Privacy-preserving Outsourcing Schemes of Modular Exponentiations Using Single Untrusted Cloud Server

Ling Zhao[1], Mingwu Zhang[1,*], Hua Shen[1], Yudi Zhang[1] and Jian Shen[2]
[1] School of Computer, Hubei University of Technology
Wuhan 430068, Hubei - P.R.China
[e-mail: zhaoling_0@126.com]
[2] School of Computer and Software, Nanjing University of Information Science and Technology
Nanjing 210044, Jiangsu - P.R.China
*Corresponding author: Mingwu Zhang

Received July 24, 2016; revised November 17, 2016; accepted December 23, 2016; published February 28, 2017

Abstract

Outsourcing computation is one of the most important applications in cloud computing, and it has a huge ability to satisfy the demand of data centers. Modular exponentiation computation, broadly used in the cryptographic protocols, has been recognized as one of the most time-consuming calculation operations in cryptosystems. Previously, modular exponentiations can be securely outsourced by using two untrusted cloud servers. In this paper, we present two practical and secure outsourcing modular exponentiations schemes that support only one untrusted cloud server. Explicitly, we make the base and the index blind by putting them into a matrix before send to the cloud server. Our schemes provide better performance in higher efficiency and flexible checkability which support single cloud server. Additionally, there exists another advantage of our schemes that the schemes are proved to be secure and effective without any cryptographic assumptions.

Keywords: cloud computing, modular exponentiations, outsourcing computation, checkability, efficiency

This work is supported by the National Natural Science Foundation of China under grants 61370224 and 61672010, the Key Laboratory of Mathematics and Interdisciplinary Sciences of Guangdong Higher Education Institutes, Guangzhou University, and the CICAEET fund and the PAPD fund..

https://doi.org/10.3837/tiis.2017.02.011 **ISSN : 1976-7277**

1. Introduction

Cloud computing is a kind of Internet-based computing that provides shared processing resource and data to computers and other devices on demand. As we know, cloud computing is a combination and development of Parallel Computing, Distributed Computing, Grid Computing, Network Storage Technologies, Virtualization and Load Balance.

With the advent of the data age, people can enjoy increasing new experience brought by network, but it also caused a series of resource constraints problems (i.e., computing resources, storage resources and bandwidth). Cloud computing, help people to realize the dream of treating computer as infrastructure, has become a research focus in academia, industry and the government. Cloud computing grows quickly and has the widespread application in various trades and occupations, which is being used in extensive fields such as medical and pharmacy (Ming Li et al. [1]), E-government (Ali O, Soar J and Yong J [2]), mobile network (Han N D, Chung Y and Jo M [3]) and data storage (Patel H B et al. [4]). And a number of researches have been made for ensuring the security of networks and cloud data, see for example [5, 6].

Outsourcing computation [7] is one of the most important applications in cloud computing. For example, a client, who has limited resources (i.e., computing power and storing space), can outsource some complex and time-consuming tasks to cloud servers. In this way, the client can improve efficiency of calculation and save resources.

However, in the process of outsourcing computing there exits some new security concerns and challenges. We know that the client may be deceived by the cloud due to the internal structure and the operational details of the cloud server are not transparent enough to customers. The cloud server has motivation of behaving unfaithfully and returning incorrect results. When customers outsource a huge computing task to the cloud, there is a big problem whether the cloud server would be “lazy”, especially if customers cannot distinguish the correctness of computation results. In this case, the cloud server would return incorrect or inconsistent results. On the other hand, there might also be a consideration that malicious outsider attackers can monitor the process of computing. The adversaries are likely to gather or tamper personal information, in order to obtain additional information, commercial interests, and intangible needs, to make the outsource tasks fail of security guarantee. How to protect user’s privacy and ensure the correctness of calculation results is a circuital problem. Verifiable Computation (VC) [8] can solve these problems well, in which there exists a secure two-party protocol between a client and a cloud server. The private information is preserved to cloud servers, while users can also verify the correctness of the calculation results after cloud servers return the results.

Modular exponentiation [9] is widely used in the field of cryptography, mainly used in public-key cryptography (RSA, ELGamal) [10], digital signature schemes [11] and one-way hash functions [12], including their applications [13] and so on, as well as being viewed as one of the most time-consuming operation. However, for a n-bit exponent, the traditional square-and-multiply method requires $1.5n$ modular multiplications on average,

where n is the index of a bit length. Thus, the modular exponentiation is a time consuming operation for limited computation resources.

In this article we make a study on how to secure outsource modular exponentiations to single untrusted cloud server. As we know, some special functions, such as modular exponentiation [13, 14, 15], polynomial evaluation [16] and attribute-based encryption [17], are difficult to calculate, especially for the client who has limited resources. Most of present outsourcing computation schemes of modular exponentiation are based on two untrusted cloud servers. However, to ensure that there is no collusion between two untrusted cloud servers is difficult in the real-world. To solve this problem, we propose the privacy-preserving outsourcing scheme of modular exponentiations using single untrusted cloud server. Our schemes not only reduce the number of cloud servers, but also increase the efficiency and verifiably. In this paper, we analyze our schemes and made comparison with the existing schemes.

1.1 Related Work

In the cloud computing environment, users do not have the capacity to complete some complex scientific calculators or cryptographic computations due to their limited computing power and storage space. Secure outsourcing computation has long been an important issue in the academic circles. In 2001, Atallah et al. [18] researched securely outsourcing on large-scale numerical analysis for the first time by adopting various camouflage and blinding technologies. Then the security outsourcing framework which applied in many kinds of scientific computations was put forward. The framework, however, did not take the verifiability of results into account.

In the cryptographic community, Chaum and Pedersen firstly put forward the notion of wallets with observers, a piece of secure hardware installed on the user's computer to perform some expensive computations. After that, many outsourcing schemes were proposed in [17, 19, 20]. Jakobsson and Wetzel [21] proposed a secure outsource signature by using outsourcing modular exponentiations, in which the user had to blind the exponent before send it to cloud, and it is only efficient for batch exponentiations with small size. During the implementation of these schemes, the user needed to compute and store several exponentiations as verification values. A lot of works have been done for improving the speed of modular exponentiations. Hohenberger and Lysyanskaya [15] proposed a secure outsourcing model of modular based on two untrusted cloud servers, and the computing time complexity is O ($log^2 n$). Dijk et al. [22] proposed some schemes for speeding up fixed base-variable exponent exponentiation and variable base-fixed exponent exponentiation by using a single untrusted cloud server. In their protocols, they need $3(logn-1)/2$ over the square-and multiply algorithm. In 2013, Ma et al. [23] presented two secure outsourcing schemes of modular exponentiations. The first was a secure outsourcing scheme for fixed base-variable exponent exponentiation, while the second was for variable base-fixed exponent. Chen et al. [13] put forward a new secure outsourcing of modular exponentiation, which is also based on two non-collusion untrusted servers, in terms of computational efficiency and verifiable had been upgraded than Hohenberger's, the verifiability of Hohenberger's scheme is 1/2 and that of Chen's is increased to 2/3. Both of these schemes

guarantee the privacy of the index and the base. But they need plenty of complex precomputations before outsourcing computing tasks, and thus made these schemes not efficient enough. Meanwhile, their schemes require two untrusted cloud servers, which cannot resist the collusive attack of the two cloud servers. Then, some outsourcing schemes were proposed which based on only one untrusted server, in which the index and the base are secret to cloud server [24, 25].

1.2 Contributions

In this paper, we mainly focus on how to improve the efficiency and security when outsourcing modular exponentiations to the untrusted cloud servers. We propose two new secure outsourcing schemes of modular exponentiation for single server. One is the proposed secure outsourcing scheme Exp for variable base-variable exponent exponentiation, the other is a new outsourcing scheme Sexp for simultaneous modular exponentiations. In our schemes the sensitive information would not be exposed to the server, and the client can detect whether the server is deceptive or not. Our schemes can achieve security and privacy of the base and the index by putting values into a large matrix. Compared with related schemes, our schemes only need few amount of precomputation and provide an improvement efficient of outsourcing.

Similar to [13, 14], we also make some logical divisions on the base and the index so that the server could not get information about them. The difference is that the approach used in our scheme is matrix blind which can archive higher security and efficiency. Moreover, our schemes are secure outsourcing of modular exponentiations by using single untrusted cloud server and do not need any cryptographic assumption.

1.3 Organization

The paper is organized as follows. We give the system model for outsourcing computation in section 2. In this paper, we propose two secure modular exponentiation outsourcing schemes. The algorithm for modular exponentiations Exp which makes $u^a \bmod p$ is a secure outsourcing scheme in the proposed model and its security analysis and the comparison with others are presented in section 3. In section 4, we propose the algorithm for simultaneous modular exponentiations (Sexp) which makes ${u_1}^a {u_2}^b \bmod p$ is a secure outsourcing scheme in the proposed model. Finally, we conclude the paper in section 5.

2. Preliminaries

The VC scheme, basic tool and system model are formally defined in this section.

2.1 VC scheme

Verifiable outsourcing computation is a scheme which can verify the correctness of outsourcing results. User received results after cloud server completes the privacy-preserving outsourcing computation, and there is a verification request message sent by user to cloud

server, and the cloud server will return some evidences for user to verify the correctness. The user observes the verified results before deciding whether outsourcing compute results are correct or not. Generally speaking, the VC scheme processes as shown in the **Fig. 1**.

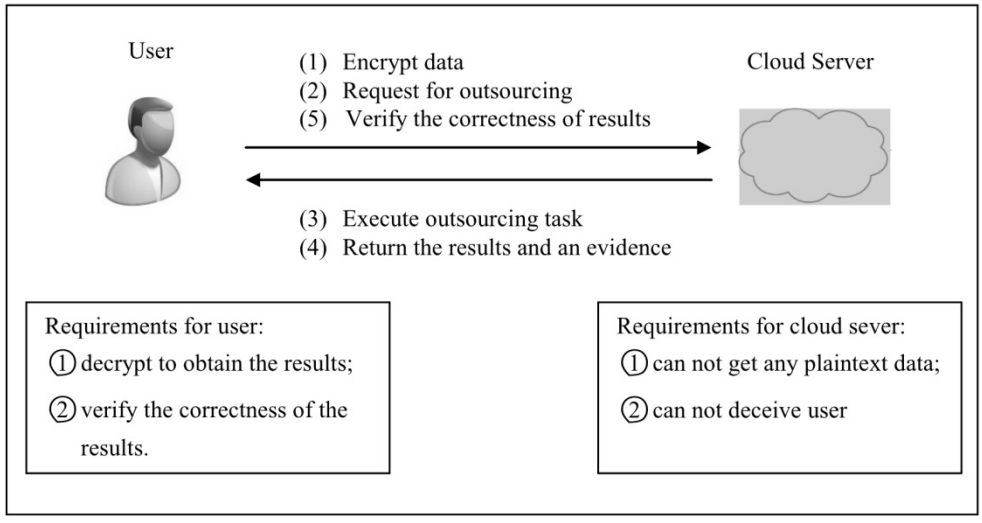


Fig. 1. VC scheme

The VC scheme can not only protect the correctness of results, but also has the specific properties of non-repudiation and anti-counterfeiting. In recent years, several kinds of the VC schemes put forward by many scholars around the world. Moreover, the verification efficiency is gradually raised. We can verify the correctness of the results efficiently by using verifiable homomorphism encryption technology. The non-interactive verifiable computation was first proposed by Gennaro et al. in 2010 [8]. In several years, this scheme was extended to multi-user environment by the agent oblivious transfer technology [26]. A new method for making the verifiable outsourcing computation was put forward by Parno et al. [17]. As we know, there are many studies that specific to verifiability in all kinds of common algorithms (i.e., bilinear pairings, sequence comparison, matrix multiplication and inverse operation) have been done over the years [27, 28, 29].

2.2 Rand tool

In the paper, we invoke a subroutine named *Rand* which can generate random number pairs to speed up the computations. In the construction from [15], the inputs for *Rand* are a secure prime p, a base $g \in Z_p^*$, or some other values. The outputs for *Rand* are some random invocation lists, these are independent pairs of the form $(b, g^b mod\ p)$, where $b \in Z_q$.

According to [15], there are two approaches to implement the subroutine *Rand*. The first method is to compute a table of random, independent pairs and triplets in advance implemented by a trusted server and load the table into T's memory. When T wants to invoke

Rand, T simply retrieves the next value in the table. The second method is use the EBPV generator designed by Nguyen, Shparlinski and Stern [30] to obtain the independent random number pairs. The EBPV generator, which adds a feedback extension (i.e., reuse of the output pairs) to the BPV generator proposed by Boyko, Peinado and Venkatesan [20], which works by taking a subset of truly random (k, g^k) pairs and combing them with a random walk on expanders on Cayley graphs to reduce the dependency of the pairs in the output sequence, runs in time $O(log^2 n)$ for a n-bit exponent. Note that it can resist on adaptive attack from adversary in this way. We adopt the EBPV generator to implement *Rand* in this paper.

2.3 System model

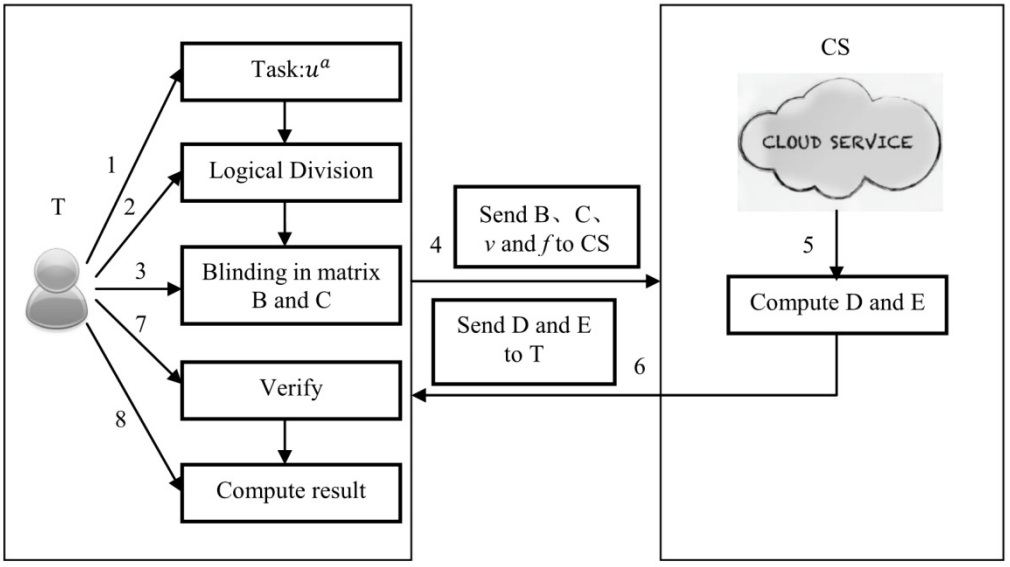


Fig. 2. Proposed model

There are two participants in the scheme, one is the user T, the other is an untrusted cloud server CS. The specific description of the proposed model works as shown in **Fig. 2.**

User T provides u as the base and a as the exponent of modular exponentiation which will be computed in CS. At the same time, T performs some transformations and blinds before sends these original values to the cloud server CS. Firstly, T makes some logical divisions on the base u and the exponent a, and puts these numbers into matrix B and matrix C (we assume that B and C are large enough in our schemes). Then T sends these transformation values to CS. CS makes computations on these values after receiving these transformation values, and returns the results to T. Finally, T verifies the correctness and consistency of the calculation results. T will complete the final calculation operation to get the result using the return values when the verifications are right. Once failed in verifications, T will abort the protocol.

3. Outsourcing scheme of variable base-variable exponent exponentiation

3.1 Outsourcing scheme of Exp

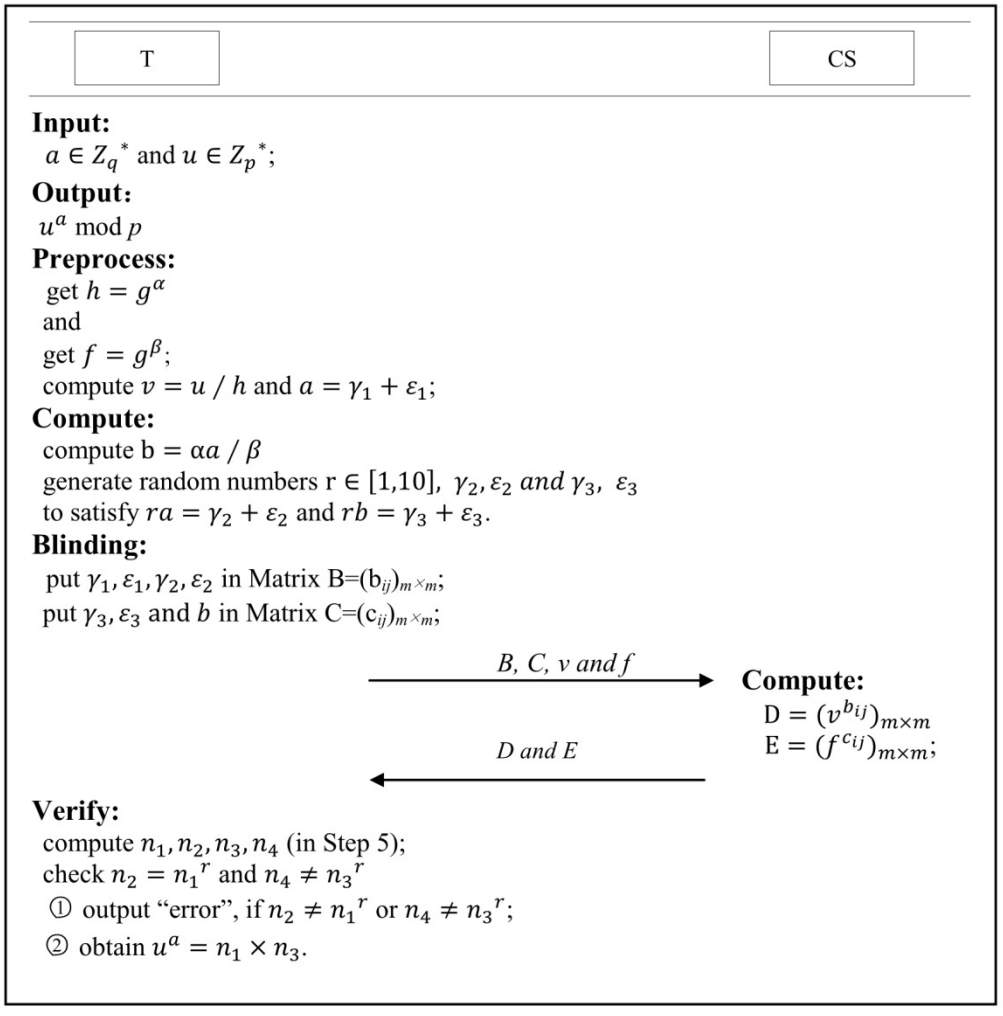


Fig. 3. Our scheme Exp

Let p, q be two large primes and $q|p-1$. Similar to [13, 15], the inputs of Exp scheme are $a \in {Z_q}^*$ and $u \in {Z_p}^*$, such that $u^q = 1 \, mod \, p$. The output of Exp is $u^a \, mod \, p$. Actually, all the computations are executed within the cyclic group G_p and G_q, where p and q are large primes, and g is the generator of G_p. Let T be the client who wants to outsource the computation task to cloud server, and CS be an untrusted cloud server who has enough computing resources. In this scheme, both of a and u are computationally blinded to CS. In

the following scheme Exp, we make some logical divisions on the basis of the secure outsourcing scheme of modular exponentiation proposed by Hohenberger et al. in 2005 and made some improvements on theirs.

The Exp scheme is described as follows:

Step1. *Run Rand*

T firstly runs *Rand* twice to obtain two blinding pairs (α, g^{α}) and (β, g^{β}). Then T expresses them as $h = g^{\alpha} \bmod p$ and $f = g^{\beta} \bmod p$.

Step2. *Logical division*

$$u^{a} = (v \cdot g^{\alpha})^{a} = v^{\gamma_1} v^{\varepsilon_1} g^{\alpha a},$$

where $v = u/h$, $a = \gamma_1 + \varepsilon_1$.

T computes b to obtain the equation $\alpha a = \beta b$. Then randomly chooses a small number *r* (e.g. $r \in [1,10]$), and randomly chooses γ_2, ε_2 and γ_3, ε_3 to satisfy equations $ra = \gamma_2 + \varepsilon_2$ and $rb = \gamma_3 + \varepsilon_3$.

Step3. *Blinding in matrix*

T puts four numbers $\gamma_1, \varepsilon_1, \gamma_2, \varepsilon_2$ into a matrix $B_{m_1 \times m_2}$ (for simplicity, we assume $m_1 = m_2 = m$) and three numbers γ_3, ε_3 and b into a matrix $C_{m_1 \times m_2}$ (for simplicity, we assume $m_1 = m_2 = m$) that were obtained in step 2. The seven numbers' positions are determined by the user T. The positions of these numbers are T's privacy information that cannot be revealed to CS. We assume that $b_{11} = \gamma_1, c_{25} = b, b_{33} = \varepsilon_2, b_{41} = \varepsilon_1, b_{54} = \gamma_2, c_{62} = \gamma_3, c_{75} = \varepsilon_3$, that

$$\mathrm{B} = \begin{bmatrix} \gamma_1 & * & * & * & * & * & * & \cdots & * \\ * & * & * & * & * & * & * & \cdots & * \\ * & * & \varepsilon_2 & * & * & * & * & \cdots & * \\ \varepsilon_1 & * & * & * & * & * & * & \cdots & * \\ * & * & * & \gamma_2 & * & * & * & \cdots & * \\ * & * & * & * & * & * & * & \cdots & * \\ * & * & * & * & * & * & * & \cdots & * \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ * & * & * & * & * & * & * & \cdots & * \end{bmatrix}_{m \times m} \quad \mathrm{C} = \begin{bmatrix} * & * & * & * & * & * & * & \cdots & * \\ * & * & * & * & b & * & * & \cdots & * \\ * & * & * & * & * & * & * & \cdots & * \\ * & * & * & * & * & * & * & \cdots & * \\ * & * & * & * & * & * & * & \cdots & * \\ * & \gamma_3 & * & * & * & * & * & \cdots & * \\ * & * & * & * & \varepsilon_3 & * & * & \cdots & * \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ * & * & * & * & * & * & * & \cdots & * \end{bmatrix}_{m \times m}$$

Then, T sends B, C, v and f to CS.

Step4. *Computation*

After receiving B, C, *v* and *f*, CS computes $\mathrm{D} = (v^{b_{ij}})_{m \times m}$ and $\mathrm{E} = (f^{c_{ij}})_{m \times m}$.

$$D = \begin{bmatrix} v^{\gamma_1} & * & * & * & * & * & * & \cdots & * \\ * & * & * & * & * & * & * & \cdots & * \\ * & * & v^{\varepsilon_2} & * & * & * & * & \cdots & * \\ v^{\varepsilon_1} & * & * & * & * & * & * & \cdots & * \\ * & * & * & v^{\gamma_2} & * & * & * & \cdots & * \\ * & * & * & * & * & * & * & \cdots & * \\ * & * & * & * & * & * & * & \cdots & * \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ * & * & * & * & * & * & * & \cdots & * \end{bmatrix}_{m\times m} \quad E = \begin{bmatrix} * & * & * & * & * & * & * & \cdots & * \\ * & * & * & * & f^{b} & * & * & \cdots & * \\ * & * & * & * & * & * & * & \cdots & * \\ * & * & * & * & * & * & * & \cdots & * \\ * & * & * & * & * & * & * & \cdots & * \\ * & f^{\gamma_3} & * & * & * & * & * & \cdots & * \\ * & * & * & * & f^{\varepsilon_3} & * & * & \cdots & * \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ * & * & * & * & * & * & * & \cdots & * \end{bmatrix}_{m\times m}$$

Then, CS sends D and E to user T.

Step5. *Verification*

T computes $n_1 = d_{11} \cdot d_{41}$, $n_2 = d_{54} \cdot d_{33}$, $n_3 = e_{25}, n_4 = e_{62} \cdot e_{75}$.

T verifies the correctness of the calculation results by testing two equations $n_2 = {n_1}^r$ and $n_4 = {n_3}^r$. If not, T outputs "error" and aborts the protocol.

Otherwise, T obtains $u^a = n_1 \times n_3$.

Remark: In proposed model, we randomly placed these transformation values about u and a into two big enough matrixes B and C before send them to CS. Anyone cannot get information about computation task from matrixes except T. Moreover, we designed the scheme by using one untrusted cloud server. These are the major differences from the technique in [13] and [15].

3.2 Scheme Analysis

A secure outsourcing computation scheme should satisfy the following conditions.

Correctness: Any honest cloud server must produce an output that can be decrypted and verified successfully by the user.

Soundness: Any incorrect results generated by the dishonest cloud server can be decrypted and verified successfully by the user with negligible probability.

Input/output privacy: No sensitive information about the user's private data can be recovered or regained by the cloud server during executing the protocol.

Verifiability: Users can verify the correctness of the outputs returned by the honest cloud server with great probability, and can also verify the incorrectness of the outputs returned by the dishonest cloud server.

Efficiency: The computation burdens that user need under this scheme should not more than computation burdens that the user computes by himself.

We refer to the method of calculating mathematical probability which used in the papers about matrix calculation [28, 31] to analyze the performance and security of our scheme.

Theorem 1: Our scheme Exp is secure and verifiable which supports only single untrusted cloud server.

Proof: The outsourcing scheme Exp satisfies correctness, soundness, privacy, verifiability and efficiency, which are proved as follows.

- *Correctness:* In our scheme, if the cloud server CS honestly follows the protocol, the outputs of CS will be correct and accepted by the user T.
  Since $v = u/h$, $a = \gamma_1 + \varepsilon_1$, then $n_1 = v^{\gamma_1}v^{\varepsilon_1} = v^a$.
  According to $ra = \gamma_2 + \varepsilon_2$, we can know $n_2 = v^{\gamma_2}v^{\varepsilon_2} = v^{ra}$, therefore $n_2 = {n_1}^r$.
  According to $rb = \gamma_3 + \varepsilon_3$, we can know $n_4 = f^{\gamma_2}f^{\varepsilon_2} = f^{rb}$, therefore $n_4 = {n_3}^r$.
  It can be seen that the verification equation will establish if the cloud server is honest.
  Since $u^a = (v \cdot g^\alpha)^a = v^{\gamma_1}v^{\varepsilon_1}g^{\alpha a} = v^{\gamma_1}v^{\varepsilon_1}f^b \ mod\ p$, then $u^a = n_1 \cdot n_3$ establish as well.
- *Soundness:* There are three types of dishonest behaviors of CS. In the first type, the CS may change the values in the matrix B and C. In this case, if CS wants to pass the two verification equations $n_2 = {n_1}^r(d_{54} \cdot d_{33} = (d_{11} \cdot d_{41})^r) and\ n_4 = {n_3}^r(e_{62} \cdot e_{75} = {e_{25}}^r)$ to influence the correctness of u^a, then CS must guess the right positions of $(\gamma_1, \varepsilon_1, \gamma_2, \varepsilon_2)$ in matrix B, $(\gamma_3, \varepsilon_3, b)$ in matrix C and obtain the value of *r* at the same time. Obviously, the matrix B has m rows and m columns, so there are m^2 elements in the matrix. The probability that the CS finds the right positions of $(\gamma_1, \varepsilon_1, \gamma_2, \varepsilon_2)$ in matrix B is $\frac{1}{\mathrm{m}^2(\mathrm{m}^2-1)(\mathrm{m}^2-2)(\mathrm{m}^2-3)}$, that is $\frac{(\mathrm{m}^2-4)!}{\mathrm{m}^2!}$. We can also obtain the probability that CS finds the right positions of $(\gamma_3, \varepsilon_3, \mathrm{b})$ in matrix C is $\frac{1}{\mathrm{m}^2(\mathrm{m}^2-1)(\mathrm{m}^2-2)}$ and the probability that CS can obtain r is $\frac{1}{10}$. It can be seen that the probability in the first type is $\frac{(\mathrm{m}^2-4)!}{10\mathrm{m}^2!\mathrm{m}^2(\mathrm{m}^2-1)(\mathrm{m}^2-2)}$.
  In the second type, CS may change the values in matrix D and matrix E. In this case, the value of u^a will not be affected since CS does not change the values of $(d_{11}, d_{41}, d_{54}, d_{33})$ in matrix D and (e_{25}, e_{62}, e_{75}) in matrix E. Otherwise, CS must guess all of these values' positions and obtain the value of *r* if CS wants to pass through the verification in step 5. The probability that the CS guesses the right positions of all of the seven numbers in both D and E is $\frac{1}{m^2(m^2-1)(m^2-2)(m^2-3)} \times \frac{1}{m^2(m^2-1)(m^2-2)}$, that is $\frac{(m^2-4)!}{m^2!m^2(m^2-1)(m^2-2)}$, and the probability CS can obtain r is $\frac{1}{10}$. It can be seen that the probability in the second type is $\frac{(m^2-4)!}{10m^2!m^2(m^2-1)(m^2-2)}$.
  In the third type, CS randomly chooses two numbers v' and f', s.t. $v' \neq v$, $f' \neq f$, when CS computes D and E. Then CS returns two revised matrix C′ and D′ to T. In this case, if CS can pass the verification in step 5, it has to guess four numbers $(d_{11}, d_{41}, d_{54}, d_{33})$ in D and $(\mathrm{e}_{25}, \mathrm{e}_{62}, \mathrm{e}_{75})$ in E, at the same time CS needs to obtain the right value of *r*. The probability that the CS finds the right positions is $\frac{(m^2-4)!}{m^2!m^2(m^2-1)(m^2-2)}$, and the probability CS can obtain *r* is $\frac{1}{10}$. It can be seen that the probability in the third type is $\frac{(m^2-4)!}{10m^2!m^2(m^2-1)(m^2-2)}$.

It is easily to see that the probability of these outputs that a dishonest cloud server can pass the last verification is almost negligible.

- *Input/output Privacy:* Note that a, u and u^a are privacy information for the user. We will not leak any valuable information during executing the protocol. We only send the matrix B, matrix C, *v* and *f* to the cloud server. If the CS or an attacker wants to recover a, then they need to obtain the values of γ_1 and ε_1 in the same time, and the successful probability is $\frac{1}{m^4}$. Similarly, if the CS or an attacker wants to recover u, then they need to get the right value of *h*. However, *h* is generated randomly by *Rand* and computationally blinded to the CS or the attacker. Thus the successful probability is almost negligible. In addition, if the CS or the attacker wants to obtain u^a, he must obtain the values of d_{11} , d_{41} and e_{25} at the same time. Obviously, in this case the probability is $\frac{1}{m^6}$.

  That is, the probability for the untrusted CS or attacker recovers some private data form the scheme is at most $\frac{1}{\mathrm{m}^6}$.
- *Verifiability:* We assume that the cloud server is controlled by a PPT attacker, which will obtains all the information that the cloud server receives and computes. Let X be the event of "CS can pass verification equations $d_{54} \cdot d_{33} = (d_{11} \cdot d_{41})^r$ and $e_{62} \cdot e_{75} = {e_{25}}^r$ and affect the correctness of u^a". When it comes to the first type of dishonest behaviors described in soundness, the probability of X is Pr[X]= $\frac{(m^2-4)!}{10m^2!m^2(m^2-1)(m^2-2)}$ for the PPT attacker. When it comes to the second type of dishonest behaviors described in soundness, the probability of X is Pr[X]= $\frac{(m^2-4)!}{10m^2!m^2(m^2-1)(m^2-2)}$ for the PPT attacker. $\frac{(m^2-4)!}{10m^2!m^2(m^2-1)(m^2-2)}$. When it comes to the third type of dishonest behaviors described in soundness, the probability of X is Pr[X]= $\frac{(m^2-4)!}{10m^2!m^2(m^2-1)(m^2-2)}$ for the PPT attacker.

  Let Y be the event of "attacker can forge results and successfully cheat user, and the correctness of u^a can be affected". Thus the probability of Y is $\Pr(Y) \leq \frac{3(m^2-4)!}{10m^2!m^2(m^2-1)(m^2-2)}$. That is, the user can detect the incorrectness with probability at least $1 - \frac{3(m^2-4)!}{10m^2!m^2(m^2-1)(m^2-2)}$. We notice that the checkability is approximately equal to one when the matrix size *m* is large enough.
- *Efficiency:* In general, it takes roughly 1.5n MM to compute $u^a \ mod \ p$ by the square-and-multiply method, where *n* is the bit of a. Also, Exp scheme takes O ($log^2 n$) MM using the EBPV generator or O (1) using the table-lookup method, respectively. Our scheme Exp requires 8 MM, 2 MInv and 2 invocation of *Rand*, and our schemes only need to rent one untrusted cloud server. In addition, our schemes don't need any cryptographic assumptions.

3.3 Comparison

We now compare our scheme Exp with the related schemes in [15] and [13]. In general, u^a can be computed by user with O (n) multiplications, where n is the bit lengths of exponent a. Hohenberger and Lysyanskaya [15] firstly proposed secure outsourcing scheme for variable base-variable exponent modular exponentiation by using two non- collusion untrusted cloud servers. In their scheme, the subroutine *Rand* was invocated many times, and the user requires O($log^2 n$) or O(1) MM using the EBPV generator or table-lookup method respectively. The scheme had checkability with only 1/2. Chen et al. [13] proposed a new secure outsourcing scheme for modular exponentiation by using two non-collusion cloud servers. Their algorithm was superior in both efficiency and checkability than [15], but their scheme had checkability with only 2/3. Then, a lot of scholars proposed secure outsourcing schemes for modular exponentiations [23, 32, 33].

Table 1. Comparison of other schemes vs our Exp

Scheme	[15]	[13].Exp	Our Exp
MM	9	7	8
MInv	5	3	2
Invoke(Rand)	6	5	2
Matrix blinding	No	No	Yes
support single CS	No	No	Yes
Checkability	1/2	2/3	$\geq 1 - 1/0(\mathrm{m}^{14})$

We propose a secure outsourcing scheme for variable base-variable exponent exponentiation by using single untrusted cloud server. The checkability of our Exp is greater than $1 - 1/0(m^{14})$, and the checkability is approximately equal to one when the matrix size m is large enough. Thus our scheme can improve the performance when compared with other proposed schemes. In addition, we adopt the method of matrix blinding in our scheme Exp to achieve privacy-preserving. **Table 1** shows that our scheme Exp makes 2calls to *Rand* plus 5 modular multiplications (MM) and 2 modular inverses (MInv) in order to compute $u^a \ mod \ p$. Other operations such as modular additions are omitted.

Meanwhile, our scheme can ensure the user's private information and calculate the correct result successfully, and the efficiency has an improvement advantage compared with the previous schemes. According to the above comparison and analysis, we can see that our scheme Exp indeed improves the performance and security.

4. Secure outsourcing scheme of simultaneous exponentiation

4.1 Outsourcing scheme of Sexp

In this section, we provide a secure outsourcing scheme for simultaneous exponentiation. Let p, q be two large primes and $q|p-1$. The inputs of Sexp are $a, b \in {Z_q}^*$ and $u_1, u_2 \in {Z_p}^*$, such that ${u_1}^q = 1 \ mod \ p$ and ${u_2}^q = 1 \ mod \ p$. The output of Sexp is ${u_1}^a {u_2}^b \ mod \ p$.

In this scheme, both of a, b and u_1, u_2 are computationally blinded to CS. Here we will also adopt the method of logical divisions which mentioned in [13, 15].

The detailed process of outsourcing scheme Sexp is described as below:

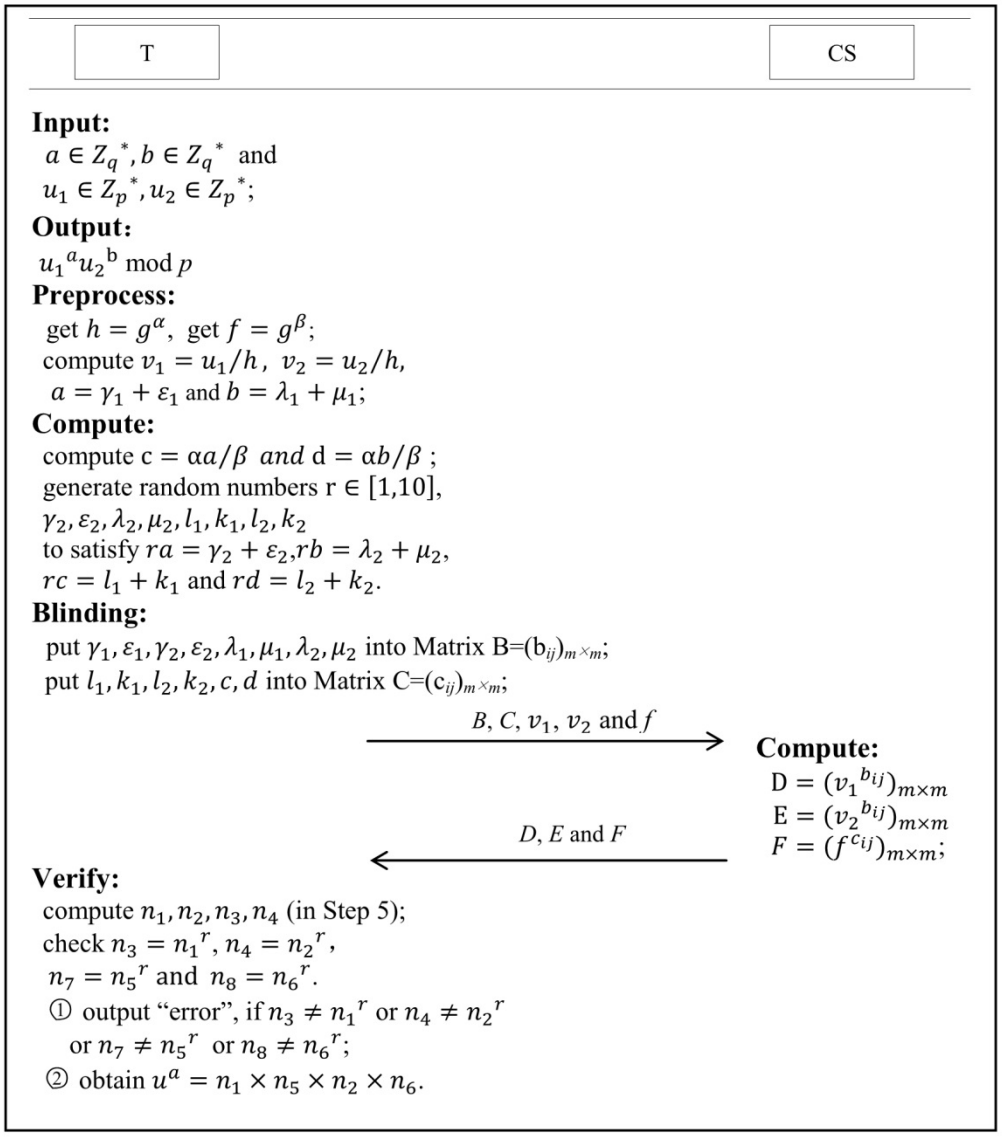


Fig. 4. Our scheme Sexp

Step1. *Run Rand*

T firstly runs *Rand* twice to obtain two blinding pairs (α, g^α) and (β, g^β).

Then T expresses them as $h = g^{\alpha} \bmod p$ and $f = g^{\beta} \bmod p$.

Setp2. *Logical division*

$$u_1{}^{a} u_2{}^{\mathrm{b}} = (v_1 \cdot g^{\alpha})^{a} (v_2 \cdot g^{\alpha})^{\mathrm{b}} = v_1{}^{\gamma_1} v_1{}^{\varepsilon_1} g^{\alpha a} \cdot v_2{}^{\lambda_1} v_2{}^{\mu_1} g^{\alpha b},$$

where $v_1 = u_1/h$, $v_2 = u_2/h$, $a = \gamma_1 + \varepsilon_1$ and $b = \lambda_1 + \mu_1$.

T computes c and d to get the equations $\alpha a = \beta c$ and $\alpha b = \beta d$, and then randomly chosen a small number r (e.g. $r \in [1,10]$), and randomly chosen eight numbers $\gamma_2, \varepsilon_2, \lambda_2, \mu_2, l_1, k_1, l_2, k_2$, such that $ra = \gamma_2 + \varepsilon_2$, $rb = \lambda_2 + \mu_2$, $rc = l_1 + k_1$ and $rd = l_2 + k_2$.

Step3. *Blinding in matrix*

T puts these numbers $\gamma_1, \varepsilon_1, \gamma_2, \varepsilon_2, \lambda_1, \mu_1, \lambda_2, \mu_2$ into a matrix $\mathrm{B}_{m_1 \times m_2}$ (here we suppose $m_1 = m_2 = m$) and l_1, k_1, l_2, k_2, c, d into a matrix $\mathrm{C}_{m_1 \times m_2}$ (here we suppose $m_1 = m_2 = m$) that obtained in step 1 and step 2. The fourteen numbers' positions are determined by the user T. The positions of these numbers are T's privacy information that should not be known by CS.

Here we assume that

$b_{11} = \gamma_1, b_{25} = \lambda_1$, $b_{32} = \varepsilon_1$, $b_{41} = \lambda_2$, $c_{54} = c$, $b_{62} = \varepsilon_2$, $b_{73} = \mu_1$, $b_{85} = \mu_2$, $c_{93} = d$, $b_{46} = \gamma_2$, $c_{36} = l_1, c_{55} = k_1, c_{67} = l_2, c_{75} = k_2$, that

$$\mathrm{B} = \begin{bmatrix} \gamma_1 & * & * & * & * & * & \cdots & * \\ * & * & * & * & \lambda_1 & * & \cdots & * \\ * & \varepsilon_1 & * & * & * & * & \cdots & * \\ \lambda_2 & * & * & * & * & \gamma_2 & \cdots & * \\ * & * & * & * & * & * & \cdots & * \\ * & \varepsilon_2 & * & * & * & * & \cdots & * \\ * & * & \mu_1 & * & * & * & \cdots & * \\ * & * & * & * & \mu_2 & * & \cdots & * \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ * & * & * & * & * & * & \cdots & * \end{bmatrix}_{m \times m} \quad \mathrm{C} = \begin{bmatrix} * & * & * & * & * & * & * & \cdots & * \\ * & * & * & * & * & * & * & \cdots & * \\ * & * & * & * & * & l_1 & * & \cdots & * \\ * & * & * & * & * & * & * & \cdots & * \\ * & * & * & c & k_1 & * & * & \cdots & * \\ * & * & * & * & * & * & l_2 & \cdots & * \\ * & * & * & * & k_2 & * & * & \cdots & * \\ * & * & * & * & * & * & * & \cdots & * \\ * & * & d & * & * & * & * & \cdots & * \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ * & * & * & * & * & * & * & \cdots & * \end{bmatrix}_{m \times m}$$

Then, T sends B, C, v_1, v_2 and f to CS.

Step4. *Computation*

After receiving these information, CS computes $\mathrm{D} = (v_1{}^{b_{ij}})_{m \times m}$, $\mathrm{E} = (v_2{}^{b_{ij}})_{m \times m}$ and $\mathrm{F} = (f^{c_{ij}})_{m \times m}$. (We can get the three matrixes according to formulas described above)

Then, CS sends the calculation results D, E and F to T.

Step5. *Verification*

T computes $n_1 = d_{11} \cdot d_{32}$, $n_2 = e_{25} \cdot e_{73}$, $n_3 = d_{46} \cdot d_{62}, n_4 = e_{41} \cdot e_{85}, n_5 = f_{54}, n_6 = f_{93}$, $n_7 = f_{36} \cdot f_{55}$, $n_8 = f_{68} \cdot f_{75}$.

Then, T verifies the correctness of the calculation results by judging the equations $n_3 = {n_1}^r$, $n_4 = {n_2}^r$, $n_7 = {n_5}^r$, $n_8 = {n_6}^r$. If not, T outputs "error" and aborts the protocol. Otherwise, T computes ${u_1}^a {u_2}^b = n_1 \times n_5 \times n_2 \times n_6$.

4.2 Security Analysis

Our secure outsourcing scheme of Sexp is with high checkability and efficiency.

Theorem 2: Our scheme Sexp is secure and verifiable which support only single cloud server.

Proof: The outsourcing scheme Sexp satisfies correctness, soundness, privacy, verifiability and efficiency, which are proved as follows.

- *Correctness:* In our scheme, if the cloud server CS honestly implements the scheme, the outputs that CS produces will be correct and accepted by the user T.

  Since $v_1 = u_1/h$, $v_2 = u_2/h$, $a = \gamma_1 + \varepsilon_1$ and $b = \lambda_1 + \mu_1$, then $n_1 = {v_1}^{\gamma_1} {v_1}^{\varepsilon_1} = {v_1}^a$ and $n_2 = {v_2}^{\lambda_1} {v_2}^{\mu_1} = {v_2}^b$.

  According to $ra = \gamma_2 + \varepsilon_2$, we can know $n_3 = {v_1}^{\gamma_2} {v_1}^{\varepsilon_2} = {v_1}^{ra}$, therefore $n_3 = {n_1}^r$. Similarly, on the base of $rb = \lambda_2 + \mu_2$, we can know $n_4 = {v_2}^{\lambda_2} {v_2}^{\mu_2} = {v_2}^{rb}$, therefore $n_4 = {n_2}^r$

  According to $rc = l_1 + k_1$, we can know $n_7 = f^{l_1} f^{k_1} = f^{rc}$, therefore $n_7 = {n_5}^r$. Similarly, on the base of $rd = l_2 + k_2$, we can know $n_8 = f^{l_2} f^{k_2} = f^{rd}$, therefore $n_8 = {n_6}^r$.

  Since ${u_1}^a {u_2}^{\mathrm{b}} = (v_1 \cdot g^{\alpha})^a (v_2 \cdot g^{\alpha})^{\mathrm{b}} = {v_1}^{\gamma_1} {v_1}^{\varepsilon_1} g^{\alpha a} \cdot {v_2}^{\lambda_1} {v_2}^{\mu_1} g^{\alpha b}$, then ${u_1}^a {u_2}^b = n_1 \times n_5 \times n_2 \times n_6$ establish as well.

  It can be seen that verification equation will establish if the cloud server is honest.
- *Soundness:* The probability of these outputs returned by a dishonest cloud server can pass the last step (verification) is almost negligible. The analysis process is similar to the Theorem 1 in section 3.2. There are three types of dishonest behaviors. In the first type, the CS may change the values in matrix B and matrix C. If CS wants to pass verification equations in step 5, then CS needs to guess the right positions of $(\gamma_1, \varepsilon_1, \gamma_2, \varepsilon_2, \lambda_1, \mu_1, \lambda_2, \mu_2)$ in matrix B, $(l_1, k_1, l_2, k_2, c, d)$ in matrix C and obtain the value of r at the same time. In the second type, CS may change the values in matrixes D, E, F and obtain the value of r at the same time. In other words, CS needs to guess the right positions of $(d_{11}, d_{32}, d_{46}, d_{62})$ in matrix D, $(e_{25}, e_{73}, e_{41}, e_{85})$ in matrix E, $(f_{54}, f_{93}, f_{36}, f_{55}, f_{68}, f_{75})$ in matrix F and obtain the right value of r. The probabilities that the cloud server dishonestly act can pass the verifications are $\frac{(m^2-8)!(m^2-6)!}{10(m^2!)^2}$, $(\frac{(m^2-4)!}{m^2!})^2 \cdot \frac{(m^2-6)!}{10m^2!}$ and $(\frac{(m^2-4)!}{m^2!})^2 \cdot \frac{(m^2-6)!}{10m^2!}$, respectively.
- *Privacy:* The proof of inputs and outputs is similar to the Theorem 1 in section 3.2. The probability of the CS or attacker try to get some private data during executing the outsourcing scheme is at most $\frac{1}{m^{12}}$.
- *Verifiability:* We can analyze the verifiability of Sexp by using the same method as described in the Theorem 1 in section 3.2. Let X be the event of "CS can pass the

verification equations and affect the correctness of $u_1{}^a u_2{}^b$". There are three types of dishonest behaviors. When it comes to the first type of dishonest behaviors described in soundness, the probability of X is Pr[X]= $\frac{(m^2-8)!(m^2-6)!}{10(m^2!)^2}$ for the PPT attacker. When it comes to the second type of dishonest behaviors described in soundness, the probability of X is Pr[X]= $(\frac{(m^2-4)!}{m^2!})^2 \cdot \frac{(m^2-6)!}{10m^2!}$ for the PPT attacker. When it comes to the third type of dishonest behaviors described in soundness, the probability of X is Pr[X]=$(\frac{(m^2-4)!}{m^2!})^2 \cdot \frac{(m^2-6)!}{10m^2!}$ for the PPT attacker. The probability of three dishonest behaviors that CS may shows are $\frac{(m^2-8)!(m^2-6)!}{10(m^2!)^2}$, $(\frac{(m^2-4)!}{m^2!})^2 \cdot \frac{(m^2-6)!}{10m^2!}$ and $(\frac{(m^2-4)!}{m^2!})^2 \cdot \frac{(m^2-6)!}{10m^2!}$, respectively.

Let Y be the event of "attacker can forge results and successfully cheat user, and the correctness of $u_1{}^a u_2{}^b$ can be affected". The $Pr(Y) \leq \frac{(m^2-8)!(m^2-6)!}{10(m^2!)^2} + \left(\frac{(m^2-4)!}{m^2!}\right)^2 \cdot \frac{(m^2-6)!}{5m^2!}$. The user can detect the incorrectness with probability at least $1 - \frac{(m^2-8)!(m^2-6)!}{10(m^2!)^2} - (\frac{(m^2-4)!}{m^2!})^2 \cdot \frac{(m^2-6)!}{5m^2!}$.

- *Efficiency:* The proof of efficiency is similar to the Theorem 1 in section 3.2. Our scheme Exp requires 17 MM, 4 MInv and 2 invocation of *Rand*, and our schemes only need to rent one untrusted cloud server. In addition, our schemes don't need any cryptographic assumptions.

4.3 Comparison

Compared with [15] and [13], the weakness of our scheme Sexp is increased MM, but our scheme only requires 4 MInv (we omit other operations such as modular additions) and 2 invocations of *Rand*. The comparison is listed in **Table 2**.

Table 2. Comparison of other schemes vs our Sexp

Scheme	[15]	[13].Sexp	Our Sexp
MM	9	10	17
MInv	5	4	4
Invoke(Rand)	6	5	2
Matrix blinding	No	No	Yes
support single CS	No	No	Yes
Checkability	1/2	1/2	$\geq 1 - 1/O(m^{28})$

Our scheme is secure and efficient with a higher checkability by using only one untrusted cloud server. The two schemes proposed in [15] and [13] require two non-collusion cloud servers. In a single-cloud server model, the checkability is greater than 1-$1/O(m^{28})$ and it is approximately equal one when the matrix size *m* is a big number in our scheme Sexp. We reduce the amount of cloud server in our scheme, and the less amount of cloud

server the lower probability of collusion. In addition, we adopt the method of matrix blinding in our scheme Sexp to achieve privacy-preserving.

5. Conclusion

In this paper, we propose two outsource-secure schemes of modular exponentiation by using single untrusted cloud server. Compared with related schemes, our schemes are more efficient in efficiency and checkability. The model of our schemes just deploys in one untrusted cloud server and reduces the users' local computation cost during the implementation process of computation. Moreover, our schemes are secure without any cryptographic assumptions, and we also analyze and prove the efficiency and security of our schemes. The analysis shows that our schemes are more efficient and practical. Secure outsourcing of modular exponentiations is widely used the field of cryptography and we will do more research on this field in future.

References

[1] M. Li, S. Yu, K. Ren and W. Lou, "Securing personal health records in cloud computing: Patient-centric and fine-grained data access control in multi-owner settings," in *Proc. of International Conference on Security and Privacy in Communication Systems. Springer Berlin Heidelberg*, pp: 89-106, 2010. Article (CrossRef Link)

[2] O. Ali, J. Soar and J. Yong, "Impact of cloud computing technology on e-government," in *Proc. of International Conference on Information and Software Technologies. Springer International Publishing*, pp: 272-290, 2014. Article (CrossRef Link)

[3] N. D. Han, Y. Chung, and M. Jo, "Green data centers for cloud-assisted mobile ad hoc networks in 5G," in *Proc. of IEEE Network*, pp: 70-76, 2015. Article (CrossRef Link)

[4] H. B. Patel, D. R. Patel, B. Borisaniya and A. Patel, "Data storage security model for cloud computing," in *Proc. of International Conference on Advances in Communication, Network, and Computing. Springer Berlin Heidelberg*, pp: 37-45, 2012. Article (CrossRef Link)

[5] J. Shen, H. Tan, J. Wang, and S. Lee, "A Novel Routing Protocol Providing Good Transmission Reliability in Underwater Sensor Networks," *Journal of Internet Technology*, vol. 16, no. 1, pp. 171-178, 2015. Article (CrossRef Link)

[6] Z. Fu, X. Sun, Q. Liu, L. Zhou and J. Shu, "Achieving Efficient Cloud Search Services: Multi-keyword Ranked Search over Encrypted Cloud Data Supporting Parallel Computing," *IEICE Transactions on Communications*, vol. E98-B, no. 1, pp.190-200, 2015. Article (CrossRef Link)

[7] R. Chow, P. Golle, M. Jakobsson, E. Shi, J. Staddon, R. Masuoka and J. Molina, "Controlling data in the cloud: outsourcing computation without outsourcing control," in *Proc. of the 2009 ACM workshop on Cloud computing security. ACM*, pp. 85-90, November 13, 2009. Article (CrossRef Link)

[8] R. Gennaro, C. Gentry, and B. Parno, "Non-interactive Verifiable Computing: Outsourcing Computation to Untrusted Workers," in *Proc. of Advances in Cryptology–CRYPTO 2010. Springer Berlin Heidelberg*, pp. 465-482, May 27, 2010. Article (CrossRef Link)

[9] S. Iftene. "Modular exponentiation," *International Journal of Computing 2.3*, pp. 49-55 2003. Article (CrossRef Link)

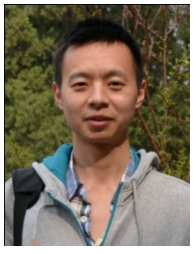
[10] P. K. Mohapatra, "Public key cryptography," *Crossroads 7.1*, pp.14-22, September 2000. Article (CrossRef Link)
[11] A. Shamir and Y. Tauman, "Improved online/offline signature schemes," in *Proc. of Advances in Cryptology—CRYPTO 2001. Springer Berlin Heidelberg*, pp. 355-367, 2001. Article (CrossRef Link)
[12] B. Schneier, "One-Way Hash Functions," *Applied Cryptography, Second Edition, 20th Anniversary Edition*, pp. 429-459, 1996. Article (CrossRef Link)
[13] X. Chen, J. Li, J. Ma, Q. Tang and W. Lou, "New algorithms for secure outsourcing of modular exponentiations," in *Parallel and Distributed Systems, IEEE Transactions on 25.9*, pp. 2386-2396, September 2014. Article (CrossRef Link)
[14] G. O. Karame and S. Čapkun, "Low-cost client puzzles based on modular exponentiation," in *Proc. of Computer Security–ESORICS 2010. Springer Berlin Heidelberg*, pp. 679-697, 2010. Article (CrossRef Link)
[15] S. Hohenberger and A. Lysyanskaya, "How to securely outsource cryptographic computations," *Theory of Cryptography. Springer Berlin Heidelberg*, pp. 264-282, 2005. Article (CrossRef Link)
[16] D. Fiore and R. Gennaro, "Publicly verifiable delegation of large polynomials and matrix computations, with applications," in *Proc. of the 2012 ACM conference on Computer and communications security*, pp. 501-512, 2012. Article (CrossRef Link)
[17] B. Parno, M. Raykova and V. Vaikuntanathan, "How to delegate and verify in public: Verifiable computation from attribute-based encryption," *Theory of Cryptography. Springer Berlin Heidelberg*, pp. 422-439, 2012. Article (CrossRef Link)
[18] M. J. Atallah, K. Pantazopoulos, J. R. Rice, and E. E. Spafford, "Secure outsourcing of scientific computations," *Computers Trends in Software Engineering*, pp. 215-272, 2002. Article (CrossRef Link)
[19] S. Benabbas, R. Gennaro and Y. Vahlis, "Verifiable delegation of computation over large datasets," in *Proc. of Cryptology–CRYPTO 2011. Springer Berlin Heidelberg*, pp. 111-131, July 11, 2011. Article (CrossRef Link)
[20] V. Boyko, M. Peinado, and R. Venkatesan, "Speeding up discrete log and factoring based schemes via precomputations," *Advances in Cryptology—EUROCRYPT'98. Springer Berlin Heidelberg*, pp. 221-235, 1998. Article (CrossRef Link)
[21] M. Jakobsson and S. Wetzel, "Secure server-aided signature generation," *Public Key Cryptography. Springer Berlin Heidelberg*, pp. 383-401, 2001. Article (CrossRef Link)
[22] M. V. Dijk, D. Clarke, B. Gassend, G. E. Suh and S. Devadas, "Speeding up exponentiation using an untrusted computational resource," *Designs, Codes and Cryptography 39.2*, pp. 253-273, May 2006. Article (CrossRef Link)
[23] X. Ma, J. Li, and F. Zhang, "Outsourcing computation of modular exponentiations in cloud computing," *Cluster computing 16.4*, pp. 787-796, December 2013. Article (CrossRef Link)
[24] Y. Wang, Q. Wu, D. S. Wong, B. Qin, S. S. M. Chow, Z. Liu and X. Tan, "Securely outsourcing exponentiations with single untrusted program for cloud storage," in *Proc. of Computer Security-ESORICS 2014. Springer International Publishing*, pp. 326-343, 2014. Article (CrossRef Link)
[25] C. Xiang and C. Tang, "Efficient outsourcing schemes of modular exponentiations with checkability for untrusted cloud server," *Journal of Ambient Intelligence and Humanized Computing 6.1*, pp. 131-139, February 2015. Article (CrossRef Link)

[26] S. G. Choi, J. Katz, R. Kumaresan, and C. Cid, "Multi-client non-interactive verifiable computation," in *Proc. of Theory of Cryptography Lecture Notes in Computer Science (TCC 2013)*, pp. 499–518, 2013. Article (CrossRef Link)
[27] C. Papamanthou, E. Shi, and R. Tamassia, "Signatures of correct computation," in *Proc. of Theory of Cryptography Lecture Notes in Computer Science (TCC 2013)*, pp. 222-242, 2013. Article (CrossRef Link)
[28] X. Hu, D. Pei, C. Tang and D. Wong, "Verifiable and secure outsourcing of matrix calculation and its application," *SCIENCE CHINA Information Sciences*, pp. 842-852, 2013. Article (CrossRef Link)
[29] Y. Peng, Z. Fan, B. Choi, J. Xu, and S. S. Bhowmick, "Authenticated subgraph similarity search in outsourced graph databases," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 7, pp. 1838-1860, Jan. 2015. Article (CrossRef Link)
[30] P. Q. Nguyen, I. E. Shparlinski, and J. Stern, "Distribution of modular sums and the security of the server aided exponentiation," *Cryptography and Computational Number Theory. Birkhäuser Basel*, pp. 331-342, 2001. Article (CrossRef Link)
[31] X. Lei, X. Liao, X. Ma, and L. Feng, "Securely and efficiently perform large matrix rank decomposition computation via cloud computing," *Cluster Computing*, vol. 18, no. 2, pp. 989-997, 2015. Article (CrossRef Link)
[32] J. Ye, X. Chen and J. Ma, "An Improved Algorithm for Secure Outsourcing of Modular Exponentiations," in *Proc. of Advanced Information Networking and Applications Workshops (WAINA), 2015 IEEE 29th International Conference on. IEEE*, pp. 73-76, March 24-27, 2015. Article (CrossRef Link)
[33] X. Chen, W. Susilo, J. Li, D. S. Wong, J. Ma, S. Tang and Q. Tang, "Efficient algorithms for secure outsourcing of bilinear pairings," *Theoretical Computer Science, Volume 562 Issue C*, pp. 112-121, January 2015. Article (CrossRef Link)

Ling Zhao received the B.E. degree from Hubei University of Technology, Wuhan, China, in 2014. She is pursuing a master's degree at the School of Computer Sciences, Hubei University of Technology. Her current research interests include the applied cryptography and outsourcing computation techniques.

Mingwu Zhang is a professor at School of Computer Sciences, Hubei University of Technology. From Aug 2010 to Aug 2012, he has been a JSPS postdoctoral fellow of Japan Society of Promotion Sciences at Institute of Mathematics for Industry in Kyushu University. From Jun 2015 to Jun 2016, he was a senior visiting professor at University of Wollongong, Australia. His research interests include cryptography technology for networks, secure computations, and privacy preservations etc. Dr. Zhang is the director of Institute of Big-data Security and Privacy Preservation of HBUT. （E-mail: mzhang@mail.hbut.edu.cn）

Hua Shen is an associate professor at School of Computer Sciences, Hubei University of Technology. Her research focuses on the security protocol design and applications in open networks.



Yudi Zhang is a master student at School of Computer Sciences, Hubei University of Technology. His research interest focuses on the applied cryptography and program.

Jian Shen received the B.E. degree from Nanjing University of Information Science and Technology, Nanjing, China, in 2007 and the M.E. and Ph.D. degrees in Computer Science from Chosun University, Gwangju, Korea, in 2009 and 2012, respectively. Since late 2012, he has been a faculty member in the School of Computer and Software at Nanjing University of Information Science and Technology, Nanjing, China. His research interests include computer networking, security systems, mobile computing and networking, ad hoc networks and systems.