A Modified Ohta-Okamoto Digital Signature for Batch Verification and Its Multi-Signature Version

[1]Ting-Yi Chang, [*2]Min-Shiang Hwang, [3]Wei-Pang Yang, [4]Kuo-Cheng Tsou
[1]Department of Industrial Education and Technology, National Changhua University of Education,
tychang@cc.ncue.edu.tw
*[*2]Department of Computer Science and Information Engineering, Asia University,*
mshwang@nchu.edu.tw
[3]Department of Information Management, National Dong Hwa University,
wpyang@mail.ndhu.edu.tw
[3] Graduate Institute of e-Learning, National Changhua University of Education,
mux05@hotmail.com

Abstract

In order to reduce the multiple-signature verification time quite, some investigations on DSA-type and RSA-type digital signatures have been done to achieve batch verification. However, in this article, we shall propose an alternative type digital signature, which is a modified version of Ohta-Okamoto digital signature scheme, to give an efficient batch verifying multiple-signature scheme instead of verifying one individual digital signature at a time. Furthermore, the propose

d scheme for batch verification is implemented by Java to show its performance, which is much better than Ohta-Okamoto scheme. On the other hand, the modified Ohta-Okamoto scheme is extended to a multi-signature, which allows multiple signers to jointly authenticate a message using a single compact signature.

Keywords: *Cryptograph; Digital Signature; Batch Verifying, Multi-Signature*

1. Introduction

Paperwork is being rapidly digitized as e-mail, electronic commerce, and electronic money, etc. have become more and more widespread [3][4][5][6][17][19][24][27][28]. In many of these new forms of communication, a digital signature is essential. Digital signatures have become more and more important in our modern electronic society because they can accomplish missions like identifying senders, authenticating message contents, preventing denial of message ownership, and protecting ownership. The fast development of digital signatures brings the demand for an efficient method of verifying digital signatures to make sure that they are signed by the same signer. For example, assume a sender is signing t documents for the receiver. The sender needs to generate t digital signatures, and it is often the case that the receiver needs to verify these t digital signatures one at a time. Since modular exponentiation means a very large modulus, it spends a lot of computation time. Verifying each DSA-type [10] or RSA-type [2][16][25] digital signature requires two or one modular exponentiation, respectively. In other words, if there are t DSA-type and t RSA-type digital signatures, the receiver will have to execute $2t$ and t modular exponentiations to verify these digital signatures, respectively, which is very inefficient.

The batch verifying multiple DSA digital signature scheme was first proposed by Naccache et al. [21]. In their scheme, the multiple DSA digital signatures are batch verified by the receiver, and all is done at a time. However, Lim and Lee [18] pointed out that Naccache et al.'s scheme [21] is insecure. Later, Harn [8] proposed a DSA-type secure interactive batch verification protocol. Later, Harn proposed two efficient non-interactive batch verification protocols: DSA-type and RSA-type multiple digital signatures [9]. Unfortunately, Shao [23] showed that the scheme proposed in [8] was not secure.

Recently, Hwang et al. [14] have pointed out in [9] that a dishonest signer can easily forge individual digital signatures and make false batch verifications valid. One of the reasons for attacks to succeed is that the dishonest signer does not make the multiple digital signatures in the original order (See [14] for a more detailed description). The signer would deny her/his having signed the messages, which means it has trouble preventing denial of message ownership. Hwang et al. proposed two simple batch-verifying multiple digital signatures called BV-DSA and BV-RSA schemes [12][13]. They added t serial numbers or weights in the batch verification to withstand the attack. However, Bao et al. [1] showed that the probability for the receiver to be successfully cheated was actually as high as fifty percent in BV-RSA scheme. It would also make it possible for the dishonest signer to forge individual digital signatures and make false batch verifications valid. There are many schemes had been proposed for solving the problem [11][15][26].

In this paper, we shall propose an alternative type of Ohta-Okamoto digital signature [22] based on the Fiat-Shamir scheme [7]. In order to achieve batch verification, we shall modify Ohta-Okamoto digital signature scheme and make the new modified scheme strong enough to withstand Hwang et al.'s attack [14] and Bao et al.'s attack [1]. Moreover, this paper will also prove that batch verification has the same security level as the original Ohta-Okatmoto digital signature scheme [22], where signatures are verified individually.

In Section 2, we shall propose our digital signature scheme, which is a modified version of Ohta-Okatmoto digital signature scheme [22]. At the same time, we shall show the batch verifying multiple digital signatures in our scheme. In Section 3, we shall analyze the security of the modified Ohta-Oktamoto digital signature scheme, neutralizing Hwang et al.'s attack [14] and Bao et al.'s attack [1]. In Section 4, we use the software to implement the proposed scheme and evaluate its results. In Section 5, the modified Ohta-Okamoto scheme is extended to a multi-signature version. Finally, we shall give a brief conclusion in Section 6.

2. Modified version of Ohta-Okatmoto digital signature scheme for batch verification

First of all, let's briefly review Ohta and Okat's digital signature scheme [22] as follows. Suppose we have a public integer L and a public modulus n which is the product of two secret large primes p and q. Next, we choose a secret integer $s \in_R Z_n$ as the private key and $y = s^{-L} \bmod n$ as the public key. Assume that a sender Alice wants to send a message m and its signature to some receiver named Bob. She performs the following steps:

Step 1: Choose an integer $r \in_R Z_n$ and compute $u = r^L \bmod n$.

Step 2: Compute $e = h(u, m)$, where $h(\cdot)$ is a public one–way hash function.

Step 3: Compute $z = r \times s^e \bmod n$.

Ohta and Oktamoto's signature of message m is (e, z). Then, Alice sends (e, z) and m to Bob. After receiving the signature, Bob first computes $\bar{u} = z^L \times y^{e} \bmod n$. The correctness of the signature on the message m can be verified by checking $e = h(\bar{u}, m)$.

Here, we make some modifications on Ohat-Oktamoto digital signature scheme as follows. The parameters $\{L, n, p, q, y, s, m, h(\cdot)\}$ remain the same. Alice performs the following steps instead:

Step 1: Choose an integer $r \in_R Z_n$ and compute $u = r^L \bmod n$.

Step 2: Compute $z = r \times s^{h(u, m)} \bmod n$.

The signature of message m is (u, z) in our modified scheme. Then, Alice sends (u, z) and m to Bob. After Bob receives (u, z) and m from Alice, he proves the correctness of the signature on the message m by checking $u = z^L \times y^{h(u, m)} \bmod n$.

Generating Multiple Digital Signatures: Without loss of generality, assume that Alice wants to send t messages $m_1, m_2 \ldots m_t$ and digital signatures $(u_1, z_1), (u_2, z_2), \ldots, (u_t, z_t)$ to Bob, where $u_i = r_i^L \bmod n$ and $z_i = r_i \times s^{h(u_i, m_i)} \bmod n$ $(i = 1, 2, \ldots, t)$ by using the Alice's private key s.

Batch Verifying Multiple Digital Signatures: After receiving these digital signatures $(u_1, z_1), (u_2, z_2), \ldots, (u_t, z_t)$ from Alice, Bob first randomly chooses an integer b and t other integers $w_1, w_2 \ldots w_t$ which are distinct and small. Then, he verifies the correctness of these multiple digital signatures $(u_1, z_1), (u_2, z_2), \ldots, (u_t, z_t)$ on messages $m_1, m_2 \ldots m_t$ by using Alice's the public key y a and the public parameters to check the following equation:

$$\left(\prod_{i=1}^{t} u_i^{w_i}\right)^{b!} = \left(\prod_{i=1}^{t} z_i^{w_i}\right)^{L\times b!} \times \left(y^{\sum_{i=1}^{t} h(u_i,m_i\,)\,\times\, w_i}\right)^{b\,!} \bmod n. \tag{1}$$

If Equation (1) holds, the digital signatures (u_1, z_1), (u_2, z_2), … , (u_t, z_t) on messages m_1, m_2… m_t are correct. When a dispute occurs, say, if Alice wants to deny her having signed the messages, then Bob can show the following equation to prove that the individual digital signatures were actually singed by Alice.

$$u_i = z_i^L \times y^{h(u_i,m_i)} \bmod n, \text{for } i = 1,2,\dots,t. \tag{2}$$

The correctness of Equation (1) can be verified as follows:

$$(\prod_{i=1}^{t} u_i{}^{w_i})^{b!} = (\prod_{i=1}^{t} r_i{}^{L\times w_i})^{b!} \bmod n,$$

and

$$\left(\prod_{i=1}^{t} z_i{}^{w_i}\right)^{L\times b!} \times \left(y^{\sum_{i=1}^{t} h(u_i,m_i)\times w_i}\right)^{b!} \bmod n,$$

$$= \left(\prod_{i=1}^{t} (r_i \times s^{h(u_i,m_i)})^{w_i}\right)^{L\times b!} \times \left(s^{\sum_{i=1}^{t} h(u_i,m_i)\times w_i}\right)^{-L\times b!} \bmod n,$$

$$= (\prod_{i=1}^{t} r_i{}^{L\times w_i})^{b!} \bmod n.$$

The correctness of Equation (2) can be verified as follows:

$$u_i = r_i^L \bmod n,$$

And

$$\begin{aligned} z_i^L \times y^{h(u_i,m_i)} &= \left(r_i \times s^{h(u_i,m_i)}\right)^L \times s^{-L\times h(u_i,m_i)} \bmod n, \\ &= r_i^L \bmod n. \end{aligned}$$

In our scheme, Bob can verify these multiple digital signatures with Alice's public key, and only one verification session is required instead of *t* verification sessions. Hence, batch verification is obviously more efficient than verifying each individual digital signature separately.

3. Security analysis

In this section, we shall first analyze the security of our modified version of Ohta-Oktamoto digital signature scheme (Attack 1-3) and then separately break the attacks proposed by Hwang et al. [14] and Bao et al. [1]. In the following passages, several possible attacks will be investigated to demonstrate the security of our scheme.

Attack 1: The private key s cannot be derived from the public key y and the public parameters n and L.
To derive the private key s from the public key $y = s^{-L}$ mod n, the attacker has to face the intractability of the factorization (FAC) problem.

Attack 2: The private key s cannot be derived from a valid digital signature (u, z).

To compute the private key s from $z = r \times s^{h(u;m)}$ mod n, the attacker should find out what the random value r is. Then, she/he has to calculate the $h(u, m)$ - th root of $(z \times r^{-1})$ mod n. However, the difficulty of retrieving r from u and extracting the $h(u, m)$-th root of $(z \times r^{-1})$ mod n is equivalent to solving the intractability of the FAC problem.

Attack 3: The digital signature (u, z) on message m cannot be forged.

In order to forge the valid digital signature of message m, the attacker first randomly chooses an integer r and computes $u = r^L$ mod n and then compute a hash value $h(u, m)$. To solve z such that $z^L = u \times \left(y^{h(u,m)}\right)^{-1}$ mod n without knowing the private s is equivalent to solving the intractability of the FAC problem. On the other hand, if the attacker randomly chooses a signature (u, z), she/he should find a message m'such that $u = z^L \times\ y^{h(u,m\prime)}$ mod n. However, it is infeasible because $h(\cdot)$ is a collision free one-way function.

Note that the random number r should be kept secret. Otherwise, the attacker can obtain the private key s from two valid digital signatures (u_1, z_1), (u_2, z_2) as follows:

$$s^{h(u_1,m_1)} = r_1^{-1} \times z_1\ m\,od\ n$$

$$s^{h(u_2,m_2)} = r_2^{-1} \times z_2\ m\,od\ n$$

If the hash values $h(u_1, m_1)$ and $h(u_2, m_2)$ are relatively prime, the attacker can use the Euclidean algorithm [20] to reveal the private key s.

Hwang et al.'s Attack 1: In *Hwang et al.'s attack 1,* they assume that the dishonest signer sends the multiple digital signatures but not in the same order as the individual digital signatures are generated. For example, Alice sends three messages m_1, m_2, m_3 and forges signatures (u_1, z_1'), (u_2, z_2') (u_3, z_3') to Bob, where $z_1' = r_1 \times s^{h(u_1,m_2)} \bmod n$, $z_2' = r_2 \times s^{h(u_2,m_3)} \bmod n$, and $z_3' = r_3 \times s^{h(u_3,m_1)}$ mod n. After receiving these, Bob first randomly chooses three integers w_1, w_2, w_3 and an integer b (b is used to withstand Bao et al.'s attack). Then, he proves the correctness of the multiple digital signatures on messages m_1, m_2, m_3 by checking Equation (1) as follows:

$$\left(\prod_{i=1}^{3} u_i^{w_i}\right)^{b!} = ({r_1}^{w_1} \times {r_2}^{w_2} \times {r_3}^{w_3})^{L\times b!} \bmod n,$$

and

$$\left(\prod_{i=1}^{3} z_i'^{w_i}\right)^{L\times b!} \times \left(y^{\sum_{i=1}^{3} h(u_i,m_i)\times w_i}\right)^{b!} \bmod n,$$

$$= (({r_1}^{w_1} \times s^{h(u_1,m_2)\times w_1}) \times ({r_2}^{w_2} \times s^{h(u_2,m_3)\times w_2}) \times\ ({r_2}^{w_2} \times s^{h(u_2,m_3)\times w_3})) \times s^{-L\times h(u_1,m_1)\times w_1} \times s^{-L\times h(u_2,m_2)\times w_2} \times s^{L\times h(u_3,m_3)\times w_3} \bmod n,$$

So

$$\left(\prod_{i=1}^{3} u_i^{w_i}\right)^{b!} \neq \left(\prod_{i=1}^{3} z_i'^{w_i}\right)^{L\times b!} \times \left(y^{\sum_{i=1}^{3} h(u_i,m_i)\times w_i}\right)^{b!} \bmod n.$$

As a result, Equation (1) does not hold, so the dishonest signer cannot forge individual digital signatures to make false batch verifications valid.

Hwang et al.'s Attack 2: In *Hwang et al.'s attack 2,* they assume that the dishonest signer Alice sends messages m_1 , $m_{2,}$ m_3 and forges digital signatures (u'_1, z_1), (u'_2, z_2) (u'_3, z_3) where $u'_1 = a_1\times u_1$, $u'_2 = a_2\times u_{2,}$ $u'_3 = a_3\times u_3$ and $\prod_{i=1}^{t} a_i = 1$ ($a_1 = 1/4$, $a_2 = 8$, $a_1 = 1/2$), to the receiver Bob. Then, Bob checks Equation (1) to verify these multiple digital signatures as follows:

$$\left(\prod_{i=1}^{3} u_i'^{w_i}\right)^{b!} = ((r_1 \times a_1)^{w_1} \times (r_2 \times a_2)^{w_2} \times \ (r_3 \times a_3)^{w_3})^{L\times b!} \times$$

$$\left(\left(r_1 \times \frac{1}{4}\right)^{w_1} \times (r_2 \times 8)^{w_2} \times \left(r_3 \times \frac{1}{2}\right)^{w_3}\right)^{L\times b} \bmod n$$

and

$$(\prod_{i=1}^{3} z_i^{w_i})^{L\times b!} \times \left(y^{\sum_{i=1}^{3} h(u_i,m_i)\times w_i}\right)^{b!} = \left(\prod_{i=1}^{3} r_i^{L\times w_i}\right)^{b!} \bmod n,$$

so

$$\left(\prod_{i=1}^{3} u_i'^{w_i}\right)^{b!} \neq \left(\prod_{i=1}^{3} z_i^{w_i}\right)^{L\times b!} \times \left(y^{\sum_{i=1}^{3} h(u_i,m_i)\times w_i}\right)^{b!} \bmod n.$$

For the same reason, the dishonest signer also cannot make the forged individual digital signatures (u_1, z'_1), (u_2, z'_2) (u_3, z'_3) valid, where $z'_1 = a_1\times z_1$, $z'_2 = a_2\times z_{2,}$ $z'_3 = a_3\times z_3$ and $\prod_{i=1}^{t} a_i = 1$ ($a_1 = 1/4$, $a_2 = 8$, $a_1 = 1/2$), Those forged multiple digital signatures still cannot pass the batch verification in Equation (1). If these signatures can pass the batch verification in Equation (1), they must be correct signatures. Hence, when a dispute occurs, Alice cannot deny her having signed the messages because Equation (2) holds.

Bao et al.'s Attack: Because the factors p and q of n are generated by Alice, she has the ability to compute a value $v^2 = 1 \bmod n$. Assume that two signatures (u'_1, z_1) and (u_2, z_2) separately on message m_1 and $m_{2,}$ are signed by Alice, where $u' = u_1 \times v \bmod n$ and $u_i = r_i^L \bmod n$ and $z_i = r_i \times s^{h(u_i,m_i)}$ $(i = 1, 2)$, Without the value $b!$ in Equation (1), Bob randomly chooses two integers w_1 and w_2 to verify these multiple digital signatures in Equations (1). Unfortunately, the value w_1 Bob chooses is a multiplier of 2. It will make Equation (1) hold. However, when a dispute occurs, Alice can deny her having signed message m_1 because $u'_1 \neq z_1^L \times y^{h(u'_1,m_1)}$. Under the attack, the probability of successful cheating is 1/2. In fact, Alice can arbitrarily choose an integer a such that $v^a = 1 \bmod n$ since she knows the factors p and q. The probability of successful cheating is not 1/2 but $1/a$, where $a \geq 2$. In other words, if the value is larger, the probability of successful cheating is smaller. To ensure the security of our scheme, the integer b should be equal to or greater than a in Equation (1).

From the above analysis, the modified version of Ohta-okamoto digital signature for batch verification strong enough to withstand Hwang et al.'s attacks [14] and Bao et al.'s attack [1].

4. Performance Analysis and Software Implement

This section shows that the computational complexity performances of the proposed scheme and Ohta-Okatmoto scheme for t digital signatures. For facilitating the computational complexity, the following notations are defined.

T_{EXP}	the time for computing a modular exponentiation computation,
T_{MUL}	the time for computing a modular multiplication computation,
T_{ADD}	the time for computing a modular addition computation,
T_{H}	the time for computing a hash value,
T_{EQL}	the time for comparing two values are equal or not,
T_{SEXP}	the time for computing a small modular exponentiation computation.

Table 1. Performance evaluations of the proposed scheme and Ohta-Okatmoto scheme

	verify *t* signatures
Our scheme	$t\times(2T_{\text{EXP}}+1T_{\text{MUL}}+1T_{\text{H}}+T_{\text{EQL}})$
Ohta-Okatmoto scheme	$2T_{\text{EXP}}+2T_{\text{SEXP}}+T_{\text{EQL}}+t\times(2T_{\text{SEXP}}+3T_{\text{MUL}}+1T_{\text{H}}+1T_{\text{ADD}})$

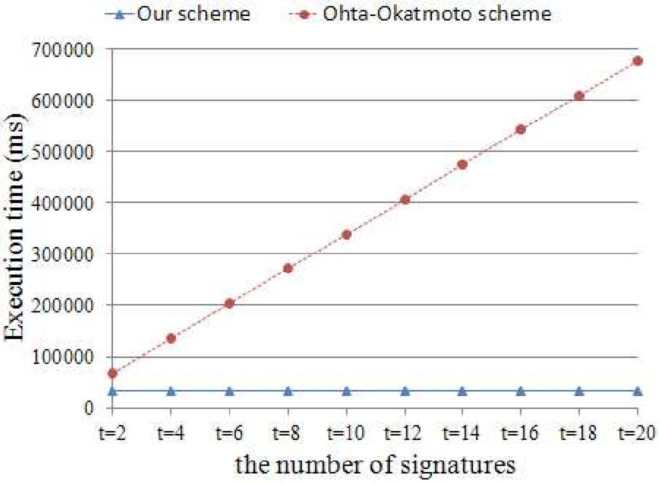

Figure 1. Compared execution time for verifying *t* signatures

According to Table 1, it is obvious that computation of verifying t signatures in our scheme is fewer than that in Ohta-Okatmoto scheme. T_{EXP} is much larger than other computations and we can see that the computations of Ohta-Okatmoto scheme increase by $2t$ whereas our scheme is fixed by a constant 2. The simulations are performed with a simulator in Java. A PC (2.49GHz Pentium 4 with 2GB of RAM) is used to execute the simulations. The key length is 2048-bit and the hash function MD5 is used in the proposed scheme.

Figure 1 is the average of 100 executed results for different numbers of signatures (from t = 2, 4, …, 20) in our scheme and Ohta-Okatmoto scheme. It can be seen that the execution time of our scheme is obviously better than that of Ohta-Okatmoto scheme.

5. Multi-signature

In this section, the modified Ohta-okamoto scheme in Section II is extended to a multi-signature version. Multi-signatures allow multiple signers to jointly authenticate a message using a single compact signature [28]. Without loss of generality, assume that k signers (U_1, U_2, …, U_k) want to sign the message m. The parameters $\{L, n, p, q, m, h(\cdot)\}$. Let $s_i \in_R Z_n$ as U_i's private key and $y = s^{-L}$ mod n as U_i's public key. The k signers perform the following steps to generate a multi-signature.

Step 1: U_i chooses an integer $r_i \in_R Z_n$ and computes $u_i = r_i^{\,L}$ mod n. Then broadcasts r_i to other signers.

Step 2: After receiving r_j from U_j ($j \neq i$), U_i computes $z_i = r_i s_i^{h(u_1,\dots,u_k,m)} \bmod n$.

Step 3: The clerk who can be any signer computes $z = \prod_{i=1}^{k} z_i \mod n$.

After receiving the multi-signature (u_1, u_2, …, u_k, z) on the message m, the correctness of the multi-signature can be verified by using signers' public keys y_1, y_2, …, y_k and the public parameters n and L to check the following equation:

$$\prod_{i=1}^{k} u_i = z^L(\prod_{i=1}^{k} y_i)^{h(u_1,\ldots,u_k,m)} \bmod n. \tag{3}$$

If Equation (3) holds, the multi-signature (u_1, u_2, …, u_k, z) on the message m is correct. The correctness of Equation (3) can be verified as follows:

$$\begin{aligned}\prod_{i=1}^{k} u_i &= z^L\left(\prod_{i=1}^{k} y_i\right)^{h(u_1,\ldots,u_k,m)} \bmod n,\\ &= (\prod_{i=1}^{k} z_i)^L(\prod_{i=1}^{k} s_i)^{-L\cdot h(u_1,\ldots,u_k,m)} \bmod n,\\ &= (\prod_{i=1}^{k} r_i s_i^{h(u_1,\ldots,u_k,m)})^L(\prod_{i=1}^{k} s_i)^{-L\cdot h(u_1,\ldots,u_k,m)} \bmod n,\\ &= (\prod_{i=1}^{k} r_i)^L \bmod n,\\ &= \prod_{i=1}^{k} u_i \bmod n.\end{aligned}$$

If t multi-signatures are generated, the batch verification is the same as that in Section 2. Therefore, the modified Ohta-Okatmoto multi-signature scheme for batch verification is still efficient.

6. Conclusion

In this article, we have proposed an alternative type of digital signature based on Ohta-Okatmoto scheme to achieve batch verification. In our scheme, the verifier can only verify multiple digital signatures at the cost of one verification session rather than verifying each individual digital signature separately. Moreover, our scheme has the same security level as the original Ohta-Okatmoto digital signature scheme, and it can withstand Hwang et al.'s attack and Bao et al.'s attack. From the results of software implement, it can be seen that the proposed is much more efficient than Ohta-Okatmoto scheme. The proposed scheme can be easily extended to a multi-signature scheme. Multiple multi-signature signatures can be also performed batch verification efficiently.

7. Acknowledgment

The authors wish to thank many referees for their suggestions to improve this paper. Part of this research was supported by the National Science Council, Taiwan, R.O.C., under contract nos. NSC100-2221-E-018-025 and NSC100-2622-E-018-004-CC3. Partial results have been published in ICNIT 2012.

8. References

[1] G. Eason, B. Noble, and I. N. Sneddon, “On certain integrals of Lipschitz-Hankel type involving products of Bessel functions,” Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955.

[2] C. C. Chang and M. S. Hwang, “Parallel computation of the generating keys for RSA cryptosystems,” IEE Electronics Letters, vol. 32, no. 15, 1996, pp. 1365-1366.

[3] T. Y. Chang, “A Convertible Multi-Authenticated Encryption scheme for group communications,” *Information Sciences*, vol. 178, no.17, 2008, pp. 3426-3434.

[4] T. Y. Chang, “An ID-Based Group-Oriented Decryption Scheme Secure against Adaptive Chosen-Ciphertext Attacks,” Computer Communications, vol. 32, no. 17, 2009, pp. 1829-1836.

[5] T. Y. Chang, “An ID-based Multi-signer Universal Designated Multi-verifier Signature Scheme,” Information and Computation, vol. 209, no. 7, 2011, pp. 1007-1015.

[6] He Du, Jian Wang, Yanan Liu, “A New Anonymous but Accountable Secure Proxy Signature Scheme”, International Journal of Digital Content Technology and its Applications, vol. 6, no. 9, 2012, pp. 204-210.

[7] A. Fiat and A. Shamir, “How to prove yourself: Practical solutions to identification and signature problems,” in Advances in Cryptology, CRYPTO'86, 1986, pp. 186-194, Lecture Notes in Computer Science.

[8] L. Harn, “DSA type secure interactive batch verification protocol,” Electronics Letters, vol. 31, no. 4, 1995, pp. 257-258.

[9] L. Harn, “Batch verifying multiple RSA digital signatures,” Electronics Letters, vol. 34, no. 12, 1998, pp. 1219-1220.

[10] M. S. Hwang, C. C. Chang, and K. F. Hwang, “An ElGamal-like cryptosystem for enciphering large messages,” IEEE Transactions on Knowledge and Data Engineering, vol. 14, no. 2, 2002, pp. 445-446.

[11] M. S. Hwang, C. C. Chang, and K. F. Hwang, “An ElGamal-like cryptosystem for enciphering large messages,” IEEE Transactions on Knowledge and Data Engineering, vol. 14, no. 2, 2002, pp. 445-446.

[12] M. S. Hwang, C. C. Lee, and Eric Jui-Lin Lu, “Cryptanalysis of the batch verifying multiple DSA-type digital signatures,” Pakistan Journal of Applied Sciences, vol. 1, no. 3, 2001, pp. 287-288.

[13] M. S. Hwang, C. C. Lee, and Y. L. Tang, “Two simple batch verifying multiple digital signatures,” Lecture Notes in Computer Science (Proceedings of Information and Communications Security), vol. 2229, 2001, pp. 233-237.

[14] M. S. Hwang, I. C. Lin, and K. F. Hwang, “Cryptanalysis of the batch verifying multiple RSA digital signatures,” Informatica, vol. 11, no. 1, 2000, pp. 15-19.

[15] S. J. Hwang, M. S. Hwang, and S. F. Tzeng, “A New Digital Multisignature Scheme with Distinguished Signing Authorities,” Journal of Information Science and Engineering, vol. 19, no. 5, 2003, pp. 881-887.

[16] C. T. Li, M. S. Hwang, and S. M. Chen, “A Batch Verifying and Detecting the Illegal Signatures,” International Journal of Innovative Computing, Information and Control, vol. 6, no. 12, 2010, pp. 5311-5320.

[17] L. H. Li, S. F. Tzeng, and M. S. Hwang, “Improvement of Signature Scheme Based on Factoring and Discrete Logarithms,” Applied Mathematics and Computation, vol. 161, no. 1, 2005, pp. 49-54.

[18] C. H. Lim and P. J. Lee, “Security of interactive DSA batch verification,” Electronics Letters, vol. 30, no. 19, 1994, pp. 1592-1593.

[19] Gu Lize Wang Feng Zhou Yousheng ,Zheng Shi-hui, “A Bilateral Secure Threshold Signature Scheme with Distinguished Signing Authorities”, International Journal of Advancements in Computing Technology, vol. 4, no. 8, 2012, pp. 100-107.

[20] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone, Hand-book of Applied Cryptography. CRC Press, 1996.

[21] D. Naccache, D. Mraihi, D. Rapheali, and S. Vaudenay, “Can DSA be improved: Complexity trade-offs with the digital signature standard,” in Proceedings of Eurocrypt'94, 1994, pp. 85-94, Lecture Notes in Computer Science.

[22] K. Ohta and T. Oktamoto, “A modification of the Fiat-Shamir scheme,” in Advances in Cryptology, CRYPTO'88, 1988, pp. 232-243, Lecture Notes in Computer Science.

[23] V.Shao, “Batch verifying multiple DSA-type digital signatures,” Computer Networks, vol. 37, no. 3-4, 2001, pp. 383-389.

[24] S. F. Tzeng and M. S. Hwang, “A Batch Verification for Multiple Proxy Signature,” Computer Standards & Interfaces, vol. 26, no. 2, 2004, pp. 61-71.

[25] S. F. Tzeng, C. C. Lee, and M. S. Hwang, “Digital Signature with Message Recovery and Its Variants Based on Elliptic Curve Discrete Logarithm Problem,” Parallel Processing Letters, vol. 21, no. 1, 2011, pp. 77.

[26] S. F. Tzeng, C. Y. Yang, and M. S. Hwang, “A Nonrepu-diable Threshold Multi-Proxy Multi-Signature Scheme with Shared Verification,” Future Generation Computer Systems, vol. 20, no. 5, 2004, pp. 887-893.

[27] S. F. Tzeng, C. Y. Yang, and M. S. Hwang, “A New Digital Signature Scheme Based on Factoring and Discrete Logarithms,” International Journal of Computer Mathematics, vol. 81, no. 1, 2004, pp. 9-14.

[28] Luo Wenjun, Li Changying, Bai Guojing, Chenlong, “A Certificateless Sequential Multi-Signature Scheme without Pairings”, International Journal of Advancements in Computing Technology, vol. 4, no. 9, 2012, pp. 193-199.

[29] Y. Zheng, T. Matsumoto, and H. Imai, “Structural properties of one-way hash functions,” in Advances in Cryptology, CRYPTO'90, 1990, pp. 285-302, Lecture Notes in Computer Science.: University Science, 1989