A Full Homomorphic Message Authenticator with Improved Efficiency

Wenbin Chen and Hao Lei

***Abstract*—In the system of a fully homomorphic message authenticator, anyone can make any computations based on authenticated data. At the same time, a short tag is produced to authenticate the result of the computation. Without the underlying data, the user can use his private key to verify this tag for ensuring that the claimed result is correct. Recent, Gennaro and Wichs use a fully homomorphic encryption to construct a fully homomorphic message authenticator. In this paper, we propose a fully homomorphic message authenticator with improve efficiency than Gennaro and Wichs's construction. Our fully homomorphic message authenticator is of less verification complexity than Gennaro and Wichs's construction.**

***Index Terms*—Fully homomorphic message authenticators, fully homomorphic encryption, cloud security.**

I. Introduction

In the cloud computing background, sometimes users hope that a remote service provider can do reliable computations over the outsourced data. The first fully homomorphic encryption [1], [2] scheme developed by C. Gentry make it possible to perform arbitrary computation over outsourced data at the same time the privacy of the outsourced data is kept. After C. Gentry's construction, many full homomorphic encryption schemes have been developed [3]-[10].

Being enlighten by the idea of fully homomorphic encryption, Gennaro and Wichs instantiateand study the question of authenticating arbitrary computations over outsourced data: "if the remote server claims that the execution of some program P over the user's outsourced data results in an output *y*, how can the user be sure that this is indeed the case?" [11]

In order to solve this problem, Gennaro and Wichs propose a fully homomorphic message authenticator in [11]. A fully homomorphic message authenticator can be viewed as a symmetric-key version of fully homomorphic signatures that are introduced by Boneh and Freeman [12]. However, how to construct a fully homomorphic signature is still an important open problem.

Almost all previous research work about homomorphic message authentication and signatures are restricted to linear homomorphisms. The first linear homomorphic signature scheme is proposed by Johnson *et al*. in [13]. Since then, many linear homomorphic signatures and message authentication in the context of network coding are developed [14]-[20]. Linear homomophic signature and message authentication schemes can be used to construct proofs of storage and retrievability [21]-[24]. In [12], Boneh and Freeman propose a homomorphic signature for polynomial functions. Their security is based on hard problems on ideal lattice. Recently, Gennaro and Wichs design the first fully homomorphic message authenticator based on the fully homomorphic encryption systems [11].

The interactive and non-interactive proofs can also be used to make someone believe that some computation is correct [25]-[26]. But their constructions rely on the random-oracle model and don't support arbitrary composition.

There are research works about the problem of delegating computation [27]-[30], which is similar to a fully homomorphic message authenticator. However, the delegating computation is different from a fully homomorphic message authenticator. In those delegating computation schemes, when the function *f*, the input *x* and the output y are known for a user, but he does not want to perform the computation work of *f*(*x*), the sever is required to make the user believe that *f*(*x*) = *y* is correct. Compared to the fully homomorphic authenticators, several disadvantages are pointed out by Gennaro and Wichs about delegating computations: interaction, single use, bounded size, no composition.

In this paper, we propose a fully homomorphic message authenticator with improved efficiency than Gennaro and Wichs's construction: our scheme is of less verification complexity.

Compared with Gennaro and Wichs's algorithm, the novelty of our algorithm is as follows. In our algorithm, the tag of one bit contains some encryption of 1 except those encryption of 0 and that bit. In our verification algorithm, we only need two computaiton, while Gennaro and Wichs's algorithm needs *n*/2 computation.

The remainder of the paper is organized as follows. Section II reviews some basic definitions. In Section III, we design an algorithm for our algorithm for fully homomorphic authenticator. Finally, Section IV gives some open problems.

II. Definitions

In the following, we repeat some basic definitions of full homomorphic authenticators from [11].

Definition 2.1. A labeled-program $P = (f, \tau_1, \ldots, \tau_k)$

Manuscript received December 9, 2013; revised April 2, 2014. This work was supported in part by the science research funding of Huawei Technologies Co., Ltd. under Grant No. YBCB2012055, by the National Natural Science Foundation of China (NSFC) under Grant No.11271097 and the research project of Guangzhou Education Bureau under Grant No.2012A074.

Wenbin Chen is with the Department of Computer Science, Guangzhou University, Guangzhou, China (e-mail: cwb2011@gzhu.edu.cn).

Hao Lei is with Shield Lab., Huawei Technologies Co., Ltd., China (e-mail: leiyok@163.com).

consists of a circuit $f : \{0, 1\}^k \rightarrow \{0, 1\}$ along with a distinct input label $\tau_i \in \{0, 1\}$ for every input wire $\tau_i \in [k] = \{1, . . . , k\}$ [11].

Definition 2.2. Given a circuit $g : \{0, 1\}^t \rightarrow \{0, 1\}$ and some labeled programs $P_1, \dots, P_t$, the composed program that is denoted by $P^* = g(P_1, . . . , P_t)$ is defined as the computing g on the outputs of $P_1, \dots, P_t$. Given the canonical identity circuit g_{id} and some label $\tau \in \{0, 1\}^*$, the identity program with label τ is defined as $I\tau = (g_{id}, \tau)$ [11].

Definition 2.3. (Fully Homomorphic Authenticator [11]). A fully homomorphic authenticator scheme consists of the probabilistic polynomial time algorithms (KeyGen, Auth, Ver, Eval) with the following syntax:

1) KeyGen(1^n) → (evk, sk): Outputs the secret key sk and an evaluation key evk.
2) Auth$_{sk}(b, \tau) \rightarrow \sigma$: Create a tag σ that authenticates the bit $b \in \{0, 1\}$ under the label $\tau \in \{0, 1\}^*$.
3) Eval$_{evk}(f, \sigma) \rightarrow \psi$: The deterministic evaluation procedure takes a vector of tags $\sigma = (\sigma_1, \dots, \sigma_k)$ and a circuit $f : \{0; 1\}^k \rightarrow \{0, 1\}$. It outputs a tag ψ. If each σ_i authenticates a bit b_i as the output of some labeled-program P_i (possibly the identity program), then ψ should authenticate $b^* = f(b_1, . . . , b_k)$ as the output of the composed program $P^* = f(P_1, . . . , P_k)$.
4) Ver$_{sk}(e, P, \psi) \rightarrow$ {accept, reject}: The deterministic verification procedure uses the tag ψ to check that $e \in \{0, 1\}$ is the output of the program P on previously authenticated labeled data.

The scheme need satisfy the following several properties: authentication correctness, evaluation correctness, succinctness. Their detailed description of these properties can be found in [11].

In the following, we introduce the definition of authenticator security.

Definition 2.4. (Authenticator Security [11]). Consider the following game ForgeGame$^A(1^n)$ between an attacker $A(1^n)$ and a challenger:

1) The challenger chooses $(evk, sk) \leftarrow$ KeyGen(1^n) and gives evk to A. It initializes $T = \Phi$;
2) The attacker A can adaptively submit arbitrarily many authentication queries of the form (b, τ) to the challenger. On each such query, if there is some $(\tau, \dots) \in T$ (i.e. the labelτ is not fresh), then the challenger ignores it. Else it updates $T = T \bigcup \{(\tau, b)\}$, associating the label τ with the authenticated bit b, and replies with σ Auth$_{sk}(b, \tau)$.
3) Finally, the attacker outputs some forgery $(e^*, P^* = (f, \tau_1^*, . . . , \tau_t^*), \psi^*)$.

The output of the game is 1 iff V ersk(e^*, P^*, ψ^*) = accept and one of the following two conditions holds:

Type I Forgery: There is some $i \in [k]$ such that the label $(\tau_i^*, ..)$ does not appear in T. (i.e., No bit was ever authenticated under the label τ_i^* involved in the forgery.)

Type II Forgery: The set T contains tuples $(\tau_1, b_1), . . . , (\tau_k, b_k)$, for some bits $b_1, . . . , b_k \in \{0, 1\}$ such that $f^*(b_1, . . . , b_k) \neq e^*$. (i.e., The labeled program P^* does not output e^* when executed on previously authenticated labeled data $b_1, . . . , b_k$).

We say that a homomorphic authenticator scheme is secure (without verification queries) if, for any probabilistic polynomial-time A, we have Pr[ForgeGame$^A(1^n)$ = 1] ≤negl(n).

Definition 2.5. (Fully Homomorphic Encryption [11]). A fully homomorphic encryption (FHE) scheme is a quadruple of PPT algorithms HE=(HE .KeyGen, HE.Enc, HE.Dec, HE.Eval)defined as follows.

1) HE.KeyGen(1^n) → (pk, evk, sk): Outputs a public encryption key pk, a public evaluation key evk and a secret decryption key sk.
2) HE.Enc$_{pk}(b) \rightarrow c$: The ciphertext of a bit $b \in \{0, 1\}$ is c under the public key pk.
3) HE.Dec $_{sk}(c) \rightarrow b$: Decrypts ciphertext c using sk to a plaintext bit b.
4) HE.Eval$_{evk}(g, c_1, \dots, c_t) \rightarrow c^*$:

The deterministic evaluation algorithm takes the evaluation key evk, a boolean circuit g: $\{0; 1\}^t \rightarrow \{0, 1\}$, and a set of t ciphertexts $c_1, . . . , c_t$. It outputs the result ciphertext c^*.

An FHE should also be of the following properties: encryption correctness, evaluation correctness, succinctness, semantic Security.

Canonical FHE meaning that the HE.Eval procedure just evaluates the circuit recursively, level-by-level and gate-by-gate [11].

Definition 2.6. Hash Tree of a Circuit [11]. If $g : \{0, 1\}^k \rightarrow \{0, 1\}$ is a circuit and $H : \{0; 1\}^* \rightarrow \{0, 1\}^m$ is some hash function, the hash tree $g^H : (\{0; 1\}^*)^k \rightarrow \{0, 1\}^m$ is a function which takes as input strings $v \in \{0, 1\}^*$ for each input wire of g. For every wire w in the circuit g, the value of $g^H(v_1, . . . , v_k)$ at w is defined inductively as:

1) val(w) = $H(v_i)$ if w is the i-th input wire of g.
2) val(w) = H(val(w_1), . . . , val(w_k)) if w is the output wire of some gate with input wires $w_1, . . . , w_t$.

The output of the function $g^H(v_1, . . . , v_k)$ is defined to be its value at the output wire of g.

III. Algorithms for Fully Homomorphic Authentication

In this section, we give the construction of our algorithm for fully homomorphic authenticator. Our construction is based on Gennaro and Wichs's fully homomorphic authentication algorithm in [11]. But our algorithm improve the efficiency of Gennaro and Wichs's construction since ourconstruction is of less verification complexity.

Let $\{f_K: \{0, 1\}^* \rightarrow \{0, 1\}^{r(n)}\}$ be a (variable-input-length) pseudo-random function PRF family, $K \in \{0,1\}^n$. Let FH be a family of (variable-length) collision-resistant hash functions (CRHF) H: $\{0, 1\}^* \rightarrow \{0, 1\}^{m(n)}$. Let HE = (HE.KeyGen, HE.Enc, HE.Dec, HE.Eval) be a canonical fully homomorphic encryption scheme, where the encryption algorithm uses $r = r(n) = \omega(\log(n))$ random bits. Our fully homomorphic authentication algorithm (KeyGen, Auth, Eval, V er) is as follows:

KeyGen(1^n): Choose a PRF key $K \leftarrow \{0, 1\}^n$, a CRHF $H \leftarrow$FH and an encryption key (pk, evk', sk') HE.KeyGen(1^n). Select a subset $S \subseteq [n]$ and $|S| = 2n/3$, where each index $i \in [n]$ is added to the set S independently with probability 1/2 . Randomly select a subset $T \in \{0,1\}^{2n/3}$ such that T is of n/3 0's and $n/3$ 1's. Randomly select a position t_0 from $n/3$ 0's positions and a position t_1 from $n/3$ 1's positions. Output $evk = (evk', H)$, $sk = (pk, evk', H, sk', K, S, T, t_0, t_1)$.

Authsk(b, τ): Given $b \in \{0, 1\}$ and $\tau \in \{0, 1\}^*$ do the following:

1) Set n random strings $rand_i = f_K((\tau, i))(1 \leq i \leq n)$ and $v= f_K(\tau)$.
2) Produce n ciphertexts $c_1, \ldots, c_n$ as follows. For $i \in [n]\backslash S$, compute c_i = HE.Encpk(b; $rand_i$) as encryptions of the bit b. For $i \in S$, compute c_i = HE.Encpk (0; $rand_i$) as encryption of 0 if i is the j-th position of T and the j-th position of T is 0. Otherwise, compute c_i = HE.Encpk(1; $rand_i$) as encryption of 1.
3) Output $\sigma = (c_1, \ldots, c_n, v)$.

Eval$_{evk}$(g, σ) : Given $\sigma = (\sigma_1, \ldots, \sigma_t)$, where each $\sigma_j = (c_{1,j}, \ldots, c_{n,j}, v_j)$ $(1 \leq j \leq t)$, do the following:

1) For each $i \in [n]$, compute c_i^* = HE.Eval$_{evk'}$(g, $c_{i,1}, \ldots, c_{i,t}$).
2) Compute $v^* = g^H(v_1, \ldots, v_t)$ as the value of the hash tree g^H of g at $v_1, \ldots, v_t$.
3) Compute e_0 = HE.Eval$_{evk'}$(g, 0, . . . , 0) and e_1 = HE.Eval$_{evk'}$(g, 1, . . . , 1).

Output = $(c_1^*, \ldots, c_n^*, v^*, e_0, e_1)$.

Versk(e,P, ψ): Parse $P = (g, \tau_1, \ldots, \tau_t)$ and $\psi = (c_1^*, \ldots, c_n^*, v^*, e_0, e_1)$.

1) Compute $v_i = f_K(\tau_i)$ for all $1 \leq i \leq t$ and $v' = g^H(\tau_1, \ldots, \tau_t)$. If $v' \neq v^*$, output reject.
2) For $i \in S$, when i is the j-th position of T and the j-th position of T is 0 and $j \neq t_0$, decrypt e_i =HE.Dec$_{sk'}$(c_i^*) and if $e_i \neq e_0$ output reject; when $j = t_0$, for each $1 \leq h \leq t$, compute $rand_{i,h} = f_K((\tau_h, i))$ and $c_{i,h}$ =HE.Encpk(0, $rand_{i,h}$). Further, we compute c_i' = HE.Eval$_{evk'}$ (g, $c_{i,1}, \ldots c_{i,t}$), and if $c_i' \neq c_i^*$ output reject; when i is the j-th position of T and the j-thposition of T is 1, decrypt e_i =HE.Dec$_{sk'}$(c_i^*) and if $e_i \neq e_1$ output reject; when $j = t_1$, for each $1 \leq h \leq t$, compute $rand_{i,h} = f_K((\tau_h, i))$ and $c_{i,h}$ =HE.Encpk(1, $rand_{i,h}$). Further, we compute c_i' = HE.Eval$_{evk'}$ (g, $c_{i,1}, \ldots c_{i,t}$), and if $c_i' \neq c_i^*$ output reject;
3) For each $i \in [n]\backslash S$, decrypt e_i =HE.Dec$_{sk'}$(c_i^*) and if $e \neq e_i$ output reject.

If the above doesn't reject, output accept.

For our algorithm, we show that it is secure without verification queries as follows.

Theorem 3.1. Our homomorphic authenticator scheme is secure without verification queries when $\{f_K\}$ is a PRF family, H is a CRHF family and HE is a semantically secure canonical FHE.

Proof: Based on the encryption correctness of HE, it is easy to know that the authentication correctness of our algorithm holds. By the evaluation correctness of HE and HE is canonical, it is also easy to know that the evaluation correctness of our algorithm holds.

The security of our algorithm without verification queries is showed as follows. Let A be some PPT attacker and let $\mu(n)$ = Pr[ForgeGame$^A(1^n)$ = 1]. In order to prove the security of our algorithm without verification queries, it is enough to show that $\mu(n)$ is negligible. We use a series of hybrid games to prove that $\mu(n)$ is negligible.

Game$_1$: In ForgeGame, replace the PRF with a truly random function. That is, each call to f_K is replaced by a call to a completely random function F:$\{0,1\}^* \rightarrow \{0,1\}^{r(n)}$. Based on the pseudo-randomness $\{f_K\}$, it is easy to get: Pr[Game$^A{}_1(n)$ = 1] $\geq \mu(n)$- negl(n), where negl(n) denotes negligible.

Game$_2$: Game$_2$ is defined by modifying the winning condition in Game$_1$: the game outputs 1 on type II forgery and outputs 0 on a type I forgery. Let E denote the event that attacker wins with a type I forgery in Game$_1$. In the following, we show that Pr[E] = negl(n).

Since H is a collision-resistant hash function, the probability of finding collisions on H is negligible, i.e. Pr[$H(x) = H(y)$] = negl(n) for any $x \neq y \in \{0, 1\}^*$. By the definition of the hash tree g^H, when $x \neq y$, if $g^H(v_1, \ldots, x, \ldots, v_k) = g^H(v_1, \ldots, y, \ldots, v_k)$, then there are some collisions on H at some level of g^H. So Pr[$g^H(v_1, \ldots, x, \ldots, v_k) = g^H(v_1, \ldots, y, \ldots, v_k)$] is less than the probability of finding collisions on H. Thus, Pr[$g^H(v_1, \ldots, x, \ldots, v_k) = g^H(v_1, \ldots, y, \ldots, v_k)$] is negligible.

When event E occurs, by the definition of type i forgery, the attacker submits a forgery e^*, $P^* = (g, \tau_1^*, \ldots, \tau_t^*)$, $\psi^* = (c_1^*, \ldots, c_t^*, v^*)$ such that one of some i ($i \in [t]$) does not appear in authentication queries and $V\,er_{sk}(e^*, P^*, \psi^*) = 1$. Thus, in the step 1 of verification process, the value$v_i = F(\tau_i^*) \in \{0, 1\}^{r(n)}$ is produced randomly. We re-sample a value v_i' from $\{0, 1\}^{r(n)}$ randomly and independently again. Let B denote the event that verification accepts both times. Then Pr[E]$^2 \leq$ Pr[B]. Let D denote the event that $v_i \neq v_i'$ and $\bar{D}$ denote the event that $v_i = v_i'$. Then Pr[$\bar{D}$] = $2^{-r(n)}$. Since Pr[B] = Pr[$B \cap D$] + Pr[$B \cap \bar{D}$], Pr[B] $\leq$Pr[$B \cap D$] + Pr[$\bar{D}$] = Pr[$B \backslash D$] + $2^{-r(n)}$. So Pr[E]$^2 \leq$ Pr[$B \cap D$] + $2^{-r(n)}$. The event $B \cap D$ occurs means that verification accepts both times and $v_i \neq v_i'$. From the step 1 of verification process, we have Pr[$g^H(v_1, \ldots, v_i, \ldots, v_t) = g^H(v_1, \ldots, v_i', \ldots, v_t)$] = v^*. Thus Pr[$B \cap D$] $\leq$Pr[$g^H(v_1, \ldots, v_i, \ldots, v_t) = g^H(v_1, \ldots, v_i', \ldots, v_t)$]. So Pr[$B \cap D$] is negligible. Hence Pr[E]^2 is negligible. Thus, Pr[E] = negl(n).

So, we get: Pr[Game$^A{}_2(n)$ = 1]$\geq$ Pr[Game$^A{}_1(n)$ = 1] - Pr[E] $\geq \mu(n)$ - negl(n).

Game$_3$: In Game$_3$, the winning condition is modified. The challenger remembers the corresponding tag σ when he answers authentication queries. For any type II verification query e^*, $P^* = (g, \tau_1^*, \ldots, \tau_t^*)$, $\psi^* = (c_1^*, \ldots, c_n^*, v^*)$, the challenger can now recall the correct bits b_j and tags $\sigma_j = (c_{1,j}, \ldots, c_{n,j})$ associated with the input labels τ_j^* for $j \in [t]$.

For $i \in [n]$, let $\tilde{c}_i$ = HE.Eval$_{evk}$(g, $c_{i,1}, \ldots, c_{i,t}$) be the "honest ciphertexts", which an honestly generated tag would contain for the program P^*. In Game$_3$, we replace step (3) of the verification procedure as follows:

3'. For each $i \in [n]\backslash S$: if $\tilde{c}_i = c_i^*$ then output reject.

In Game$_3$, step (3') is different from the original step (3) in Game$_2$. Let $e = g(b_1, \ldots, b_t)$ be the honest output of P^*. In an accepting type II forgery, we must have $e^* \neq e$ but the decryption of the "honest ciphertexts" is e, i.e. HE.Dec$_{sk'}$($\tilde{c}_i$) = e. So, for any accepting type II forgery in Game$_2$, $c_i^* \neq \tilde{c}_i$ holds for all $i \in [n]\backslash S$.

Therefore, any accepting type II forgery in Game$_2$ is also accepting in Game_3. Hence, we get: $\Pr[\text{Game}^A{}_3(n) = 1] \geq \Pr[\text{Game}^A{}_2(n) = 1](n) \geq \mu(n)$ -negl(n).

Game_4: We define Game_4 by modifying answering authentication queries in Game_3. When the challenger answers authentication queries in step (2) of the authentication procedure, all of c_i (even for $i \in S$) are computed as encryptions of the correct bit b. Then, by the semantic security of the encryption scheme HE, we get: $\Pr[\text{Game}^A{}_4\ (n) = 1] > \Pr[\text{Game}^A{}_3(n) = 1]$ - negl$(n) \geq \mu(n)$ - negl(n).

Hence, $\mu(n) \leq \Pr[\text{Game}^A{}_4(n) = 1]$+negl$(n)$. In the following, we show that $\Pr[\text{Game}^A{}_4\ (n) = 1]$ is negligible.

In Game_4, the set $S \subseteq [n]$ is picked by the challenger during verification and is ignored when answering authentication queries. For any type II forgery e^*, P^*, $\psi^* = (c_1{}^*, \ldots, c_n{}^*, v^*)$, let $c_1', \ldots, c_n'$ be the "honest ciphertexts", which can honestly generated tag ψ would contain for the program P^* (see description of Game_3). We use S' to denote the set of indices on which the forged and honest ciphertexts match, i.e. $S' = \{i \in [n], c_i{}^* = c_i'\}$. Only if steps (3') of verification pass, the attacker wins. Thus, if $i \in [n]\backslash S$, then $c_i' \neq c_i{}^*$. So $[n]\backslash S \subseteq [n]\backslash S'$. Hence, $S' \subseteq S$. Over the random choice of S, the probability is $1/2^{n-|S'|}$

Since $|S| = 2n/3$, $|S'| \leq 2n/3$. So $n - |S/| \geq n/3$. Hence, $1/2^{n-|S'|} \leq 2^{-n/3}$.

Therefore, we get: $\mu(n) \leq \Pr[\text{Game}^A{}_4(n) = 1]$ + negl$(n) \leq 2^{-n/3}$ + negl(n) = negl(n).

Theorem 3.2. Our algorithm is of less verification complexity than Gennaro and Wichs's fully homomorphic authentication algorithm.

Proof: In the verification process of Gennaro and Wichs's algorithm, it needs $n/2$ computation of the program P. However, our algorithm need two these computation. So, our algorithm is of less verification complexity than Gennaro and Wichs's algorithm.

IV. Conclusion

In this paper, we propose a fully homomorphic message authenticator. Our algorithm is of less verification complexity than Gennaro and Wichs's construction. Our algorithm save much computation in the verification process.

There are still many open questions left. For example, how to reduce the tag size from n ciphertexts to smaller ciphertexts? Maybe the most ambitious questions is how to construct fully homomorphic signatures.

Acknowledgment

We would like to thank the anonymous referees for their careful readings of the manuscripts and many useful suggestions.

References

[1] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proc. 41st ACM STOC*, 2009, pp. 169-178.

[2] C. Gentry, "A fully homomorphic encryption scheme," Ph.D. dissertation, Stanford University, 2009.

[3] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "(Leveled) Fully homomorphic encryption without bootstrapping," in *Proc. Innovations in Theoretical Computer Science*, 2012, pp. 309-325.

[4] Z. Brakerski and V. Vaikuntanathan, "Fully homomorphic encryption from ring-LWE and security for key dependent messages," in *Proc. 2011 CRYPTO*, pp. 505-524.

[5] Z. Brakerski and V. Vaikuntanathan, "Efficient fully homomorphic encryption from (Standard) LWE," in *Proc. the 52nd Annual Symposium on Foundations of Computer Science*, 2011, pp. 97-106.

[6] J. S. Coron, A. Mandal, D. Naccache, and M. Tibouchi, "Fully homomorphic encryption over the integers with shorter public keys," in *Proc. 2011 CRYPTO*, 2011, pp. 487-504.

[7] M. Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, "Fully homomorphic encryption over the integers," in *Proc. 2010 EUROCRYPT*, 2010, pp. 24-43.

[8] C. Gentry, "Toward basing fully homomorphic encryption on worst-case hardness," in *Proc. 2010 CRYPTO*, 2010, pp. 116-137.

[9] C. Gentry and S. Halevi, "Fully homomorphic encryption without squashing using depth-3 arithmetic circuits," in *Proc. 2011 FOCS*, 2011, pp. 107-109.

[10] C. Gentry, S. Halevi, and N. P. Smart, "Fully homomorphic encryption with polylog overhead," in *Proc. EUROCRYPT*, 2012, pp. 465-482.

[11] R. Gennaro and D. Wichs, "Fully homomorphic message authenticators," in *Proc. Advances in Cryptology - ASIACRYPT 2013*, pp. 301-320.

[12] D. Boneh and D. M. Freeman, "Homomorphic signatures for polynomial functions," in *Proc. EUROCRYPT*, 2011, pp. 149-168.

[13] R. Johnson, D. Molnar, D. X. Song, and D. Wagner, "Homomorphic signature schemes," in *Proc. 2002* CT-*RSA*, 2002, pp. 244-262.

[14] S. Agrawal and D. Boneh, "Homomorphic MACs: MAC-based integrity for network coding," in *Proc. ACNS*, 2009, pp. 292-305.

[15] N. Attrapadung and B. Libert, "Homomorphic network coding signatures in the standard model," in *Proc. PKC*, 2011, pp. 17-34.

[16] D. Boneh and D. M. Freeman, "Linearly homomorphic signatures over binary fields and new tools for lattice-based signatures," in *Proc. PKC*, 2011, pp. 1-16.

[17] D. Boneh, D. Freeman, J. Katz, and B. Waters, "Signing a linear subspace: Signature schemes for network coding," in *Proc. PKC*, 2009, pp. 68-87.

[18] D. Catalano, D. Fiore, and B. Warinschi, "Efficient network coding signatures in the standard model," in *Proc. 15th International Conference on Practice and Theory in Public Key Cryptography*, Darmstadt, Germany, 2012, pp. 680-696.

[19] D. M. Freeman, "Improved security for linearly homomorphic signatures: A generic framework," in *Proc. 15th International Conference on Practice and Theory in Public Key Cryptography*, Darmstadt, Germany, 2012, pp. 697-714.

[20] R. Gennaro, J. Katz, H. Krawczyk, and T. Rabin, "Secure network coding over the integers," in *Proc. PKC*, 2010, pp. 142-160.

[21] G. Ateniese, R. C. Burns, R. Curtmola, J. Herring, L. Kissner, Z. N. J. Peterson, and D. Song, "Provable data possession at untrusted stores," in *Proc. ACM CCS*, 2007, pp. 598-609.

[22] G. Ateniese, S. Kamara, and J. Katz, "Proofs of storage from homomorphic identification protocols," in *Proc. ASIACRYPT*, 2009, pp. 319-333.

[23] Y. Dodis, S. P. Vadhan, and D. Wichs, "Proofs of retrievability via hardness amplification," in *Proc. TCC*, 2009, pp. 1090-127.

[24] H. Shacham and B. Waters, "Compact proofs of retrievability," in *Proc. ASIACRYPT*, 2008, pp. 90-107.

[25] S. Goldwasser, S. Micali, and C. Rackoff, "The knowledge complexity of interactive proof systems," *SIAM Journal on Computing*, vol. 18, no. 1, pp. 186-208, 1989.

[26] S. Micali, "CS proofs (extended abstracts)," in *Proc. FOCS*, 1994, pp. 436-453.

[27] B. Applebaum, Y. Ishai, and E. Kushilevitz, "From secrecy to soundness: Efficient verification via secure computation," in *Proc. ICALP*, 2010, pp. 152-163.

[28] K. M. Chung, Y. Kalai, and S. P. Vadhan, "Improved delegation of computation using fully homo- morphic encryption," in *Proc. CRYPTO*, 2010, pp. 483-501.

[29] S. Goldwasser, Y. T. Kalai, and G. N. Rothblum, "Delegating computation: interactive proofs for muggles," in *Proc. 40th ACM STOC*, 2008, pp. 113-122.

[30] R. Gennaro, C. Gentry, and B. Parno, "Non-interactive verifiable computing: Outsourcing computation to untrusted workers," in *Proc. CRYPTO*, 2010, pp. 465-482.

Wenbin Chen received his M.S. degree in mathematics from Institute of Software, Chinese Academy of Science in 2003, and the Ph.D. degree in computer science from North Carolina State University, U.S.A in 2010. He is currently an associate professor at the College of Computer Science and Educational Software, Guangzhou University. His research interests include algorithm design and analysis, bioinformatics algorithms, graph algorithms, graph mining, computational complexity, database, etc.

Hao Lei received his Ph.D. degree in cryptography from SKLOIS, Chinese Academy of Science in 2006. He is currently a researcher at Shield Lab., Huawei Technologies Co., Ltd. China. His research focuses on the area of public key cryptography, with specific interests in tackling the dilemma between privacy and security concern in untrusted and distributed scenarios such as privacy protection, cryptography access control, cloud computing security and usability.