EVOC: More Efficient Verifiable Outsourced Computation from Any One-way Trapdoor Function

Jun Zhou, Zhenfu Cao, Xiaolei Dong, Xiaodong Lin
Shanghai Key Lab for Trustworthy Computing
East China Normal University
Shanghai 200062, China
Email: zhoujun_tdt@sjtu.edu.cn, zfcao@sei.ecnu.edu.cn, dongxiaolei@sei.ecnu.edu.cn
Faculty of Business and Information Technology
University of Ontario Institute of Technology
Oshawa,Canada
Email: xiaodong.lin@uoit.ca

***Abstract*—Verifiable outsourced computation enables a computational resource-constrained mobile device to outsource the computation of a function F on multiple inputs $x_1, \cdots, x_n$ to the cloud that is generally assumed to possess abundant powers. The most existing work depends on Yao's Garbled Circuit and fully homomorphic encryptions that took considerable computational overhead on weak clients. In this paper, a more efficient verifiable outsourced computation of encrypted data EVOC supporting any functions from any one-way trapdoor function is proposed, based on our newly-devised privacy preserving data aggregation supporting both addition and multiplication operations without exploiting fully homomorphic encryption (FHE). It solves the open problem suggested by Gennaro et al. that how to devise a verifiable computation scheme that used a more efficient primitive than FHE. Finally, the formal security proof and extensive efficiency evaluations demonstrate our proposed EVOC satisfies the target security and privacy requirements and far outperforms the state-of-the-art in terms of computational and communication complexity.**

I. Introduction

Cloud computing has been increasingly ubiquitous owing to the bloom of wireless mobile communications where the resources are generally asymmetrically allocated [1,2]. The mobile devices are resource-constrained and can hardly afford the large volumes of data processing especially in the big data era. Instead, they always delegate the energy-consuming computation tasks to the cloud possessing abundant powers in a pay-per-use manner. Sometimes, the outsourced data and applications are so critical that it is required to rule out all accidental errors on the final result. For example, in the e-healthcare systems, a set of body sensors is deployed on, in or around the patient to monitor the realtime health condition such as the body temperature, the pulse and the blood pressure and multidimensional data would be aggregated and processed by calculating certain statistics for data mining. The final result would be extremely vital for the physicians to make corresponding medical treatment. Another example is the smart grid where a series of smart meters are widely deployed in each unit to monitor the realtime electricity usage for dynamically deciding the peak and valley electricity consumption periods and achieving differential pricing. Unfortunately on the other hand, it is likely for the cloud providing computation services to hold financial incentive to return incorrect answers that are derived for significantly less expense without the detection of users (i.e. the owners of the mobile devices).

Therefore, in all the above-mentioned scenarios, a series of elemental requirements has to be achieved for verifiable outsourced computation [3,6,8]. First of all, the amount of work performed by the mobile devices to initialize and verify the correctness of the computation task is required to be substantially less than the work on computing the underlying function from scratch. Additionally, both the input and output privacy should be well protected, that is, the inputs and the outcome of the delegated computation cannot be exposed to even the collusion of the cloud and malicious clients. As is in the scenarios presented above, if the patient's individual health data and the realtime power consumption data were exposed, it would be likely for the user to experience labor contract termination or robbery when their homes are estimated empty. Finally, the work on outsourced computation performed by cloud is required to be close to computing the original function itself. It guarantees the cloud is able to fulfill the computation task in a tolerable amount of time.

Recently, the fundamental research in fully homomorphic encryption (FHE) [4] provides a convincing answer to allow the cloud to perform the delegated computation on arbitrary functions in the encrypted domain. Unfortunately, they did not suffice to outsourced computation since the FHE provides no guarantee on correct computation verification. The concept of verifiable computation (VC) was firstly introduced by R. Gennaro et. al. and a non-interactive verifiable computation outsourced to untrusted workers [3] was proposed, which mainly depended on Yao's Garbled Circuit and FHE, but only permits private verifiable computation. Most recently, M. Barbosa et al. proposed a delegatable homomorphic encryption with applications to secure outsourcing computation [6]. Nevertheless, all the verifiable outsourced computation schemes presented above adopted time-consuming FHE as a cryptographic primitive which resource-constrained mobile

978-1-4673-6432-4/15/$31.00 ©2015 IEEE

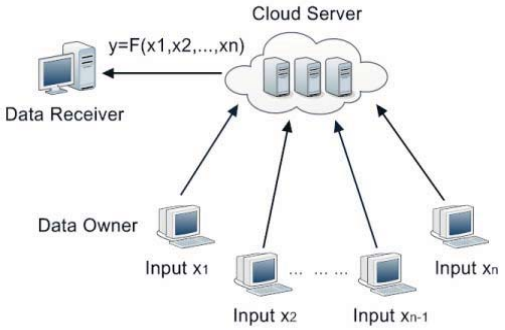

Fig. 1: Network Architecture of Verifiable Outsourced Computation

devices cannot afford. Therefore, how to devise a authorized verifiable computation using a more efficient primitive than FHE still remains an open problem. Papamanthou et al. proposed signatures of correct computation [7], but it guaranteed neither the input/output privacy nor the evaluation privacy, followed by a nearly practical verifiable computation Pinocchio proposed by Parno et al.. However, the involved pairing operations would still bring about a heavy load on the power-restrained devices. In this paper, a more efficient verifiable outsourced computation of encrypted data EVOC from any one-way trapdoor function is proposed. The main contributions are outlined as follows.

(1) Firstly, an efficient privacy-preserving data aggregation supporting both addition and multiplication operations is proposed without adopting FHE as a primitive. In contrast to the state-of-the-art, it is required to compute the one-way trapdoor function only once to aggregate n dimensional data.

(2) Secondly, a more efficient verifiable outsourced computation is proposed by exploiting our newly-devised privacy-preserving data aggregation and Yao's Garbled Circuit. Besides the efficiency enhancement, it also permit authorized verifiable computation.

(3) Formal security proof and extensive performance evaluations demonstrate our proposed EVOC satisfies the security and privacy requirement of verifiable outsourced computation and far outperforms the existing work in both computational and communication overhead.

The remainder of this paper is organized as follows. Network architecture and security models are presented in the next section. A more efficient verifiable outsourced computation EVOC exploiting our newly devised privacy-preserving data aggregation is proposed in Sec. III. The formal security proof is given in Sec. IV, followed by the performance evaluations in Sec. V. Finally, we conclude our paper in Sec. VI.

II. Network Architecture and Security Models

A. Network Model

A verifiable outsourced computation is generally executed in the power asymmetric scenario comprising the following entities: the data owner, the cloud and the data receiver. Both the data owner and receiver (i.e. their held wireless mobile devices) are assumed to be extremely resource-constrained and the cloud is power abundant. As is illustrated in Fig. 1, at the network initialization phase, each user registers to the cloud server and rents its computation resources in a pay-per-use manner. Then, a series of data inputs $\vec{x} = \{x_1, \cdots, x_n\}$ from one specific data owner are sent to the cloud in the encrypted form. The cloud server performs the delegated computation of $y = F(\vec{x})$ and returns the result to the data receiver. Authorized receivers can successfully decipher the original function output and verifies its correctness. Specifically, a verifiable outsourced computation comprises the following algorithms.

KeyGen$(F, \lambda) \rightarrow (PK, SK)$: On input the security parameter λ, it randomly generates a public key PK encoding the underlying function F which is used by the cloud to compute F. A corresponding secret key SK is also generated and privately kept by the data owner.

ProbGen$(SK, \vec{x}) \rightarrow (\sigma_{\vec{x}}, \tau_{\vec{x}})$: This algorithm encodes the inputs $\vec{x}$ of the function F using the secret key SK to generate a public value $\sigma_{\vec{x}}$ exploited for outsourced computation and a secret value $\tau_{\vec{x}}$ privately kept by the client.

Compute$(PK, \sigma_{\vec{x}}) \rightarrow \sigma_y$: This algorithm is executed by the cloud which outputs the encoded function result $\sigma_y = F(\sigma_{\vec{x}})$ using public key PK and the encoded function F's input $\sigma_{\vec{x}}$.

Verify$(SK, \tau_{\vec{x}}, \sigma_y) \rightarrow y \; or \; \perp$: This algorithm is performed by the data receiver that outputs the original result of $y = F(\vec{x})$ by exploiting the secret key SK, the secret decoding $\tau_{\vec{x}}$ and the encoded output σ_y if it indeed represents the valid output of F on $\vec{x}$; otherwise it outputs $\perp$.

B. Security Model

The security of a verifiable outsourced computation means given function F and its input $\vec{x_i} = (x_{i,1}, x_{i,2}, \cdots, x_{i,n})$, the cloud can hardly convince the verification algorithm to output $\hat{y}$ such that $\hat{y} \neq F(\vec{x})$. The formal security game $Game_{VC}^{vef}(F, \lambda)$ is defined as follows.

$$\begin{aligned}
&(PK, SK) \leftarrow_R KeyGen(F, \lambda);\\
&For\ i = 1, \cdots, l;\\
&\vec{x}_i \leftarrow \mathcal{A}(PK, \vec{x}_1, \sigma_1, \cdots, \vec{x}_i, \sigma_i);\\
&(\sigma_i, \tau_i) \leftarrow ProbGen(SK, \vec{x}_i);\\
&(i, \hat{\sigma}_y) \leftarrow \mathcal{A}(PK, \vec{x}_1, \sigma_1, \cdots, \vec{x}_l, \sigma_l);\\
&\hat{y} \leftarrow Verify(SK, \tau_i, \hat{\sigma}_y)\\
&If\ \hat{y} \neq \perp\ and\ \hat{y} \neq F(\vec{x}_i), output\ 1; otherwise\ 0.
\end{aligned} \tag{1}$$

Definition 1. A verifiable outsourced computation VC is secure on function F, if and only if for any adversary probabilistically polynomial bounded $\mathcal{A}$,

$$ADV_{\mathcal{A},VC}^{vef}(F, \lambda) = Prob[Game_{VC}^{vef}(F, \lambda) = 1] \leq neg(\lambda) \tag{2}$$

where $neg(\lambda)$ is a negligible function on λ.

We continue to define the input and output privacy of our proposed more efficient verifiable outsourced computation. The input privacy is formally defined by $Game_{VC}^{Pri}(F, \lambda)$

presented as follows and the output privacy can be defined the same way.

$$\begin{aligned}&(PK, SK) \leftarrow_R KeyGen(F, \lambda);\\ &(\vec{x}_0, \vec{x}_1) \leftarrow \mathcal{A}^{O^{ProbGen}(SK,\cdot)}(PK);\\ &(\sigma_0, \tau_0) \leftarrow ProbGen(SK, \vec{x}_0);\\ &(\sigma_1, \tau_1) \leftarrow ProbGen(SK, \vec{x}_1);\\ &b \leftarrow_R \{0,1\}; \hat{b} \leftarrow \mathcal{A}^{O^{ProbGen}(SK,\cdot)}(PK, \vec{x}_0, \vec{x}_1, \sigma_b);\\ &If\ \hat{b} = b, output\ 1; otherwise\ 0.\end{aligned} \tag{3}$$

Definition 2. A verifiable outsourced computation VC is private on function F, if and only if for any adversary probabilistically polynomial bounded $\mathcal{A}$,

$$ADV^{pri}_{\mathcal{A},VC}(F, \lambda) = Prob[Game^{Pri}_{VC}(F, \lambda) = 1] \leq neg(\lambda) \tag{4}$$

where $neg(\lambda)$ is a negligible function on λ.

III. Our Construction

In this section, a more efficient verifiable outsourced computation of encrypted data EVOC from any one-way trapdoor function is proposed. It is observed that in Gennaro et al.'s proposed construction [3], the fully homomorphic encryption is only utilized when computing the circuit describing the range differentiating machine M in the encrypted domain and deciphering the final wire output, without individual wire input decryption afterwards. In this paper, the efficiency of our EVOC has been significantly optimized, by exploiting a newly-devised privacy-preserving data aggregation supporting both addition and multiplication operations from any one-way trapdoor function. The key idea of our EVOC is a tradeoff between the efficiency optimization and weakening the individual decryption functionality we would not use in EVOC, but still holding the valuable property of addition and multiplication aggregation.

A. Privacy-preserving Data Aggregation

In this subsection, an efficient privacy-preserving data aggregation from any one-way trapdoor function, supporting both addition and multiplication operations, from any one-way trapdoor function is proposed. It mainly comprises the following algorithms **AGG.KGen**, **AGG.Enc**, **AGG.Eval** and **AGG.Dec**.

AGG.KGen: On input 1^λ where λ is the security parameter, it runs a trapdoor function generator denoted as a probabilistic polynomial time (PPT) algorithm $\mathcal{G}$ and outputs a tuple of functions (f, f^{-1}) on $\{0,1\}^{2\lambda}$ with a pair of corresponding keys (pk_f, sk_f). It also outputs two hash functions $H_0, H_1 : \{0,1\}^* \rightarrow \{0,1\}^{2\lambda}$. The public parameters are $PPR = (pk_f, H_0, H_1)$ and the secret key is sk_f assigned to the receiver. p, q are kept secret by the sender.

AGG.Enc: The data sender randomly selects two big primes p, q of $|p| = |q| = \lambda$ and computes the publicized $N = pq$. Let $i = 1, \cdots, n$ and m_i be a series of original messages generated by the sender during each time slot t_i. Then, the sender randomly selects $U_i^{add}, U_i^{mul} \in_R \mathbb{Z}_N, r^{'} \in_R \{0,1\}^\lambda$, and computes

$$m_{i,p} \equiv m_i \ mod\ p, m_{i,q} \equiv m_i \ mod\ q. \tag{5}$$

Since we have

$$1 \equiv q^{-1}q \ mod\ p, 1 \equiv p^{-1}p \ mod\ q, \tag{6}$$

the sender calculates the ciphertexts as follows,

$$\begin{aligned}&C_{1,1} = f(p \parallel r^{'})\\ &C_{2,i} = q^{-1}qm^p_{i,p} + p^{-1}pm^q_{i,q} + U_i^{add} \ mod\ N,\\ &C_{3,i} = (q^{-1}qm^p_{i,p} + p^{-1}pm^q_{i,q})U_i^{mul} \ mod\ N,\end{aligned} \tag{7}$$

where $\parallel$ means the random padding operation of p to the length of 2λ. When one time period T including n multiple time slots ends, the sender publicizes

$$U_T^{add} = \sum_{i=1}^{n} U_i^{add} \ mod\ N, U_T^{mul} = \prod_{i=1}^{n} U_i^{mul} \ mod\ N, \tag{8}$$

computes $C^{add}_{ram} = H_0(p \parallel U_T^{add})$, $C^{mul}_{ram} = H_0(p \parallel U_T^{mul})$, and sends $C_{u,i} = (C_{1,1}, C_{2,i}, C_{3,i}, C^{add}_{ram}, C^{mul}_{ram})$ to the evaluator.

AGG.Eval: The evaluator performs the addition and multiplication aggregation operations

$$\begin{aligned}&C_T^{add} = \sum_{i=1}^{n} C_{2,i} - U_T^{add} \ mod\ N\\ &= q^{-1}q\sum_{i=1}^{n} m^p_{i,p} + p^{-1}p\sum_{i=1}^{n} m^q_{i,q} \ mod\ N\\ &= q^{-1}q(\sum_{i=1}^{n} m_{i,p})^p + p^{-1}p(\sum_{i=1}^{n} m_{i,q})^q \ mod\ N,\\ &C_T^{mul} = \prod_{i=1}^{n} C_{3,i}(U_T^{mul})^{-1} \ mod\ N\\ &= \prod_{i=1}^{n}(q^{-1}qm^p_{i,p} + p^{-1}pm^q_{i,q}) \ mod\ N\\ &= q^{-1}q(\prod_{i=1}^{n} m_{i,p})^p + p^{-1}p(\prod_{i=1}^{n} m_{i,q})^q \ mod\ N,\\ &C_3^{add} = H_1(C_T^{add} \parallel C^{add}_{ram}), C_3^{mul} = H_1(C_T^{mul} \parallel C^{mul}_{ram})\end{aligned} \tag{9}$$

and sends $C_A = (C_1, C_T^{add}, C_T^{mul}, C_3^{add}, C_3^{mul})$ to the receiver.

AGG.Dec: The receiver firstly computes $p \parallel r^{'} = f^{-1}(C_{1,1})$ by using her/his secret key sk_f, removes the last λ-bits of $p \parallel r^{'}$ to derive p, and checks whether all of $C^{add}_{ram} = H_0(p \parallel U_T^{add}), C^{mul}_{ram} = H_0(p \parallel U_T^{mul})$ and $C_3^{add} = H_1(C_T^{add} \parallel C^{add}_{ram}), C_3^{mul} = H_1(C_T^{mul} \parallel C^{mul}_{ram})$

hold. If not, this algorithm outputs $\perp$; otherwise, the receiver continues to compute $q = Np^{-1}$ and

$$\begin{aligned} C_T^{add} \bmod p &= q^{-1}q(\sum_{i=1}^{n} m_{i,p})^p \bmod p = \sum_{i=1}^{n} m_{i,p} \bmod p, \\ C_T^{add} \bmod q &= p^{-1}p(\sum_{i=1}^{n} m_{i,q})^q \bmod q = \sum_{i=1}^{n} m_{i,q} \bmod q, \\ C_T^{mul} \bmod p &= q^{-1}q(\prod_{i=1}^{n} m_{i,p})^p \bmod p = \prod_{i=1}^{n} m_{i,p} \bmod p, \\ C_T^{mul} \bmod q &= p^{-1}p(\prod_{i=1}^{n} m_{i,q})^q \bmod q = \prod_{i=1}^{n} m_{i,q} \bmod q. \end{aligned} \tag{10}$$

It is noted that the fully homomorphic property is also preserved on both the addition and multiplication operations respectively with modulus p and q as presented above. Then, the receiver can recover the fully homomorphic results M_T^{add} and M_T^{mul} by exploiting the Chinese Remainder Theorem (CRM) on Eqn. (10) as follows,

$$\begin{aligned} M_T^{add} &= M_p^{add'} q M_{T,p}^{add} + M_q^{add'} p M_{T,q}^{add} \bmod N, \\ M_T^{mul} &= M_p^{mul'} q M_{T,p}^{mul} + M_q^{mul'} p M_{T,q}^{mul} \bmod N, \end{aligned} \tag{11}$$

where $M_p^{add'}, M_q^{add'}, M_p^{mul'}, M_q^{mul'}$ respectively satisfies

$$\begin{aligned} M_p^{add'} q \equiv 1 \bmod p, M_q^{add'} p \equiv 1 \bmod q, \\ M_p^{mul'} q \equiv 1 \bmod p, M_q^{mul'} p \equiv 1 \bmod q, \end{aligned} \tag{12}$$

which can be efficiently computed since the greatest common divisor of p and q namely $(p, q) = 1$.

Remark: It is noted that in the proposed scheme, the sender and receiver are assumed to be distinct entities, and permits only the authorized receiver holding sk_f can recover the original function result and verify its correctness. It is cornerstone of our proposed EVOC in Sec. III.B to achieve authorized verification. The scenario that the sender and receiver are the same is a special case of our construction and can also be applied to our EVOC to achieve only private verification (i.e. the existing work [3] merely allows this situation). The correctness of our proposed privacy-preserving data aggregation has been obviously demonstrated in algorithms **AGG.Eval** and **AGG.Dec**. It is proposed based on any one-way trapdoor function f which can be flexibly implemented by kinds of cryptographic primitives such as public key encryptions, identity-based encryptions and attribute-based encryptions to achieve various security and privacy requirements under different network scenarios.

B. Verifiable Outsourced Computation

In this subsection, a more efficient verifiable outsourced computation of encrypted data EVOC from any one-way trapdoor function is proposed, by exploiting the techniques of the privacy-preserving data aggregation proposed in Sec. III.A and the Yao's Garbled Circuit. It mainly comprises the following algorithms **KeyGen**, **ProbGen**, **Compute** and **Verify**.

KeyGen$(F, \lambda) \rightarrow (PK, SK)$: Following Yao's Garbled Circuit construction, represent the function F as a circuit C and randomly selects two values $w_i^0, w_i^1 \in_R \{0,1\}^\lambda$ for each wire w_i to denote the bit values 0 or 1 on that wire, where λ is the security parameter. Then for each gate g with input wires w_i, w_j and output wire w_z, compute the following four ciphertexts

$$\begin{aligned} &\gamma_{00}^g = E_{w_i^0}(E_{w_j^0}(w_z^{g(0,0)})), \gamma_{01}^g = E_{w_i^0}(E_{w_j^1}(w_z^{g(0,1)})), \\ &\gamma_{10}^g = E_{w_i^1}(E_{w_j^0}(w_z^{g(1,0)})), \gamma_{00}^g = E_{w_i^1}(E_{w_j^1}(w_z^{g(1,1)})) \end{aligned} \tag{13}$$

where E is a Yao's secure symmetric encryption scheme with an elusive range. The public key PK will be the set of ciphertexts, namely $PK = \cup_g(\gamma_{00}^g, \gamma_{01}^g, \gamma_{10}^g, \gamma_{11}^g)$ and the secret key will be the values chosen for each wire, namely $SK = \cup_i(w_i^0, w_i^1)$ privately kept by the data owner.

ProbGen$(SK, \vec{x}) \rightarrow \sigma_{\vec{x}}$: Run the algorithm **AGG.KGen** of the privacy-preserving data aggregation proposed in the previous subsection to generate the public parameters PPR and a key pair (pk_f, sk_f) of the underlying one-way trapdoor function. Let $w_i \subset SK$ be the wire values representing the binary expression of $\vec{x}$ and then set $\sigma_{\vec{x}} = AGG.Enc(PPR, pk_f, w_i)$ and $\tau_{\vec{x}} = sk_f$. The data owner additionally computes $C_{1,3} = f(SK)$, compared to the ciphertexts generated in **AGG.Enc** of our data aggregation scheme proposed in Sec. III.A.

Compute$(PK, \sigma_{\vec{x}}) \rightarrow \sigma_y$: The cloud firstly calculates **AGG.Enc**(PPR, pk_f, γ_i) and constructs a circuit Λ that on input w, w', γ outputs $D_w(D_{w'}(\gamma))$, where D is the decryption algorithm of the encryption algorithm used in Yao's Garbled Circuit. Then, the cloud repeatedly computes **AGG.Eval**(Λ,**AGG.Enc**(PPR, pk_f, w_i),pk_f, γ_i) to decrypt the way through the ciphertexts as is the same to the evaluation of Yao's Garbled Circuit. The result is $\sigma_y = AGG.Enc(PPR, pk_f, w_z)$, where w_z refers to the binary representation of the output wire values $y = F(\vec{x})$.

Verify$(SK, \sigma_y) \rightarrow y \cup \perp$: The authorized receiver firstly deciphers SK, p, N using $\tau_{\vec{x}} = sk_f$ and continues to decrypt $AGG.Enc(PPR, pk_f, w_z)$ for w_z by calling **AGG.Dec**. Then, it maps the output wire values to the original output y using SK. If the decryption or mapping process fails, it outputs $\perp$.

Remark: It is required that the underlying secure symmetric encryption E to possess an efficiently verifiable range. Specifically, given the key k, it is possible to efficiently decide whether a specific ciphertext falls in the range of the encryption under k. That is, there exists an efficient machine M such that $M(k, \gamma) = 1$ if and only if $\gamma \in Range_\lambda(k)$. A suggested symmetric encryption scheme E [3] satisfying the above-mentioned properties is presented as follows. Let $\mathcal{F} = \{f_k\}$ be a family of pseudorandom functions, where $f_k : \{0,1\}^\lambda \rightarrow \{0,1\}^{2\lambda}$ for $k \in \{0,1\}^\lambda$. Then,

we define $\gamma = E_k(x) = (r, f_k(r) \oplus x \parallel 0^\lambda)$, where $x \in \{0,1\}^\lambda, r \in_R \{0,1\}^\lambda$, and $x \parallel 0^\lambda$ represents the concatenation of x and 0^λ. This construction is secure under chosen-plaintext attacks (CPA) according to the Bellare and Rogaway's Model [11]. Then, the checking machine can be constructed as follows. Given k and $\gamma = (c_1, c_2)$, it is required to compute $f_k(c_1)$ and verify whether the last λ bits of $f_k(c_1)$ equal to the last λ bits of c_2. If it does, $\gamma \in Range_\lambda(k)$; otherwise, $\gamma \notin Range_\lambda(k)$.

In our proposed verifiable outsourced computation scheme, it is required to perform this check in the encrypted domain $AGG.Enc(PPR, pk_f, k)$ and apply the property of our proposed fully homomorphic aggregation supporting both addition and multiplication operations on the checking machine M. Then, the cloud firstly computes $C_i = AGG.Enc(PPR, pk_f, M(k, \gamma_i))$ and it is noted that there exists only one ciphertext encrypting 1 that correctly adapts to the γ_i. After that, the cloud computes $D_i = AGG.Enc(PPR, pf_f, D_k(\gamma_i))$ using our proposed efficient privacy-preserving data aggregation supporting both addition and multiplication operations in the previous subsection. Finally, the cloud outputs $C_g = AGG.Enc(PPR, pk_f, \Sigma_i M(k, \gamma_i) D_k(\gamma_i))$ as the correct output of gate g in the encrypted form.

IV. Security Proof

In this section, we firstly prove the security of our proposed efficient privacy-preserving data aggregation scheme in Sec. III.A, which serves the basis of the formal security proof of our final verifiable outsourced computation EVOC.

Theorem 1: **Input Privacy** The data owner's individual data is unconditionally-secure from even the collusion between the cloud and the receiver, namely $H(m_i|C_i) = H(m_i)$ where $C_i = (C_1, C_{2,i}, C_{3,i})$, and $H(\cdot)$, $H(\cdot|\cdot)$ respectively refer to the entropy function and the conditional entropy function.

Proof: It is noted that in our proposed construction, the data owner's encrypted individual data $C_{2,i} = q^{-1} q m_{i,p}^p + p^{-1} p m_{i,q}^q + U_i^{add} \ mod \ N$ and $C_{3,i} = (q^{-1} q m_{i,p}^p + p^{-1} p m_{i,q}^q) U_i^{mul} \ mod \ N$. In the ciphertexts $C_{2,i}, C_{3,i}$, $m_{i,p}, m_{i,q}$ are blinded by the secret knowledge of factoring the large integer N which is well protected by the one-way trapdoor function in C_1. On the meanwhile, they are also respectively doubly-blinded by the random numbers U_i^{add}, U_i^{mul} uniformly distributed in $\mathbb{Z}_N$. Consequently, the probability of the adversary to correctly guessing the owner's individual data m_i is the same as the one given the ciphertext C_i. It means our proposed privacy-preserving data aggregation achieves the information-theoretic security for the input/output privacy, namely $H(m_i|C_i) = H(m_i)$ and theorem 1 holds. ■

Theorem 2: **Output Privacy** Let $\mathcal{A}$ be a malicious adversary defeating our proposed privacy-preserving data aggregation with a nonnegligible advantage defined as $\epsilon^{',n(\lambda)}$, where $n(\lambda)$ refers to the total number of queries made to the oracles and λ is the security parameter. There exists a simulator $\mathcal{B}$ who can use $\mathcal{A}$ to invert the one-way trapdoor function with the probability:

$$\epsilon \geq \epsilon^{',n(\lambda)} - \frac{n(\lambda)}{2^{\lambda-1}}. \tag{14}$$

It is obviously observed that Theorem 2 holds since our proposed construction is devised based on the paradigm of Bellare and Rogaway [11] in the random oracle model. The complete proof is detailed in the full version of our paper.

Theorem 3: Let E be a Yao-secure symmetric encryption scheme and AGG be a privacy-preserving data aggregation satisfying input/output privacy and evaluation privacy. Then, our proposed EVOC is a secure, outsourceable and authorized verifiable computation scheme.

Proof: Exploiting the same proof technique in [3], we can straightforwardly demonstrate Yao's garbled circuit scheme is a one-time secure verifiable computation scheme that can be used to compute function F securely on a single input. Then, the only task left is to prove the multiple executions of our construction. Specifically, it is assumed that there exists an adversary $\mathcal{A}$ such that $ADV^{Vef}_{\mathcal{A},VC}(F,\lambda) \geq \varepsilon$ with a non-negligible probability ε, there would be another adversary $\mathcal{A}^{'}$ querying $\mathcal{A}^{'}$ querying $O^{ProbGen}$ only once such that $ADV^{Vef}_{\mathcal{A},VC_{Yao}}(F,\lambda) \geq \varepsilon^{'}$ with a non-negligible $\varepsilon^{'}$.

Let l be the total number of queries $\mathcal{A}$ makes to its $O^{ProbGen}$ and $i \in [1, l]$ be the index randomly selected by $\mathcal{A}^{'}$. A series of games $Game^k_{\mathcal{A},VC}(F,\lambda)(k = 1, \cdots, l-1)$ are defined. The oracle $O^{ProbGen}$ answers the queries from $\mathcal{A}$ as follows.

(1) If $j \leq k \wedge j \neq i$, it randomly chooses $(PK^j_{AGG} = (N, pk_f), SK^j_{AGG} = (sk_f, p))$ and encrypt a random λ-bit string under PK_{AGG};

(2) Else if $j > k \vee j = i$, it responds as the VC performs, encrypting correct input labels in Yao's Garbled Circuit under PK^j_{AGG}.

In the end, the output of $Game^k_{\mathcal{A},VC}$ is 1 if $\mathcal{A}$ successfully i-th input, otherwise is 0. It is observed that $Game^0_{\mathcal{A},VC}$ is identical to $Game^{vef}_{\mathcal{A},VC}(F,\lambda)$ except for the index i selection, therefore $ADV^0_{\mathcal{A},VC}(F,\lambda) = \frac{ADV^{Vef}_{\mathcal{A},VC}(F,\lambda)}{l} \geq \frac{\varepsilon}{l}$. On the other hand, $ADV^{l-1}_{\mathcal{A},VC}(F,\lambda) = ADV^{vef}_{\mathcal{A}',VC_{Yao}}(F,\lambda)$. $ADV^{l-1}_{\mathcal{A},VC}(F,\lambda) = ADV^{vef}_{\mathcal{A}',VC_{Yao}}(F,\lambda)$.

For the intermediate games, it is observed that if we modify $Game^{k-1}_{\mathcal{A},VC}$ by replacing the random label encryption with PK^k_{AGG} and the ciphertexts the encrypted input wire values of x, we will actually run $Game^k_{\mathcal{A},VC}$. Therefore, if the intermediate games can be distinguished from each other with a non-negligible probability, the unconditional security of our proposed privacy-preserving data aggregation will be defeated, which is a contradiction. Consequently, we can derive we can derive

$$ADV^{Vef}_{\mathcal{A},VC_{Yao}}(F,\lambda) \geq \frac{\varepsilon}{l} - \frac{l}{N}. \tag{15}$$

■

The input and output privacy of our proposed EVOC can be deduced in the same way. We leave the proof details in the full version of our paper.

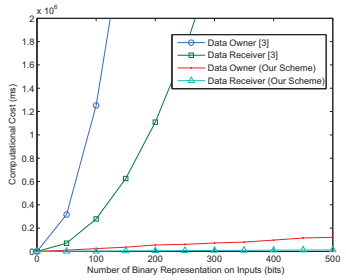

Fig. 2: Computational Cost Comparison on Clients

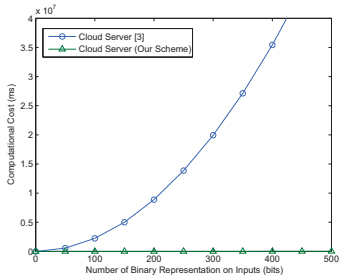

Fig. 3: Computational Cost Comparison on Cloud

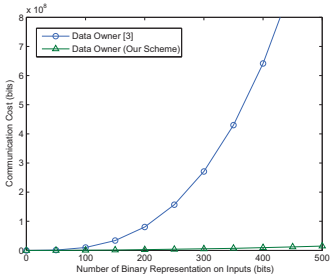

Fig. 4: Communication Cost Comparison on Clients

TABLE I: Efficiency Comparison

Schemes	Primitive	Input Pri.	Output Pri.
Gennaro's[3]	FHE	CPA	CPA
Our Construction	Any OWTF	Unconditionally Secure	CCA2

V. Performance Evaluation

In this section, we mainly evaluate the efficiency advantage over the existing work [3] in terms of computational and communication cost on both the clients (i.e. data owner and receiver) and the cloud ends. It is noted that our newly proposed technique of privacy-preserving data aggregation realizes an efficiency tradeoff by supporting both addition and multiplication operations, perfectly fitting the functionality required by verifiable outsourced computation, but disenabling individual decryption.

We conduct the experiments by exploiting PBC [12] and MIRACLE [13] libraries running on Linux platform with 2.93GHz processor to study the operation costs. Table 1 illustrates the security and privacy property comparisons between [3] and our construction. Figs. 2 and 3 illustrate as the number of binary representation on the function input increases, both the computational cost on the data owner, receiver and the cloud of our proposed verifiable outsourced computation is dramatically slighter than [3]. The reason is that the adopted fully homomorphic encryption [4] in [3] has to encode all inputs x_i and γ_i and realize fully homomorphic aggregation bit by bit; while in our construction, the one-way trapdoor function implemented by RSA is required to perform only once to achieve both additional and multiplicative aggregation. Fig. 4 demonstrate the communication cost on data owner of our construction is also significantly less than [3] since the ciphertext size is reduced by avoiding bit-by-bit encryption.

VI. Conclusion

In this paper, a more efficient verifiable outsourced computation of encrypted data EVOC from any one-way trapdoor function is proposed by combining a newly devised privacy-preserving data aggregation supporting both addition and multiplication operations with Yao's Garbled Circuit. Formal security proof and extensive evaluations demonstrate our proposed EVOC satisfies the target security and privacy requirements of verifiable outsourced computation and the efficiency advantage over the state-of-the-art in terms of both computational and communication overhead.

Acknowledgment

This work was supported by the National Program on Key Basic Research Project (973 Program) (Grant No. 2012CB723401), the National Natural Science Foundation of China (Grant No. 61411146001, 61373154 and 61371083) and the Prioritized Development Projects of the Specialized Research Fund for the Doctoral Program of Higher Education of China (Grant No. 20130073130004).

References

[1] *Waht is Cloud Computing*, Amazon web services. 2013-03-19. Retrieved 2013-03-20.
[2] C. Castelluccia, A.C.-F. Chan, E. Mykletun and G. Tsudik, *Efficient and provably secure aggregatioin of encrypted data in wireless sensor networks*, ACM Trans. Sen. Netw. 5(20): 1-36, 2009.
[3] R. Gennaro, C. Gentry and B. Parno, *Non-interactive verifiable computing: outsourcing computation to untrusted workers*, In: CRYPTO 2010.
[4] K. Lauter, M. Naehrig and V. Vaikuntanathan, *Can Homomorphic Encryption Be Practical?*, In: ACM CCSW 2011.
[5] J. Zhou, Z. Cao, X. Dong, N. Xiong and A.V. Vasilakos, *4S: A Secure and Privacy-preserving Key Management Scheme for Cloud-assisted Wireless Body Area Network in m-Healthcare Social Networks*, Information Sciences, DOI:10.1016/j.ins.2014.09.003, to appear.
[6] M. Barbosa and P. Farshim, *Delegatable homomorphic encryption with applications to secure outsourcing of computation*, In: CT-RSA 2012, LNCS 7178, pp. 296-312. Springer, Heidelberg (2012).
[7] C. Papamanthou, E. Shi and R. Tamassia, *Signatures of correct computation*, In: TCC 2013.
[8] B. Parno, J. Howell, C. Gentry and M. Raykova, *Pinocchio: Nearly Practical Verifiable Computation*, In: IEEE Symposium on Security and Privacy (IEEE S&P 2013), pp. 238-252, 2013.
[9] J. Zhou, X. Lin, X. Dong and Z. Cao, *PSMPA: Patient Self-controllable and Multi-level Privacy-preserving Cooperative Authentication in Distributed m-Healthcare Cloud Computing System*, IEEE Transactions on Parallel and Distributed Systems, DOI: 10.1109/TPDS.2014.2314119, to appear.
[10] J. Zhou, X. Dong, Z. Cao and A. V. Vasilakos, *Secure and Privacy Preserving Protocol for Cloud-based Vehicular DTNs*, IEEE Transactions on Information Forensics and Security, to appear.
[11] M. Bellare and P. Rogaway, *Random oracles are practical: a paradigm for designing efficient protocols*, In: ACM Conference on Computer and Communications Security, 1993.
[12] B. Lynn, *PBC Library*, http://crypto.stanford.edu/pbc/.
[13] *Multiprecision integer and rational arithmetic c/c++ library*, http://www.shamus.ie/.
[14] J. Zhou, Z. Cao, X. Dong, X. Lin and A. V. Vasilakos, *Securing m-healthcare social networks: challenges, countermeasures and future directions*, IEEE Wireless Communications, vol. 10, no. 4, pp. 12-21, 2013.