International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 05 Issue: 02 | Feb-2018 www.irjet.net p-ISSN: 2395-0072

A Scheme of Verifiable Computing with Context Awareness and Privacy Preservation

Gomathipriya R[1] , Dr.M.Nithya[2], Dr. K.Sasikala[3]

[1,2,3]*VMKV Engineering College, Salem*

Abstract - *Internet of Things (IoT) has developed to produce different intelligent applications based on the data collected by different "things".It is an effective scheme to achieve both cloud data privacy and verifiability of cloud data processing. The proposed scheme can play as a generic framework for verifiable computing in cloud with context awareness and privacy preservation in IoT cloud computing. It supports various data processes by applying full homomorphic technologies and deploying an auditing protocol to verify the correctness of encrypted data processing. To apply full homomorphic encryption (FHE) technologies to process data in an encrypted form at CSP in order to protect the privacy of data providers and data owners, four optional auditing protocols in order to satisfy different security and performance requirements. Three of them can ensure system security in case that RPs could collude with CSP.. In a further deployment an auditing protocol to verify the correctness of encrypted data processing by applying a Trusted Auditing Proxy (TAP). Concretely, a Data Provider (DP) encrypts its collected data with the homomorphic key offered by the TAP and signs it with a data context identifier (ID).It analyzed scheme security and evaluate the performance of the proposed protocols through rigorous analysis and comparison in order to show their pros and cons, as well as applicability.*

1.INTRODUCTION

Internet of Things (IoT) is going to create a world where physical objects are seamlessly integrated into information networks in order to provide advanced and intelligent services for human-beings. The interconnected "things" such as sensors and mobile devices sense, moni-tor and collect all kinds of data about human social life. These data can be further aggregated, fused, processed, analyzed and mined in order to extract useful infor-mation to enable intelligent and ubiquitous services. The IoT is evolving as an important part of next generation networking paradigm and service infrastructure. Various applications and services of IoT have been emerging into markets in broad areas, e.g., surveillance, health care, se-curity, transport, food safety, and distant object monitor and control.

On the other hand, cloud computing offers a new way of service provision by re-arranging various resources (e.g., storage, computing and applications) and providing them to users based on their demands. The cloud compu-ting provides a large resource pool by linking network resources together. It has desirable properties, such as scalability, elasticity, fault-tolerance, and pay- per-use. Thus, it has become a promising service platform by rear-ranging the way that information technology services are provided and consumed.

1.1 Motivations

In practice, cloud computing can cooperate with IoT by providing computing services in order to release the load of big data processing at user devices and some service providers. One practical scenario is that the data moni-tored or sensed from the 'things' can be aggregated and sent to the cloud to process in order to provide data com-putation and processing results to requesting parties (e.g., IoT service providers). However, Cloud Service Provider (CSP) is a party that cannot be fully trusted by IoT data providers and data requesting parties. The CSP could disclose the privacy of data providers or owners by mali-ciously miming data. It may provide wrong data pro-cessing results to the requesting parties to intentionally destroy IoT service quality. In this case, how to ensure the facticity and genuine of data sources and the correctness of IoT data processing, computation, and mining becomes a practically crucial issue that greatly impacts the contin-uous success of IoT and cloud computing, as well as the future Internet.

Since the CSP cannot be fully trusted and the privacy of monitored objects should be preserved, data collected by 'things' are generally provided to the CSP in an en-crypted form for further processing. In practice, different types of data could be collected and processed at the cloud. For example, a personal mobile phone can collect its user information about location, calling, radio connec-tivity quality, inbound/outbound data traffic, personal heart beat rate, blood pressure, breathing vol-ume/frequency and so on. The collected data can be fur-ther processed and used by different IoT services to offer a diversity of smart services. The algorithms used for computing or processing different types of data could be different at the cloud. For the purpose of preserving data privacy, the cloud processes the encrypted data and pro-vides processing results to requesting parties mostly in an encrypted form. How to ensure, audit and verify the fac-ticity and correctness .

1.2 Main Contributions

In this paper, we propose a scheme of verifiable compu-ting with context awareness and privacy preservation in IoT cloud computing. We first apply full homomorphic encryption (FHE) technologies to process data in an en-crypted form at

CSP in order to protect the privacy of data providers and data owners. We further deploy an auditing protocol to verify the correctness of encrypted data processing by applying a Trusted Auditing Proxy (TAP) . Concretely, a Data Provider (DP) encrypts its col-lected data with the homomorphic key offered by the TAP and signs it with a data context identifier (ID) . Then it transmits the encrypted data, context ID and the signa-ture to the CSP as an input of multi- party computation. The CSP computes the encrypted data from all DPs based on the context IDs by selecting a corresponding algorithm and signs the computation result (which is in an encrypt-ed form). For accessing the computation result, a request-ing party (RP) requests the result from the CSP with re-gard to a context, the CSP passes the request to the TAP to check its eligibility in order to allow the RP to access the data processing result.

When the RP wants to verify the correctness of data processing and computation of CSP, it reports the pro-cessing result signed by the CSP and its hash code to the TAP. The TAP queries the CSP to get the encrypted data with regard to the RP request in order to audit the data processing at the CSP. For supporting the CSP to check the facticity and genuine of data sources, the scheme re-quests that DP signs its provided data regarding a context in order to allow the TAP later on to figure out malicious DPs during auditing by finding malicious data input through analysis and mining. We design four auditing protocols to satisfy different security requirements. Their performance is evaluated and compared in order to show the pros and cons of each protocol and its feasibility in different scenarios.

Specifically, the contribution of this paper can be summarized as below:

We motivate context-aware verifiable computing for cloud and propose an effective scheme to achieve both cloud data privacy and verifiability of cloud data pro-cessing.

To the best of our knowledge, our scheme is one of the first to realize verifiable cloud computing with con-text-awareness. It supports various data processes by ap-plying full homomorphic technologies and deploying an auditing protocol to verify the correctness of encrypted data processing. The proposed scheme can play as a ge-neric framework for verifiable computing in cloud. But one important issue missed in the above studies is verifiable computing that can audit the correctness of en-crypted data processing

2. RELATED WORK

2.1. Privacy-Preserving Data Mining (PPDM)

Privacy-Preserving Data Mining (PPDM) aims to support data mining related computations, processes or opera-tions with privacy preservation [32]. PPDM is a 'must-solve' problem in IoT for securely and intelligently sup-porting various IoT services in a pervasive and personal-ized way. From the practical point of view, PPDM is still a challenge, considering trustworthiness, computation complexity and communication cost [32].

Mishra and Chandwani proposed an architecture to enable Secure Multi-party Computation (SMC) by hiding the identities of the parties that take part in the process of Business Process Outsourcing [4]. A class of functions was pro posed to provide additional abilities to a party to split its huge data before providing it for computation, making it almost intractable for other parties to know the actual source of the data in order to achieve secure and privacy-preserving data mining. Liu et al. proposed a secure multi-party multi-data ranking protocol and a se-cure multi -party addition protocol to complete private-preserving sequential pattern mining [5].

A number of operations on securely input data are supported in PPDM. Zhu et al. proposed schemes for se-curely extracting knowledge from two or more parties' private data [6]. They presented three different approach-es to privacy-preserving Add to Multiply Protocol, as well as further extended it to privacy- preserving Adding to Scalar Product Protocol. Wang and Luo studied a pri-vate preserving shared dot product protocol that is a main building block of various data mining algorithms with privacy concerns, and provides fundamental securi-ty guarantee for many PPDM algorithms [7]. They con-structed a privacy-preserving two-party shared dot prod-uct protocol based on some basic cryptographic tech-niques, which is provably secure in a semi-honest model. Shen, Han and Shan proposed a Horizontal Distribution of the Privacy Protection DK- Means (HDPPDK-Means) algorithm based on Horizontal partitioned database and DK-means idea to realize distributed clustering and ap-plied a secure multi-party computation protocol to achieve privacy preservation [8].

Many studies focused on supporting specific data min-ing techniques with privacy preservation. Statistical hy-pothesis test is an important data analysis technique. Liu and Zhang investigated nonparametric Sign Test (NST) theory in such a context that two parties, each with a pri-vate dataset, would like to conduct a sign test on their joint dataset, but neither of them is willing to disclose its private dataset to any other third parties [9]. Their pro-posed protocol does not make use of any third party nor cryptographic primitives.

To exchange the data while keeping it private by using homomorphic encryption techniques [10]. Kant-arcioglu and Clifton [11] proposed two protocols to im-plement privacy-preserving mining of association rules over horizontally partitioned data. Zhang and Zhao fur-ther revised its security proof [12]. Privacy-preserving association rule mining was surveyed by Wang [13] with regard to basic concepts, general principles and methods.

In the case that agencies want to conduct a linear re-gression analysis with complete records without disclos-ing values of their own attributes, Ashish et al. described an algorithm that

enables agencies to compute the exact regression coefficients of the global regression equation and also perform some basic goodness-of- fit diagnostics while protecting the confidentiality of their data [14]. This work can be applied for distributed computation for re-gression analyses in data mining.

Finding the nearest k objects to a query object (k-NN queries) is a fundamental operation for many data mining algorithms to enable clustering, classification and outlier detection tasks. Efficient solutions for k- NN queries for vertically partitioned data were proposed by Amirbekyan and Estivill-Castro [18]. These solutions include L ∞ (or Chessboard) metric as well as detailed privacy-preserving computation of all other Minkowski metrics, privacy pre-serving algorithms for combinations of local metrics into global metric that handles large dimensionality and di-versity of attributes common in vertically partitioned da-ta, a privacy-preserving SASH (a very successful data structure for associative queries in high dimensions) for managing very large data sets. Liu et al. presented priva-cy preserving algorithms for DBSCAN clustering for the horizontally, vertically and arbitrarily partitioned data distributed between two parties [15]. DBSCAN [16] is also a popular density-based clustering algorithm for discov-ering clusters in large spatial databases with noise.

Gradient descent is a widely used method for solving optimization and learning problems. Wan et al. presented a generic formulation of secure gradient descent methods with privacy preservation [17]. It underlies many com-monly used techniques in data mining and machine learning, such as neural networks, Bayesian networks, genetic algorithms, and simulated annealing.

Current solutions of PPDM are still not practical. The existing methods are impractical, ineffective, inefficient or inflexible with regard to generality, trustworthiness, computation complexity and communication cost. Sel-dom, a PPDM protocol can satisfy all essential require-ments for practical usage.

2.1. SMC Applications

Secure Multi-party Computation (SMC) deals with the problem of secure computation among participants who are not trusted with each other, particularly with the preference of privacy preserving computatio nal geome-try. SMC refers to the parties, who participate in the com-putation with their own secret inputs, wish to coopera-tively compute a function. When the computation is over, each party can receive its own correct output with cor-rectness insurance, and know its own output only with privacy preservation. It is an important research topic in IoT. The problems of SMC are specifically different in different scenarios. Based on the problems solved, SMC mechanisms can be classified into four categories: priva-cy-preserving database query, privacy-preserving scien-tific computation, privacy-preserving intrusion detection and privacy-preserving data mining. A detailed survey on SMC technologies is provided in [32].

SMC together with homomorphic encryption is widely applied into many areas, such as distributed electronic contract management [19], smart meter based load man-agement [20], healthcare frauds and abuses [21], policy-agile encrypted networking for defense, law enforcement, intelligence community, and commer cial networks [22], privacy preserving path inclusion [29], privacy preserving string matching [23], privacy- enhanced recommender system in a social trust network [24], user profile match-ing in social networking [25], credit check applications [26], private collaborative forecasting and benchmarking [27], privacy-preserving genomic computations [28], pro-tection against insider threats (e.g., business partners) [30], privacy preserving electronic voting [31], and so on. But one important issue missed in the above studies is verifiable computing that can audit the correctness of en-crypted data processing.

2.3. Cloud Computing Auditing

Current researches about verifiable cloud computing fo-cus on auditing cloud data storage and data integrity with regard to data management, such as insertion, dele-tion, and addition [1- 3, 38, 41]. It has not yet investigated data calculation and computation seriously. Yang et al. proposed an approach to fast detect data errors in big sensor data sets based on a scale-free network topology and most of detection operations can be conducted in limited temporal or spatial data blocks instead of a whole big data set [39]. Some researchers applied a MapReduce framework to anonymize large-scale data sets in cloud [40].

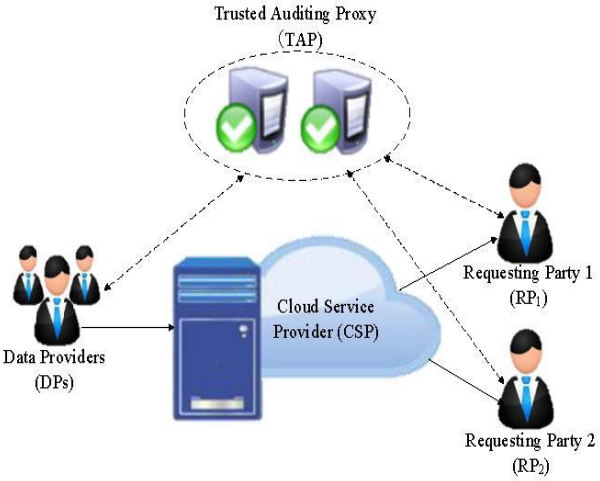
We notify that all above presented work did not con-sider how to solve the problems of verifiable computing with context awareness as described in Section 1 in cloud.

3. PROBLEM STATEMENT

3.1. System and Threat Models

We consider an IoT cloud computing system that in-volves four different kinds of entities, as illustrated in Fig.1: the Data Providers (DPs) that interact with the physical world, detect/monitor/sense information of objects in different contexts and provide collected data to a CSP for processing; the CSP that has functions and ca-pabilities that the DPs do not have and can process the data provided by the DPs. Thus, the DPs encrypt the data provided to the CSP; the TAP is responsible for issuing essential keys to DPs for homomorphic encryption, checking the eligibility of data access at the CSP, issuing access keys to eligible Requesting Parties (RPs), and auditing the facticity and genuine of data sources and the correctness of CSP data processing and computation; the RP that requests CSP's data processing results in different contexts for offering intelligent and

ubiquitous services to IoT end users. We assume that all system entities communicate with each other via a secure channel (e.g., OpenSSL).



In the system, TAP and CSP do not collude. RP can on-ly access the final data processing results from CSP. CSP has no rights to offer the data collected from DPs to RPs. RP can request TAP and delegate TAP to audit the fac-ticity and genuine of data collection and processing.

1) Security and safety: it is hard or impossible for CSP to get the raw data during data processing; the data processed at the cloud can only be accessed by eligible system entities;

2) Gener-ality: the proposed scheme supports various data pro-cesses in different contexts in the cloud. 3) Reasonable overhead: the scheme fulfills data processing and audit-ing with reasonable computation and communication costs.

Introducing TAP enhances the facticity of DP. TAP can mine the collected data of CSP and analyze if the data source has some abnormal behav-iors by comparing such an assumption: CSP does not collude with its users, e.g., colluding with some RPs to gain raw data and allow other RPs to access data stored at CSP. But some inci-dental collusion between CSP and RPs could happen, which requests higher level security assurance in the sys-tem design.

3.2. Design Goals

To achieve trustworthy data processing and avoid poten-tial risks in cloud services, our design should achieve the following security and performance goals: 1) Security and safety: it is hard or impossible for CSP to get the raw data during data processing; the data processed at the cloud can only be accessed by eligible system entities; 2) Gener-ality: the proposed scheme supports various data pro-cesses in different contexts in the cloud. 3) Reasonable overhead: the scheme fulfills data processing and audit-ing with reasonable computation and communication costs.

4. PROPOSED SCHEME

4.1. Preliminaries, Notations and Definitions

4.1.1. Full Homomorphic Encryption

Full Homomorphic Encryption (FHE) mainly consists of four algorithms: Key-Generate (KG), Encrypt (E), Decrypt (D) and Evaluate ((, ,) , where pk is a public key used to calculate ciphertexts. In these four algorithms, the Evaluate algorithm does computation on a set of cipher-text = ⟨ !, … , !⟩ (! = (, !)) , which is the input of circuit Cir. Circuit Cir represents a function or an algo-rithm. Based on these four algorithms, we define FHE as below.

For any key pair (,) generated by KG algorithm, any circuit Cir, any plaintext = ⟨ !, … , ! ⟩, and any ciphertext = ⟨ !, … , ! ⟩ ! = (, !) :

( )

if ! = (, ,) (1)

then (, ! =) (!, … , !). (2)

This means that we can do operations on the ciphertext in order to get the encrypted version of the result of plaintext operations. In scheme implementation and evaluation, we use Brakerski Gentry Vaikuntanathan (BGV) FHE [33].

4.1.2. Limitaions of Partial Homomorphic Encryption

Partial Homomorphic Encryption (PHE) is a cryptograph-ic algorithm that can achieve the same goal of processing ciphertexts as that of the FHE. It preforms much better than FHE in terms of computation and storage overhead. However, it can only support some specific operations, e.g., addition and subtraction. This means it is only appli-cable in some specific scenarios. The goal of our work is to design a generic scheme of verifiable cloud computing that can adapt to various operations,

Protocol 1

1)Data provision

This step is the same as the data provision described in Fig. 3.

2)Privacy preserving data computing

This step is the same as the privacy preserving data computation as described above in Fig 3.

3)RP data request and authorization

! requests CSP for the result of data processing and computation in ! by sending a requesting message that contains ! = { !"! , !} and (!"! , !) . Once re-ceiving the

request, the CSP passes the request to TAP for checking its access right. If the check based on the current access policy is positive, the TAP requests the data pro-cessing result (!, !) from CSP. After receiving (!, !) with !"# = (!"#, { (!, , !)}) , the TAP first decrypts (!, !) to get !, and then re-encrypts ! with the !'s public key !"! to get ′(!"! , !) based on a public key encryption scheme (e.g., RSA) and issues ′(!"! , !) to RP with its signa-ture !"# = (!"#, { ′(!"! , !)}).

4)Data access

After receiving ′(!"! , !), ! can decrypt it with its own secret key !"! to get of !.

5)Data auditing

RP can request TAP to audit the correctness of data processing and computation by providing ! , the hash code of !, $h($!$)$, and !"#. Note that the auditing request should be signed by RP to ensure non-repudiation. Thereby, the auditing request ! contains ! ={ ! , $h($!$)$, !"# , (!"! , { !, $h($!$)$, The de-tailed protocols are described below.

Protocol 2

1)Data provision

This step is the same as the data provision described in Fig. 3.

2)Privacy preserving data computing

CSP processes data, it selects algorithm ! based on ! to process the collected encrypted data (!, !,!) in context ! and gains the encrypted form of data pro-cessing result (!, !) , that is: (!, !) =
!{ (!, !,!)}, (= 1, … ,). The CSP then select a ran-dom and compute (!, ! ∗).

3)RP data request and authorization

! requests CSP for the result of data processing and computation in ! by sending a requesting message that contains ! = { !"! , !} and (!"! , !) . Once re-ceiving the request, the CSP passes the request to TAP for checking its access right. If the check ba sed on the current access policy is positive, the TAP then requests the data processing result from CSP. After receiving the package (!, ! ∗), !"# , the TAP first decrypt (!, ! ∗) to get ! ∗ , and then re-encrypt ! ∗ with the ! 's public key !"! to get (!"! , ! ∗) based on a homomorphic encryption scheme and issues (!"! , ! ∗) to ! through CSP. Note that !"! used herein is a homomorphic public key of ! and !"! is a corresponding homomorphic secret key of !.

4)Data access

CSP receives (!"! , ! ∗), which means TAP is-sues ! the right to access !. CSP uses a homomor-phic multiplication algorithm to erase from
(!"! , ! ∗):
(!"! , ! ∗) ∗ (!"! , 1/) = (!"! , !). And then it issues (!"! , !) and !"# =
(!"#, { (!"! , !), !}) to ! . The ! can de-crypt it with its own secret key !"! to get the plaintext
of !.

5)Data auditing

RP may not trust the processing result of CSP. In this case, it requests TAP to audit the correctness of data pro-cessing and computation by providing !, the hash code of !, $h($!$)$, the signature of CSP data provision, i.e., !"# = (!!", { (!"! , !), !}). Note that the auditing request should be signed by RP to ensure non-repudiation. Thereby, the auditing request ! contains ! ={ ! , $h($!$)$, !"# , (!"! , { !, $h($!$)$, !"#})}. In this case, TAP handles it by querying CSP to get ! and all (!, !,!) used for generating (!, !). TAP decrypts (!, !,!) to get all !,! and input them into ! to get plain ! , that is ! =!({ !,!}) (= 1, … ,). TAP further compares the hash code of ! output from F! and the one provided by RP in order to judge if the data computation and processing at CSP is correct.

!"#})}. In this case, TAP handles it by querying CSP to get ! and all (!, !,!) used for generating (!, !). TAP decrypts (!, !,!) to get all !,! and input them into ! to get plain ! , that is ! =!({ !,!})(= 1, … ,) . TAP further compares the hash code of ! output from ! and the one provided by RP in order to judge if the data computation and processing at CSP is correct.

Comparing with the original Protocol 1, Protocol 2 per-forms re-encryption at TAP to convert the encrypted version of ! from (!, !) to !(!"! , !), which can be decrypted by corresponding ! directly. ! will never have the chance to get access to the homomorphic secret key ! . Therefore, this protocol can reduce the risk caused by the CSP-RP collusion attack.

4.2. Further Optimization

In this section, we show a potential problem of Protocol 2 and then propose two optimized protocols to ove rcome it. As specified in our design goals, data should be pro-cessed and computed in a confidential measure at CSP and only eligible parties can access the processing results. TAP is responsible for issuing access rights and auditing. Normally, we do not expect TAP to know ! during data request and access. In Protocol 2, TAP gets to know the data processing result ! before auditing during the procedure of re-encryption. This could cause some prob-lem since TAP may not be an eligible party to access ! even though it is fully trusted for auditing when RP dele-gate it for this purpose.In order to avoid this

problem, we further propose two optimized protocols. In these two protocols, we let CSP fuzzifies ! by selecting a random denoted as a mask, computing (!, ! ∗), and then using differ-ent methods to remove and finally get !. The de-tailed protocols are described below.

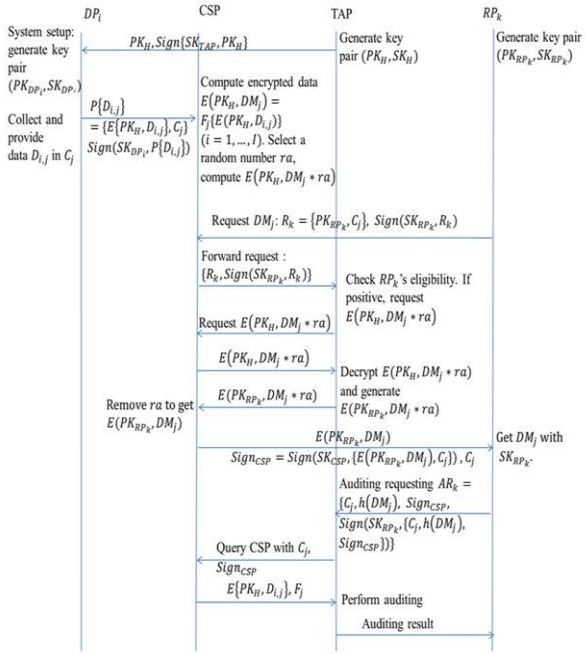


Fig.2: Protocol 3 - optimized verifiable computing protocol for avoiding TAP to know! before auditing

4.3. Justification of Design

The proposed schemes were designed due to the follow-ing advantages.

Privacy preservation: the scheme ensures data mining/processing/computing privacy at CSP. CSP has no way to learn the plain data of DPs' input and the ir processing output. Thus, it is impossible to intrude the privacy of data and related objects. ! cannot know ! , thus one ! is impossible to obtain the data of other DPs.

Verifiable computing : the correctness of the data processing result of CSP can be verified by TAP triggered by RP. Thus it is impossible for CSP to behave dishonestly or maliciously during data processing and computation.

Facticity enhancement: introducing TAP enhances the facility of DP. TAP can mine the collected data of CSP and analyze if the data source has some abnormal behaviours by comparing newly generated data patterns with historical ones and analyzing collected data from different DPs.

Context awareness: the scheme supports data pro-cessing under different contexts by applying different algorithms, requesting data processing results based on context IDs, and auditing the correctness of data processing in different contexts.

Generality: the scheme supports various data processing/computing/mining cases served at the cloud. Meanwhile, it supports auditing data processing/computing/mining at a distrusted or semi-trusted CSP under any situations and contexts. Limitation could be caused by the shortcomings of FHE operations. But our scheme provides a generic framework to perform verifiable cloud computing. It is flexible to adopt the pro-posed four protocols to satisfy different security de-mands, as analyzed in Section 5.6.

5.PERFORMANCE ANALYSIS AND EVALUATION

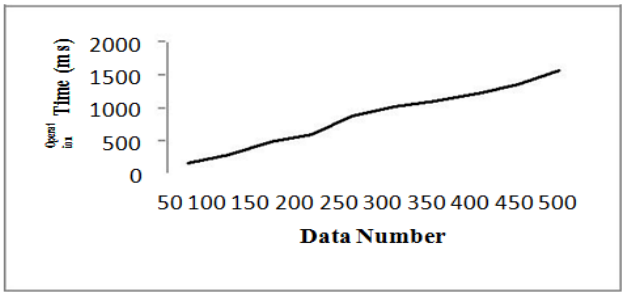
We evaluated the performance of four protocols by analyzing and comparing their security, computational complexity, and communication cost. We then report the re-sults of experimental performance tests on operation time.

We implemented the proposed protocols in a work-station with Intel(R) CoreTM i7 4710HQ CPU and 8 -GB RAM, running Ubuntu 14.04 that virtually executes the functions of DP, CSP, TAP and RP based on libraries NTL [34], GMP [35], and FHE [36]. In our implementation, we applied BGV full homomorphic encryption [33], RSA for Public Key Cryptosystem (PKC) and SHA-1 hash func-tion. We used a function provided by the FHE library to randomly generate plaintext with a length of 8 bytes to simulate the raw data provided by DP. Then we encrypt-ed plaintexts into ciphertexts and conducted a number of multiplications and additions to simulate data processes at CSP. For s imulating the audit process, we decrypted the ciphertext of data at TAP.

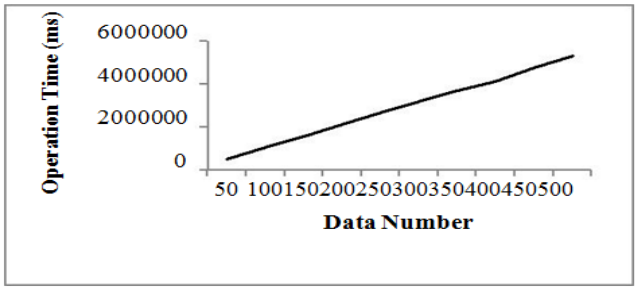
5.1 Operation Efficiency

We mainly tested the operation time of the scheme in terms of different operations: encryption, data processing and decryption, as shown in Fig.7-9. The "data number" in Fig.7-9 refers to the number of data input by DPs. In-Table 5, we report the operation time carried out by some basic operations. Compared with homomorphic opera-tions, RSA operation time is very trivial. This result sup-ports our analysis on computation complexity.

Encryption : Homomorphic encryption mainly takes place at DPs. Each encryption costs 1200 milliseconds. The encryption time is lineally increased with the number of data collected by DP, as shown in Fig.6. Each DP en crypts its data separately.



(a)



(b)

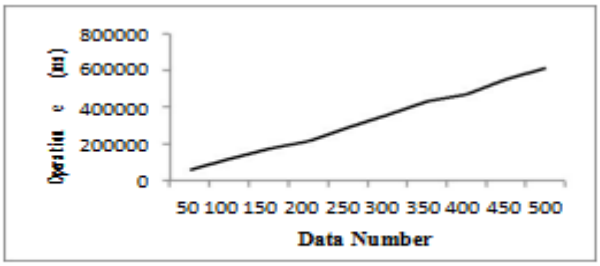
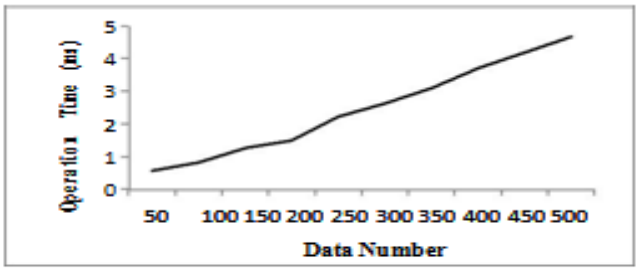
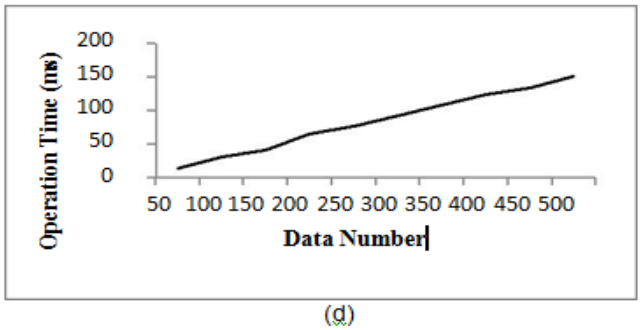


Fig. 7. Operation time of homomorphic encryption



(c)



(d)

Fig. 3. (a) Operation time of ciphertext addition; (b) Operation time of ciphertext multiplication; (c) Operation time of plaintext addition; (d) Operation time of plaintext multiplication

Data Processing: Data processing takes place at CSP and TAP. CSP deals with data in encrypted forms, while TAP in plaintext forms during auditing. Since most algo-rithms can be divided into a number of additions and multiplications, we only tested these two operations. As shown in Fig.8.a, the operation time of ciphertext addition almost proportionally increases with the number of data, which costs about 4 milliseconds (ms) to add two cipher-texts. The operation time of ciphertext multiplication

Table -1: OPERATION TIME AT EACH SYSTEM ENTITY IN PROTOCOL (UNIT: MILLISECOND)

Operation			Time
DP	Data encryption		1200
	Signature		0.3
CSP	Data processing (ciphertext)	Single addition	4
		Single multiplica-Tion	10000
TAP	Data decryption (per data)		600
	Data processing (plaintext)	Single addition	0.008
		Single multiplica-Tion	0.3
RP	Data decryption		600

Table -2: OPERATION TIME AT TAP IN THE ACCESS OF DATA PROCESSING RESULT (UNIT: MILLISECOND)

Protocol	Operation	Time
Protocol 1	1 RSA encryption	0.3
Protocol 2	1 homomorphic decryption + 1 RSA encryption	600.3
Protocol 3	1 homomorphic decryption + 1 homomorphic encryption	1800
Protocol 4	1 homomorphic decryption + 1 RSA encryption	600.3

Table 7, Table 8 and Table 9 show the differences of operation time of the four protocols at TAP, RP and CSP in terms of data processing result access. We can see from Table 7 that, Protocol 1 is the most efficient design with regard to the computation cost at TAP, while Protocol 3 is the

least efficient and Protocol 2 and Protocol 4 sit in the middle. From Table 8, we can conclude that the computa-tion workload is not heavy in each protocol. But Protocol 2 and Protocol 4 work very efficiently at RP. With regard to the computation cost at CSP, Protocol 3 is the least effi-cient, while Protocols 1, 2 and 4 have zero or a bit compu-tation overhead.

Table -3: OPERATION TIME AT CSP IN THE ACCESS OF DATA PROCESSING RESULT (UNIT: MILLISECOND)

Protocol	Operation	
Protocol 1	None	None
Protocol 2	None	None
Protocol 3	1 homomorphic encryption and 1 homomorphic multiplication	11200
Protocol 4	1 RSA encryption	0.3

5.2 Scalability

DPs: In our scheme, DPs only need to conduct homo-morphic encryptions that take about 1200ms on each data in our test. Each signature generation only takes 0.3ms, which can be ignored. Notably, the number of DPs does not affect the above operation time. This means that no matter how many DPs send their data to CSP, the time spent at each DP is fixed.

CSP: CSP takes responsibility to process ciphertexts. Since the ciphertext is the data encrypted by a full ho-momorphic encryption algorithm, the computation of the ciphertext could be very complicated. It is indicated that this cost is proportional to the number of data uploaded to CSP regarding a specific context that costs about 4ms for one addition and 10000ms for one multiplication. As-sumed that the processing capability at CSP is powerful

6. CONCLUSIONS

In this paper,we propose an effective a scheme of verifiable computing with context aware and privacy preservation,.In a further deployment an auditing protocol to verify the correctness of encrypted data processing by applying a Trusted Auditing Proxy (TAP). Concretely, a Data Provider (DP) encrypts its collected data with the homomorphic key offered by the TAP and signs it with a data context identifier (ID).It analyzed scheme security and evaluate the performance of the proposed protocols through rigorous analysis and comparison in order to show their pros and cons, as well as applicability.

REFERENCES

[1] Wang, C. Wang, K. Ren, W. Lou, J. Li, "Enabling public auditability and data dynamics for storage security in cloud computing", IEEE Trans. Parallel Distrib. Syst., vol. 22, pp. 847-859, May 2011.

[2] Wang, S. S. M. Chow, Q. Wang, K. Ren, W. Lou, "Privacy-preserving public auditing for secure cloud storage", IEEE Trans. Comput., vol. 62, no. 2, pp. 362-375, Feb. 2013.

[3] W. Liu, S.-S. Luo, Y.-B. Wang, Z.-T. Jiang, "A protocol of secure multi-party multi-data ranking and its application in privacy preserving sequential pattern mining", Proc. 4th Int. Joint Conf. Comput. Sci. Optim. (CSO), pp. 272-275, Apr. 2011.