s_A. Therefore, to solve $r_A = g'^{s_A} y^{-e'} \bmod p$ under just knowing original signer's public key y, g' and r_A is still out of question.

After the proxy signer Bob receiving a delegate certification (r_A, s_A) correctly from the original signer Alice, he cannot forge another delegate certification to create a proxy key, because it is difficult to find another r_A for create a valid delegation certification. On the other hand, Alice cannot either forge the proxy key, because the generator is blinded by a factor σ which is only known by Bob. Thus, only the authorized proxy signer can create the valid proxy key. Hence, the proposed scheme also confirms the properties of *strong unforgeability* [3] and *proxy-protected*. The proposed scheme is modified conventional DSA and the conventional DSA can be reduced to our proposed scheme in polynomial time. Furthermore, no other schemes base on standard signature DSA, so we show the differences among DSA, Mambo's proxy signature scheme and proposed scheme in Table 1.

Table 1. The differences among DSA, Mambo's proxy signature scheme and proposed scheme.

	Based on Signature	Proxy functionality	Combining with PKI	Standard Signature	Proxy-protected
DSA	ElGamal and Schnorr	No	No	Yes	No
Mambo's Scheme	ElGamal	Yes	No	No	No
Proposed scheme	ElGamal and Schnorr	Yes	Yes	Generalized Standard	Yes

Table 2. The comparison between the time complexity of the proposed scheme and the DSA.

Schemes	Proxy Generation	Proxy Verification	Signature	Verification
The proposed scheme	$241T_m$	$721T_m$	$242T_m + T_{inv}$	$725T_m + T_{inv}$
DSA	N/A	N/A	$242T_m + T_{inv}$	$483T_m + T_{inv}$

Note: T_m: The number of modular multiplications.
T_{inv}: The number of modular inverse with 160-bit modulus.

In the propose scheme, the size of q is 160 bits and the size of p is between 512 and 1024 bits. For the security reason, a 512-bit prime provides marginal security such that at least 786 bits is recommended. Suppose p is a 768-bit integer and one modular exponentiation takes on 240 modular multiplications. In Table 2, we compare the time complexity between the proposed scheme and the DSA. The major portion of time complexity is modular multiplications and modular inverses, thus we neglect the time complexity of

hash function and modular additions. In the propose scheme, the time complexity of the proxy signature is the same as the DSA, while the time complexity of the proxy signature verification requires one modular exponentiation and two modular multiplications more than the DSA.

5. CONCLUSION

The novel proxy signature combining the DSA has been proposed. In the view of proxy signature, the conventional DSA is a special case of the proposed scheme; therefore, the proposed scheme can deploy both the conventional DSA and the proxy DSA and is generalized DSA with proxy signature capability. In addition, the time complexity of proposed scheme only requires one modular exponentiation and two modular multiplications more than the DSA on verification phase. Hence, the proposed scheme can be used on many applications in practice.

REFERENCES

1. T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Transactions on Information Theory*, Vol. IT-31, 1985, pp. 469-472.
2. B. Lee, H. Kim, and K. Kim, "Secure mobile agent using strong nondesignated proxy signature," in *Proceedings of the Australasian Conference on Information Security and Privacy*, LNCS 2119, 2001, pp. 474-486,
3. B. Lee, H. Kim, and K. Kim, "Strong proxy signature and its applications," in *Proceedings of the Symposium on Cryptology and Information Security*, 2001, pp. 603-608.
4. M. Mambo, K. Usuda, and E. Okamoto, "Proxy signatures: delegation of the power to sign messages," *IEICE Transactions on Fundamentals*, Vol. E79-A, 1996, pp 1338-1354.
5. A. Menezes, P. van Oorschot, and S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1996.
6. *Digital Signature Standard* (*DSS*), National Institute for Standards and Technology, Federal Register, 56:169, 1991.
7. *Secure Hash Signature Standard* (*SHS*), National Institute of Standards and Technology, FIPSP 180-2, 2002.
8. C. P. Schnorr, "Efficient signature generation for smart cards," *Advances in Cryptology – Crypto '89*, Springer-Verlag, 1990, pp. 239-252.
9. K. Shum and Victor K. Wei, "A strong proxy signature scheme with proxy signer privacy protection," in *Proceedings of 11th IEEE International Workshops on Enabling Technologies Infrastructure for Collaborative Enterprises*, 2002, pp. 55-56.
10. C. Adms and S. Farrell, "Internet X.509 public key infrastructure certificate management protocols," RFC 2510, 1999.
11. S. Chokhan, W. Ford, R. Sabett, C. Merill, and S. Wu, "Internet X.509 public key infrastructure certificate policy and certification practices framework," RFC 3647, 2003.

12. M. Nystrom and B. Kaliski, "PKCS #10: Certification request syntax specification version 1.7," RFC 2986, 2000.

I-Te Chen (陳以德) received a M.S. in Computer Science and Information Engineering (CSIE) from National Chiao Tung University (NCTU), Taiwan in 1997. Since 2001 he has been the Assistant Researcher in Chunghwa Telecommunication Laboratories. Now he is a doctorial candidate of CSIE, NCTU. His research interests include cryptography, electronic commerce, network security and watermark.

Ming-Hsin Chang (張明信) received a M.S. and a Ph.D. degree in Engineering Science form National Cheng Kung University and Computer Science and in Information Engineering from National Chiao Tung University, Taiwan in 1990 and 2005, respectively. Since 1991 he has been the Assistant Researcher in Chunghwa Telecommunication Laboratories. His research interests include information security and network management.

Yi-Shiung Yeh (葉義雄) received a M.S. and a Ph.D. degrees in Department of Electrical Engineering and Computer Science from University of Wisconsin-Milwaukee, U.S.A. in 1980 and 1986, respectively. He is currently a professor in the department of Computer Science and Information Engineering, National Chiao Tung University. His research interests include multi-valued logical systems, genetic algorithms, reliability, game theory, information security/cryptography, and information/coding theory.