Improvement on Ahn et al.'s RSA P-Homomorphic Signature Scheme

Zhiwei Wang[1,2,3,4]

[1] College of Computer, Nanjing University of Posts and Telecommunications, Nanjing, Jiangsu 210003, China
[2] State Key Laboratory of Information Security (Institute of Information Engineering, Chinese Academy of Sciences), Beijing, 100190, China
[3] Jiangsu High Technology Research Key Laboratory for Wireless Sensor Networks, Nanjing, Jiangsu 210003, China
[4] Key Lab of Broadband Wireless Communication and Sensor Network Technology (Nanjing University of Posts and Telecommunications), Ministry of Education Jiangsu Province, Nanjing, Jiangsu 210003, China
zhwwang@njupt.edu.cn

Abstract. P-homomorphic signature is a general framework for computing on authenticated data, which is recently proposed by Ahn et al. With P-homomorphic signature, any third party can derive a signature on the object message m' from a signature of m, if m' and m satisfy $P(m, m') = 1$ for some predicate P which denotes the authenticatable relationship between m' and m. Ahn et al. proposed a RSA P-homomorphic signature scheme by using a RSA accumulator, which is very efficient in space. However, the computational cost of verification and derivation is very heavy. We present an improved P-homomorphic signature scheme based on factoring problem. In our construction, the time efficiency of both verification and derivation are much better than Ahn's scheme.

Keywords: P-homomorphic signature, signature derive, factoring problem, cloud computing.

1 Introduction

With the development of cloud computing, many secure problems have been proposed. One of the most important problem is that it's too much of a security risk to give a public cloud provider such as Amazon or Google access to unencrypted data. While data can be sent to and from a cloud provider's data center in encrypted form, the servers that power a cloud can't do any work on it that way. In 2009, Gentry proposed a fully homomorphic encryption scheme to make it possible to analyze data without decrypting it [1]. Up to now, some homomorphic encryption schemes have been proposed[1–3], while only a few homomorphic signature schemes have been presented.

In the past few years, there are about three research classes which have touch on this area: **quoting/redacting signature**, **arithmetic signature**, **transitive signature**. Quoting/redacting signature [4–8] is that given Alice's signature

A.D. Keromytis and R. Di Pietro (Eds.): SecureComm 2012, LNICST 106, pp. 19–28, 2013.

on some message m, any one can derive Alice's signature on a subset of m. Quoting/redacting signature is specially applied to signed text message and signed images. Arithmetic signature [9–13] is motivated by the application of secure network coding, which is that given Alice's signature on vectors $v_1, \cdots, v_k \in \mathbb{F}_p^n$, any one can derive Alice's signature on a vector in linear span of $v_1, \cdots, v_k$. In transitive signature[14–18], given Alice's signature on edges in a graph G, any one can derive Alice's signature on a pair of vertices u, v, if there exists a path from u to v in G.

Recently, Ahn et al. put forth a general framework of computing on signed data[19], which can cover all the three classes research above. Their definition is instantiated with any predicates, and allows to repeat derivation on the signatures. They call this general framework slightly homomorphic signature or P-homomorphic signature. In [19], they provide two general constructions for computing signatures on any univariate, closed predicates, namely predicates $P(M, m')$ where M only contains a single message and if $P(a, b) = P(b, c) = 1$ then $P(a, c) = 1$. The first construction is a brute force construction from any signature. Soundness of this construction follows from the underlying signature scheme. However, the signatures in this construction may become very large, which effects both the signing time and signature size. The second construction is a RSA accumulator-based construction, which can produce a short signature, but the computational cost of both verification and derivation is even worse than the first construction. The prime search component of hash function is the dominant factor. Ahn et al. [19] also proposed the third efficient construction, which is only suitable for quoting substrings and not a generic solution. Furthermore, the signature derivation procedure in this construction is very complex.

In this paper, we propose an improved generic construction of P-homomorphic signature from Ahn's RSA accumulator based construction. Our scheme is efficient in both in space and computational costs. The rest of this paper is organized as follows: In the next section, we review some preliminaries related to our construction. Then, we review Ahn et al.'s construction in Section 3. In Section 4, we propose our improved scheme. The security properties will be analyzed in Section 5. We conclude in Section 6.

2 Preliminaries

2.1 Some Concepts in Number Theory

Let $N = p \times q$ be a composite modulus, where p and q are two large prime numbers. Let $\mathbb{Q}_N$ denote the subgroup of squares in $\mathbb{Z}_N^*$. Then, it is well known that $\mathbb{Q}_N$ is a cyclic group with order $\phi(N)/4 = (p-1)(q-1)/4$ [20].

Factoring Problem. given a k-bit composite N, which is a multiple of two large primes p and q, to output p or q. Factoring problem is usually considered as a hard problem.

Theorem 2.1. *Let $a \in \mathbb{Q}_N$, $N = p \times q$, where p, q are large primes and $p = 2p' + 1$, $q = 2q' + 1$. p' and q' are also large primes. Then $a^{2d} \equiv a \pmod{N}$, where $d = (N - p - q + 5)/8$.*

Proof. Since $d = \frac{(N-p-q+5)}{8} = \frac{(p-1)(q-1)+4}{8} = \frac{4p'q'+4}{8}$, then $a^{2d} = a^{p'q'+1} = a \pmod N$.(We note that $\phi(N)/4 = (p-1)(q-1)/4 = p'q'$.)

Indeed, Theorem 2.1 provides a way to compute one square root of a quadratic residue $a \in \mathbb{Q}_N$.

To further understand the algorithm of computing a 2^lth root of a quadratic residue, let us introduce the following theorem.

Theorem 2.2. *Let $N = p \times q$, where p, q are large primes and $p = 2p' + 1$, $q = 2q' + 1$. p' and q' are also large primes. If $a = x^2 \in \mathbb{Q}_N$, then $a^d \in \mathbb{Q}_N$.*

Proof. Since p' and q' are also large primes, then $p' = 2k + 1$ and $q' = 2k' + 1$ for some integer k and k'. Then,$d = \frac{(N-p-q+5)}{8} = \frac{4p'q'+4}{8} = 2kk' + k + k' + 1$ is an integer. So we have $a^d = x^{2d} = (x^d)^2 \pmod N$. Thus, $a^d \in \mathbb{Q}_N$.

From Theorem 2.1 and Theorem 2.2, we can know that a square root of $a \in \mathbb{Q}_N$ computed by $a^d \pmod N$, still stays in $\mathbb{Q}_N$. Therefore, a 2^lth root of a can be computed as $a^{d^l} \pmod N$,where d^l is computed over $\mathbb{Z}_{p'q'}$.

Let N be a multiple of two large primes p,q and $a \in \mathbb{Q}_N$. If s_1 and s_2 are two square roots satisfying $s_1 \neq \pm s_2 \pmod N$, then N could be factored by computing $GCD(s_1 + s_2, N)$ or $GCD(s_1 - s_2, N)$ as the non-trivial divisor of N. However, if $s_1 = \pm s_2 \pmod N$, it will be no useful to the factorization of N. Thus, if given two random square roots, the probability of factoring N is 1/2.

2.2 Definition of P-Homomorphic Signature

Definition of Predicate P. Let $\mathcal{M}$ be a message space. A predicate P is defined as $P : 2^{\mathcal{M}} \times \mathcal{M} \to \{0, 1\}$ which maps a set of messages and a message to a bit [19]. For the quoting application, the predicate P is defined as $P(M, m') = 1$ where $M \subset \mathcal{M}$ iff m' is a quote from the set of message M. The predicate P for arithmetic computation is defined as $P((v_1, \cdots, v_k), v) = 1$ whenever v is in the span of $v_1, \cdots, v_k$.

A P-homomorphic signature scheme Π for message space $\mathcal{M}$ and predicate P consists of four algorithms: **KeyGen**, **Sign**, **SignDerive**, **Verify**. Here, **Sign** is simply a special case of **SignDerive**. We describe them as follows:

KeyGen(1^λ): This algorithm outputs a key pair (pk, sk). We can treat the secret key sk as a signature on the empty message ε.

Sign($sk, m \in \mathcal{M}$): Given the secret key sk and a message m, the algorithm outputs a signature σ.

SignDerive($pk, (\{\sigma_m\}_{m\in\mathcal{M}}, M), m', \omega$): This algorithm takes as input the public key, a set of messages M and corresponding signatures $\{\sigma_m\}_{m\in M}$, a derived message m', and possibly some auxiliary information ω. It generates a new signature σ' on m'. For complex predicate, ω can be served as a witness for $P(M, m') = 1$. For simplicity, $Sign(sk, m) = SignDerive(pk, (sk, \varepsilon), m, \cdot)$ denotes that if given sk, any messages can be derived. Here sk can be considered as a signature on the empty message ε.

Verify(pk, m, σ): If this algorithm is provided with the public key, message, and the corresponding signature σ, it returns 1 when the signature is valid, otherwise, it returns 0.

We must confirm that if $P(M, m') = 1$ then

$$SignDerive(pk, (Sign(sk, M), M)m') \neq \perp,$$

and for all signature tuples $\{\sigma_m\}_{m \in M}$ satisfying

$$\sigma' \leftarrow SignDerive(pk, (Sign(sk, M), M)m') \neq \perp,$$

$Verify(pk, m', \sigma') = 1$ holds. These two rules make the signature derivation be iterative if allowed by P.

3 Review of Ahn et al.'s RSA Accumulator-Based Scheme

In Ahn et al.'s construction[19], they only focus on *univariate, closed* predicates $P(M, m')$, namely M contains a single component and if $P(a, b) = P(b, c) = 1$ then $P(a, c) = 1$. We now describe their RSA accumulator-based scheme as follows:

KeyGen(1^λ): This algorithm selects three parameters: a 20λ-bit RSA modulus N, $a \in \mathbb{Z}_N$ and a hash function H_p which maps arbitrary strings to 2λ-bit prime numbers. The public key $pk = (N, H_p, a)$, and the secret key sk is the factorization of N.

Sign($sk, m \in \mathcal{M}$): Let $U = P(\{m\}) = \{m' | m' \in \mathcal{M} and P(m, m') = 1\}$. Compute the signature as

$$\sigma = a^{1/(\prod_{u_i \in U} H_p(u_i))} \pmod N.$$

SignDerive(pk, σ, m, m'): In this algorithm, first check that $P(m, m') = 1$, if not then outputs $\perp$. Otherwise, let $U' = P(\{m'\})$, compute the signature as

$$\sigma' = \sigma^{\prod_{u_i \in U - U'} H_p(u_i)} \pmod N.$$

The signature is essentially of the form $a^{1/(\prod_{u_i \in U'} H_p(u_i))} \pmod N$.

Verify(pk, m, σ): Let $U = P(\{m\})$, if $a = \sigma^{\prod_{u_i \in U} H_p(u_i)} \pmod N$ the outputs 1, otherwise, returns 0.

This scheme can be proved secure under RSA, and the most important advantage is that signatures only require one element in $\mathbb{Z}_N^*$. However, the computational cost is very heavy. If computing an l-symbol quote from an n-symbol message requires $\mathcal{O}(n(n-l))$ evaluation of $H_p()$ and $\mathcal{O}(n(n-l))$ modular exponentiations. Verification requires $\mathcal{O}(l^2)$ evaluation of $H_p()$ and $\mathcal{O}(l^2)$ modular exponentiations. **The computational cost of prime search in $H_p()$ is the dominating factor, since the outputs of $H_p()$ must be a prime number.**

4 Our Improved Scheme

For overcoming the above shortcoming, we propose an improved scheme which can be described as follows (We also focus on *univariate, closed* predicate.):

KeyGen(1^λ): This algorithm selects a composite number N which is a multiple of two safe large prime numbers $p = 2p' + 1, q = 2q' + 1$. p and q satisfy that $(p-1)(q-1) \geq 2^l$ and $pq < 2^{l+1}$ (l is another secure parameter derived from λ). Then, computes $d = (N - p - q + 5)/8$, and chooses $h \in \mathbb{Q}_N$ and a hash function $H() : \{0,1\}^* \to \{0,1\}^l$. The public key $pk = (N, H, h)$, while the secret key $sk = d$.

Sign($sk, m \in \mathcal{M}$): Let $U = P(\{m\}) = \{m' | m' \in \mathcal{M} \quad and \quad P(m, m') = 1\}$. Compute the signature as

$$\sigma = h^{\prod_{u_i \in U} d^{H(u_i)}} \pmod N.$$

SignDerive(pk, σ, m, m'): In this algorithm, first check that $P(m, m') = 1$, if not then outputs $\perp$. Otherwise, let $U' = P(\{m'\})$, compute the signature as

$$\sigma' = \sigma^{\prod_{u_i \in U - U'} 2^{H(u_i)}} \pmod N.$$

The signature is essentially of the form $h^{\prod_{u_i \in U'} d^{H(u_i)}} \pmod N$.

Verify(pk, m, σ): Let $U = P(\{m\})$, if $h = \sigma^{\prod_{u_i \in U} 2^{H(u_i)}} \pmod N$ the outputs 1, otherwise, returns 0.

In the above scheme, signatures still requires only one element in $\mathbb{Z}_N^*$. However, the computational burden is much better than Ahn's construction. Firstly, $H()$ is a common hash function, which does not require the output must be a prime number. Thus, there exists no prime search component in $H()$, which saves a large computational cost compared with Ahn's construction. Secondly, the modular exponentiations in SignDerive and Verify algorithm can be computed very fast, since $\sigma^{\prod_{u_i \in U - U'} 2^{H(u_i)}}$ and $\sigma^{\prod_{u_i \in U} 2^{H(u_i)}}$ can be done only through *adding* and *shifting*.

5 Security Analysis

In this section, we first describe the security properties of P-homomorphic signature. Then, we prove that our improved scheme achieves the security properties.

5.1 Security Definition

The security definition of P-homomorphic signature should capture two properties: context hiding and unforgeability[19].

Context hiding means that a signature should reveal nothing more than the message being signed. If a signature on m' was derived from a signature on m,

an attacker should not learn anything about m other than what can be deduced by m'. This should be true even the original signature on m is revealed. For example, in the case of quoting application, a signed quote should not reveal the length of original message, the position of the quote etc. Ahn et al. proposed a powerful statistic definition of context hiding called *Strong Context Hiding*.

Strong Context Hiding. *Let $M \subset \mathcal{M}$ and $m' \in \mathcal{M}$ such that $P(M, m') = 1$. Let (pk, sk) be the key pair. A P-homomorphic signature Π is strong context hiding if and only if the following distribution are statically close:*

$$(sk, \{\sigma_m\}_{m \in M} \leftarrow Sign(sk, M), Sign(sk, m'))_{sk,M,m'}$$

$$(sk, \{\sigma_m\}_{m \in M} \leftarrow Sign(sk, M), SignDerive(pk, (\{\sigma_m\}_{m \in M}, M), m'))_{sk,M,m'}$$

The distributions are taken over the coins of Sign and SignDerive. Here, for a set of message $M = \{m_1, m_2, \cdots, m_k\}$, it is convenient to let $Sign(sk, M)$ denote independently signing each of the k messages, which can be depicted as follows:

$$Sign(sk, M) = (Sign(sk, m_1), \cdots, Sign(sk, m_k)).$$

The above definition implies that a derived signature on m' is indistinguishable from a signature generated independently of M. Therefore, the derived signature cannot reveal any information about M other than what is revealed by m'. This definition uses statical indistinguishability meaning that even a unbounded attacker cannot distinguish the derived signatures from the fresh ones. Thus, it is called *strong context hiding*. Furthermore, Ahn et al. also proposed another definition called *context hiding* by using computational indistinguishability, which is very complex, since the attacker needs to be given a signing oracle. The relation of *context hiding* and *strong context hiding* can be proved that if a P-homomorphic signature scheme is context hiding then it is strong context hiding.

Unforgeability of P-homomorphic signature is that an attacker can adaptively choose messages and acquire the corresponding derived signatures, however, he/she cannot output a signature on a message that is not derivable from the set of signed messages at his hand. Ahn et al. presented the definition of unforgeability by extending the basic notion of adaptively chosen existential unforgebaility. Ahn's definition can be defined by a game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ with respect to scheme Π over message space $\mathcal{M}$.

Setup: The challenger $\mathcal{C}$ runs $\mathbf{KeyGen}(1^\lambda)$ to obtain a key pair (pk, sk) and sends pk to $\mathcal{A}$, while keeps sk for itself. Furthermore, $\mathcal{C}$ keeps a set T that is initially empty.

Queries: $\mathcal{A}$ adaptively issues the following queries to $\mathcal{C}$

1. $Sign(m \in \mathcal{M})$: The challenger $\mathcal{C}$ runs $\mathbf{Sign}(sk, m)$ to get σ, and places (m, σ) into a table T. Then $\mathcal{C}$ returns σ to $\mathcal{A}$
2. $SignDerive(m' \in \mathcal{M})$: This challenger $\mathcal{C}$ retrieves all the tuples (σ_i, m_i) in T for $i = 1, \cdots, k$. If T is empty, then $\mathcal{C}$ returns $\perp$. Otherwise, let $M =$

$\{m_1, \cdots, m_k\}$. If $P(M, m') = 1$, then $\mathcal{C}$ runs **SignDerive**$(pk, (\{\sigma_m\}_{m \in M}, M), m')$ to obtain σ'. $\mathcal{C}$ keeps (σ', m') into T, and returns σ' to $\mathcal{A}$.

Output: Finally,$\mathcal{A}$ outputs a pair (σ', m'). If $\mathcal{A}$ wins the game, the following two conditions should be satisfied.

1. Verify$(pk, m', \sigma') = 1$;
2. Let M be the set of messages in T. $P(M, m') = 0$ must hold.

Let $\mathbf{ADV}_{\mathcal{A}}$ denote the probability of $\mathcal{A}$ winning.

Unforgeability. *If $\boldsymbol{ADV}_{\mathcal{A}}$ is negligible in λ, then A P-homomorphic signature scheme Π is adaptively chosen-message attacks **unforgeable**.*

Ahn et al. also proposed a weaker notion of unforgeability[19], which is also defined by a game between challenger $\mathcal{C}$ and adversary $\mathcal{A}$. Ahn et al. call it **NHU** game, in which the adversary only makes calls to *Sign* oracle. The only difference between **NHU** game and the standard unforgeability game for a P-homomorphic signature scheme is that in this game, the adversary only wins if his forged signature on m^* such that for all $m \in T$, $P(m, m^*) = 0$, while in the standard unforgeability game, the adversary wins if his forged signature on any message that is not in T.

Ahn et al. proved that *if a P-homomorphic signature scheme is **NHU** unforgeable and strong context hiding, then it is standard-unforgeable.*[19] This implies that strong context hiding property can help simplify the security argument of standard unforgeability.

5.2 Security Proof

In this section, we will provide the security proof to our improved scheme.

Theorem 5.1. *If the factoring problem is hard, then our improved P-homomorphic signature scheme is unforgeable and context hiding in the random oracle.*

We proved Theorem 5.1 by showing that our scheme is strong context hiding and **NHU**-unforgeable.

Lemma 5.1. *The improved P-homomorphic signature scheme is strong context hiding.*

Proof. Let $pk = (N, H, h)$, and challenge be any m,m' where $P(m, m') = 1$. Let $U = P(m)$ and $U' = P(m')$. We can deduce that

$$Sign(sk, m) = \sigma = h^{d^{\prod_{u \in U} H(u)}} \pmod N$$

$$Sign(sk, m') = \sigma' = h^{d^{\prod_{u \in U'} H(u)}} \pmod N$$

$$\begin{aligned} SignDerive(pk, (\sigma, m), m') &= \sigma^{2^{\prod_{u \in U - U'} H(u)}} \pmod N \\ &= (h^{d^{\prod_{u \in U} H(u)}})^{2^{\prod_{u \in U - U'} H(u)}} \pmod N \\ &= h^{d^{\prod_{u \in U'} H(u)}} \pmod N \\ &= \sigma'. \end{aligned}$$

Since $Sign(sk, m')$ equals $SignDerive(pk, (\sigma, m), m')$, the probability that an adversary can distinguish between them is exactly 1/2. Thus, our improved P-homomorphic signature scheme is strong context hiding.

Lemma 5.1 *The improved P-homomorphic signature scheme is **NHU**-uforgeable if factoring problem is hard.*

Proof. We will prove this lemma through the **NHU** game discussed above. In the **NHU** game, the adversary $\mathcal{A}$ is only allowed to make *Sign* oracle queries. We suppose adversary $\mathcal{A}$ queries the random oracle on at most s unique inputs. If adversary $\mathcal{A}$ can outputs a successful forgery in **NHU** game, then we can construct a challenger $\mathcal{C}$ that solves the factoring problem with a non-negligible probability. Given a challenge N, $\mathcal{C}$'s goal is to output the factorization of N.

Setup: Challenger $\mathcal{C}$ chooses $s-1$ $l$bits distinct integer numbers $e_1, \cdots, e_{s-1}$ at random, but all $e_i \neq 2, e_i > 0$. Let E denote $\{e_1, \cdots, e_{s-1}\}$. Then, $\mathcal{C}$ guesses a random number $i^* \in \{1, \cdots, s\}$, and keeps it. Next, $\mathcal{C}$ randomly selects $y \in \mathbb{Z}_N^*$, and computes $h = y^{\prod_{e_i \in E} 2^{e_i}} \pmod N$. Obviously, $h \in \mathbb{Q}_N$. Finally, $\mathcal{C}$ sends N, h to $\mathcal{A}$, and will ask its queries on random oracle H interactively.

Queries: $\mathcal{C}$ answers $\mathcal{A}$'s adaptively *Hash* and *Sign* queries.

- Hash queries: When $\mathcal{A}$ makes the jth query to the random oracle, if $j = i^*$, then $\mathcal{C}$ answers 2. Otherwise, if $j < i^*$, $\mathcal{C}$ answers with e_j, and e_{j-1} otherwise. Since we assume $\mathcal{A}$'s queries are different every time, let x^* as the input when $H(x^*) = 2$.
- Sign queries: When $\mathcal{A}$ makes a sign queries on message m, $\mathcal{C}$ computes $U = P(m)$, and if $x^* \in U$, then $\mathcal{C}$ aborts. Otherwise, $\mathcal{C}$ calls H on all elements of U not previously queried to H. Let $E(U)$ denote the set of integer numbers derived by calling H on every element in U. $\mathcal{C}$ computes

$$\sigma = y^{\prod_{i \in [E - E(U)]} 2^{e_i}} \pmod N,$$

  and returns σ, m as the answer to $\mathcal{A}$.

Outputs: Eventually, $\mathcal{A}$ outputs a valid forged signature σ on message m, where m cannot be derived from any element returned by *Sign*. If m is still not queried to H, or $m \neq x^*$, then $\mathcal{C}$ aborts. Otherwise, let $U = P(\{x^*\}) - \{x^*\}$, and $E(U)$ denotes the set of integer numbers derived by calling H on every element in U. From the verification equation, the following equation holds.

$$h^{\prod_{e_i \in E(U)} d^{e_i}} = y^{\prod_{e_i \in [E - E(U)]} 2^{e_i}} = \sigma^2 \pmod N.$$

We computes $b = \sum_{i \in [E-E(U)]} e_i$, then $y^{2^b} = \sigma^2 \pmod N$. If $\sigma \neq \pm y^{2^{b-1}} \pmod N$, $\mathcal{C}$ can factoring N by computing $GCD(\sigma + y^{2^{b-1}}, N)$ or $GCD(\sigma - y^{2^{b-1}}, N)$. Since y is randomly chosen in $\mathbb{Z}_N^*$, the probability that σ and $\pm y^{2^{b-1}}$ are distinct is 1/2.

Probability Analysis: We assume that the attacker $\mathcal{A}$ can win the above game with the probability of ϵ. $\mathcal{A}$'s final forgery is based on the i^*th hash queries

$(1 < i^* < s)$, and i^* is randomly chosen from $\{1, \cdots, s\}$. So we can deduce that challenger $\mathcal{C}$ can solve the factoring problem through $\mathcal{A}$'s forgery with the probability of $\frac{\epsilon}{2s}$.

This completes our proof.

Note: Our improved scheme is proved secure under the hardness of the factoring problem, while Ahn et al.'s construction is proved secure under the RSA assumption. However, the hardness of RSA problem is not identical to the hardness of the factoring problem . It is generally believed that RSA assumption is stronger than factoring assumption[21].

6 Conclude

P-homomorphic signature is a general framework for computing on authenticated data, which can make any third party derive a signature on the object message m' from a signature of m, if m' and m satisfy $P(m, m') = 1$ for some predicate P. Similar with homomorphic encryption, P-homomorphic signature can also make cloud computing providers provide good services to customers. Cloud providers can directly compute on the existing signature files without secret keys. In this paper, we propose an improved P-homomorphic signature scheme, which is more efficient in computational cost than Ahn's scheme. Furthermore, our scheme can be proved under the hardness of the factoring problem.

Acknowledgments. This research is supported by the Project Funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions(PAPD), the Natural science fund for colleges and universities in Jiangsu Province No. 11KJB520015, and the Program for Excellent Talents in Nanjing University of Posts and Telecommunications No. NY209014.

References

1. Gentry, C.: A fully homomorphic encryption scheme. PhD thesis, Stanford University (2009)
2. Smart, N.P., Vercauteren, F.: Fully Homomorphic Encryption with Relatively Small Key and Ciphertext Sizes. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 420–443. Springer, Heidelberg (2010)
3. van Dijk, M., Gentry, C., Halevi, S., Vaikuntanathan, V.: Fully Homomorphic Encryption over the Integers. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 24–43. Springer, Heidelberg (2010)
4. Ateniese, G., Chou, D.H., de Medeiros, B., Tsudik, G.: Sanitizable Signatures. In: De Capitani di Vimercati, S., Syverson, P.F., Gollmann, D. (eds.) ESORICS 2005. LNCS, vol. 3679, pp. 159–177. Springer, Heidelberg (2005)
5. Miyazaki, K., Hanaoka, G., Imai, H.: Digitally signed document sanitizing scheme based on bilinear maps. In: ASIACCS 2006: Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security, pp. 343–354 (2006)
6. Haber, S., Hatano, Y., Honda, Y., Horne, W., Miyazaki, K., Sander, T., Tezoku, S., Yao, D.: Efficient signature schemes supporting redaction, pseudonymization, and data deidentification. In: ASIACCS 2008, pp. 353–362 (2008)

7. Brzuska, C., Fischlin, M., Freudenreich, T., Lehmann, A., Page, M., Schelbert, J., Schröder, D., Volk, F.: Security of Sanitizable Signatures Revisited. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 317–336. Springer, Heidelberg (2009)
8. Brzuska, C., Fischlin, M., Lehmann, A., Schröder, D.: Unlinkability of Sanitizable Signatures. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 444–461. Springer, Heidelberg (2010)
9. Boneh, D., Freeman, D., Katz, J., Waters, B.: Signing a Linear Subspace: Signature Schemes for Network Coding. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 68–87. Springer, Heidelberg (2009)
10. Gennaro, R., Katz, J., Krawczyk, H., Rabin, T.: Secure Network Coding over the Integers. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 142–160. Springer, Heidelberg (2010)
11. Boneh, D., Freeman, D.M.: Homomorphic Signatures for Polynomial Functions. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 149–168. Springer, Heidelberg (2011)
12. Boneh, D., Freeman, D.M.: Linearly Homomorphic Signatures over Binary Fields and New Tools for Lattice-Based Signatures. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 1–16. Springer, Heidelberg (2011)
13. Wei, L., Coull, S.E., Reiter, M.K.: Bounded vector signatures and their applications. In: ASIACCS 2011, pp. 277–285 (2011)
14. Hevia, A., Micciancio, D.: The Provable Security of Graph-Based One-Time Signatures and Extensions to Algebraic Signature Schemes. In: Zheng, Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, pp. 379–396. Springer, Heidelberg (2002)
15. Bellare, M., Neven, G.: Transitive signatures: New schemes and proofs. IEEE Transactions on Information Theory 51, 2133–2151 (2005)
16. Shahandashti, S.F., Salmasizadeh, M., Mohajeri, J.: A Provably Secure Short Transitive Signature Scheme from Bilinear Group Pairs. In: Blundo, C., Cimato, S. (eds.) SCN 2004. LNCS, vol. 3352, pp. 60–76. Springer, Heidelberg (2005)
17. Yi, X.: Directed Transitive Signature Scheme. In: Abe, M. (ed.) CT-RSA 2007. LNCS, vol. 4377, pp. 129–144. Springer, Heidelberg (2006)
18. Neven, G.: A simple transitive signature scheme for directed trees. Theoretical Computer Science 396(1-3), 277–282 (2008)
19. Ahn, J.H., Boneh, D., Camenisch, J., Hohenberger, S., Shelat, A., Waters, B.: Computing on Authenticated Data. Cryptology ePrint Archive: Report 2011/096, http://eprint.iacr.org/2011/096
20. Shoup, V.: A Computational Introduction to Number Theory and Algebra, p. 534. Cambridge University Press (2005)
21. Cao, Z., Zhu, H., Lu, R.: Provably secure robust threshold partial blind signature. Science in China Series F: Information Sciences 49(5), 604–615 (2006)