RESEARCH PAPER | Engineering | Volume : 2 | Issue : 2 | November 2012 | ISSN - 2249-555X

Linear Programming in Cloud Computing for Practical Outsourcing

KEYWORDS: Linear Programming, Cloud Servers, Encryption, Decryption

N.Srujan Kumar — PG Student, LIET, Hyderabad, India,

Murtaza Ahmed Khan — Assoc.Professor, LIET, Hyderabad, India,

ABSTRACT *Cloud computing has great potential of providing robust computational power to the society at reduced cost. It enables clients with limited computational resources to outsource their large computation workloads to the cloud, and economically enjoy the massive computational power, bandwidth, storage, and even appropriate software that can be shared in a pay-per-use manner. Despite the tremendous benefits, security is the primary obstacle that prevents the wide adoption of this promising computing model, especially for clients when their confidential data are consumed and produced during the computation. Treating the cloud as an intrinsically insecure computing platform from the viewpoint of the cloud clients, the design mechanisms that not only protect sensitive information by enabling computations with encrypted data, but also protect clients from hateful behaviors by enabling the validation of the computation result. Such a method of general safe calculation outsourcing was newly shown to be viable in theory, but to propose mechanisms that are practically capable remains a very challenging problem*

I. INTRODUCTION

Cloud Computing presents well-situated on-demand network access to a shared group of configurable computing resources that can be quickly organized with great effectiveness and negligible management operating cost [1]. The benefit of cloud model is computation outsourcing, where the computational authority of cloud clients is no longer restricted by their resource-constraint devices. By outsourcing the workloads into the cloud, clients could enjoy the precisely unrestricted computing resources in a pay-per-use mode without consigning any large capital expenditures in the procurement of hardware, software or the operational. In spite of the tremendous profit, outsourcing computation to the commercial community cloud is also reluctant clients' direct control over the systems that put away and produce their data during the computation, which predictably brings in new security apprehensions and challenges towards this talented computing replica [2].

The outsourced calculation workloads often surround sensitive data, such as the financial records, research data, or identifiable health data etc. To battle against illegal information leakage, responsive data have to be encrypted before outsourcing so as to provide end-to-end data confidentiality guarantee in the cloud and beyond. Nevertheless, ordinary data encryption methods in essence

avoid cloud from performing any significant operation of the underlying plaintext data making the computation over encrypted data a very hard difficulty [3]. The outfitted details inside the cloud are not obvious enough to clients [4]. As a result, there do exist various motivations for cloud server to behave faithlessly and to return wrong results, i.e., they may act beyond the traditional semi-honest model. For example, for the computations that need a large amount of computing resources, there are huge monetary incentives for the cloud to be "lethargic" if the clients cannot tell the exactness of the output. Besides, possible software virus, hardware failures might also influence the quality of the computed results. The cloud is basically not secure from the viewpoint of clients. Without providing a method for secure computation outsourcing, i.e., to protect the responsive input and output data of the workloads and to authenticate the honesty of the computation result, it would be hard to expect cloud clients to turn over control of their workloads from local machinery to cloud exclusively based on its financial savings and resource flexibility.

For practical kindness, such a design should advance and ensure that clients perform fewer amounts of process following the mechanism than completing the computations by themselves directly. Otherwise, there is no point for clients to request help from cloud. Although some graceful designs on secure outsourcing of scientific computations, sequence comparisons, and matrix multiplication etc. have been proposed in the work, it is still hardly possible to apply them directly in a practically efficient manner, especially for large problems. In those advances, either important cloud-side cryptographic computations [6, 7 and 8] or multi-round interactive practice executions [5], or enormous communication difficulties are involved. In brief, practically well-organized mechanisms with instant practices for secure computation outsourcing in cloud are still absent [10]. Focusing on business computing and optimization tasks, in this work, the efficient mechanisms for secure outsourcing of Linear Programming (LP) computations are studied. LP is an algorithmic and computational device which detains the first order effects of a variety of system parameters that should be optimized, and is vital to engineering optimization. It has been extensively used in various engineering disciplines that examine and optimize real-world systems, such as packet routing, flow control, etc. Because LP computations require a considerable amount of computational power and usually involve confidential data, The explicit decomposition of the LP computation outsourcing into public LP solvers running on the cloud and private LP parameters owned by the client are proposed.

The flexibility of such decomposition allows us to discover higher-level concept of LP computations than the general path representation for the practical effectiveness. This higher level demonstration allows us to apply a set of efficient privacy-preserving problem conversion methods, including matrix multiplication and affine mapping, to convert the original LP problem into some arbitrary one while protecting the sensitive input/output data. One vital benefit of this higher level trouble conversion method is that existing algorithms and devices for LP solvers can be directly recycled by the cloud server. Although the general mechanism clear at circuit level, e.g. [9], can even allow the client to cover the fact that the outsourced computation is LP, the more severe security measure than necessary would greatly influence the effectiveness are forced. To authenticate the computation result, the truth that the result is from cloud server solving the conversed LP problem is operated. In general, the basic duality theorem together with the piece-wise creation of supporting LP problem to derive a set of essential and enough situations that the correct result must fulfilled. Such a method of result explanation can be very well-organized and acquires close-to-zero additional operating cost on both client and cloud

Volume : 2 | Issue : 2 | November 2012 | ISSN - 2249-555X

server. With properly verified result, client can use the secret conversion to map back the desired solution for his original LP problem. The involvements are summarized as follows:

II. PROBLEM STATEMENT

The calculation outsourcing architecture engages two dissimilar entities, as shown in Figure 1. the cloud client, who has large amount of costly LP problems to be outsourced to the cloud; the Cloud Server (CS), which has important computation resources and provides effectiveness computing services, such as hosting the public LP solvers in a pay-per-use manner. The client has a large-scale LP problem Φ to be solved. However, due to need of computing resources, like processing power, memory, and storage etc., can't carry out such classy computation locally. Thus, the client alternatives to CS for solving the LP computation and influences its computation capacity in a pay-per-use manner. Instead of directly sending original problem ϕ, the client first uses a secret key K to map Φ into some encrypted version ΦK and outsourcers problem ΦK to CS. It uses its public LP solver to get the answer of ΦK and provides a correctness proof, but it is supposed to learn nothing or little of the sensitive data contained in the original problem descriptionϕ. After receiving the solution of encrypted problem ΦK, the client should be able to first verify the answer via the affixed proof. If it's right, it uses the secret K to map the output into the desired reply for the original problem Φ. The security threats faced by the computation representation primarily come from the cruel behavior of CS.

The CS may behave beyond "honest-but- inquiring", i.e. the semi-honest model that was implicit by many earlier researches, either because it means to do so or because it is compromised. The CS may be persistently interested in analyzing the encrypted input sent by the client and the encrypted output produced by the computation to learn the sensitive data as in the semi-honest model. In addition, CS can also behave faithlessly or intentionally damage the computation, e.g. to lie about the result to save the computing resources, while eager not to be caught at the same time. Finally note that the communication channels between each cloud server and the client is authenticated and dependable, which can be achieved in practice with little overhead. These verification handshakes are absent in the following appearance. An optimization problem is usually formulated as a mathematical programming problem that looks for the values for a set of decision variables to minimize the purpose of function representing the cost subject to a set of constraints. For LP, the aim of the function is an affine function of the decision variables, and the restrictions are a system of linear equations and inequalities. Since a restriction in the form of a linear inequality can be articulated as a linear equation by introducing a non-negative floppy variable, and a free decision variable can be expressed as the difference of two non-negative auxiliary variables, any LP problem can be expressed in the following standard form minimize cT x subject to

Ax = b, x ≥ 0. (1)

Here x is an N × 1 vector of decision variables, A is an M × N matrix, and both c and b are N × 1 vectors. It can be assumed further that M ≤ N and that A has full row rank; otherwise, extras rows can always be eliminated from A.

In this work, a general study indicates minimizing cT x subject to

Ax = b, Bx ≥ 0. (2)

In Eq. (2), the non-negative requirements in Eq. (1) by requiring that each component of Bx to be non-negative, where B is an N × N non-singular matrix, i.e. Eq. (2) de-generates to Eq. (1) when B is the identity matrix. Thus, the LP problem can be defined via the tuple Φ = (A, B, b, c) as input, and the solution x as output are replaced.

III. THE PROPOSED METHODOLOGY

The LP outsourcing method provides a complete outsourcing solution for not only the privacy protection of problem input or output, but also its efficient result checking. An overview of secure LP outsourcing design framework and discuss a few basic techniques and their demerits, which leads to a stronger problem transformation design utilizing affine mapping. The effective result verification by leveraging the duality property of LP is discussed. Finally, the full scheme description is given. Before presenting the details of our proposed mechanism, the study in this section a few basic techniques and show that the input encryption based on these techniques along may result in an unsatisfactory mechanism. However, the analysis will give insights on how a stronger mechanism should be designed. Note that to simplify the presentation, the cloud server honestly performs the computation, and defer the discussion on soundness.

A. Hiding inequality constraints (B):

The client cannot convert the inequality conditions in the similar way as used for the equality constraints. This is because for an random invertible matrix Q, Bx ≥ 0 is not equivalent to QBx ≥ 0 in general. To hide B, we can leverage the fact that a feasible solution to Eq. (2) must satisfy the equality constraints. To be more specific, the feasible regions defined by the following two groups of constraints are the same.

$$\begin{cases} \mathbf{Ax} = \mathbf{b} \\ \mathbf{Bx} \geq \mathbf{0} \end{cases} \Rightarrow \begin{cases} \mathbf{Ax} = \mathbf{b} \\ (\mathbf{B} - \lambda\mathbf{A})\mathbf{x} = \mathbf{B}'\mathbf{x} \geq \mathbf{0} \end{cases}$$

where _ is a randomly generated n×m matrix in K satisfying that |B ′| = |B − _A| 6= 0 and _b = 0. Since the condition _b = 0 is largely underdetermined, it leaves great flexibility to choose _ in order to satisfy the above conditions.

B. Hiding objective functions c and value cT x:

Given the widely application of LP, such as the estimation of business annual revenues or personal portfolio holdings etc., the information contained in objective function c and optimal objective value cT x might be as sensitive as the constraints of A, B, b. Thus, they should be protected, too. To achieve this, we apply constant scaling to the objective function, i.e. a real positive scalar γ is generated randomly as part of encryption key K and c is replaced by γc. It is not possible to derive the original optimal aim value cT x without knowing γ first, since it can be mapped to any value with the same sign. While hiding the objective value well, this method does leak structure-wise information of objective function c. namely; the number and position of zero-elements in c are not protected. Besides, the ratio between the elements in c is also preserved after constant scaling.

C. Summarization of basic techniques

The basic techniques would choose a secret key K = (Q, λ, γ) and encrypt the input tuple Φ into ΦK = (A′, B′, b′, γc), which gives reasonable strength of problem input hiding. Also, these techniques are clearly correct in the sense that solving ΦK would give the same optimal solution as solving Φ. However, it also implies that although input privacy is achieved, there is no output privacy. Essentially, it shows that although one can change the constraints to a completely different form, it is not necessary the feasible region defined by the constraints will change, and the adversary can leverage such information to gain knowledge of the original LP problem. Therefore, any secure LP method must be able to not only encrypt the constraints but also to encrypt the feasible region defined by the constraints.

D. Enhanced Techniques via Affine Mapping

To enhance the security strength of LP outsourcing, we must be able to change the feasible region of original LP and at the same time hide output vector x during the problem input encryption. To encrypt the feasible region of Φ by applying an affine mapping on the decision variables x is proposed. This design principle is based on the following observation: ideally, randomly transform the feasible area of problem Φ from one vector space to another and keep the mapping function as the secret key, there is no way for cloud server to learn the original practicable area information. Further, such

Volume : 2 | Issue : 2 | November 2012 | ISSN - 2249-555X

a linear mapping also serves the important purpose of output hiding, as illustrated below.

Let M be an N × N non-singular matrix and r be an N × 1 vector. The affine mapping defined by M and r transforms x into y = M−1(x + r). Since this mapping is an one-to-one mapping, the LP problem Φ in Eq. (2) can be expressed as the following LP problem of the decision variables y.

minimize **cT My − cT r**

subject to **AMy = b + Ar**

BMy ≥ Br.

Using the basic techniques, this LP problem can be further transformed to

minimize γ**cT My**

subject to **QAMy = Q(b + Ar),**
BMy − λQAMy ≥ Br − λQ (b + Ar).

One can denote the constraints of above LP via Eq. (3):

$$\begin{cases} A' = QAM \\ B' = (B - \lambda QA)M \\ b' = Q(b + Ar) \\ c' = \gamma M^T c \end{cases} \qquad (3)$$

If the following conditions hold, |B′| 6= 0,

_b′ = Br, and b + Ar 6= 0, (4)

then the LP problem _K = (A′,B′, b′, c′) can be formulated via Eq. (5), minimise

c′T y subject if to A′y = b′, B′y ≥ 0. (5)

IV. PERFORMANCE ANALYSIS

A. Theoretic Analysis

1) Client Side Overhead: According to our method, client side computation transparency consists of key generation, problem encryption operation, and result verification, which corresponds to the three algorithms KeyGen, ProbEnc, and ResultDec, respectively. Because KeyGen and Result-Dec only require a set of random matrix generation as well as vector-vector and matrix-vector multiplication, the computation complexity of these two algorithms are upper bounded via O(N2). Thus, it is straight-forward that the most time-consuming operations are the matrix-matrix multiplications in problem encryption algorithm ProbEnc. Since M ≤ N, the time complexity for the client local computation is thus asymptotically the same as matrix-matrix multiplication, i.e., O (N^2) for some 2 < ρ ≤ 3. In our experiment, the matrix multiplication is implemented via standard cubic-time method, thus the overall computation operating cost is O (N3).

2) Server Side Overhead: For CS, its only computation transparency is to solve the encrypted LP problem ΦK as well as generating the result proof, both of which correspond to the algorithm ProofGen. If the encrypted LP problem ΦK belongs to normal case, cloud server just solves it with the dual optimal solution as the result proof, which is usually readily available in the current LP solving algorithms and incurs no additional cost for cloud If the encrypted problem ΦK does not have an optimal solution, additional auxiliary LP problems can be solved to provide a proof. Because for general LP solvers, phase I method is always executed at first to determine the initial feasible solution, proving the auxiliary LP with optimal solutions also introduces little additional overhead. Thus, in all the cases, the computation complexity of the cloud server is asymptotically the same as to solve a normal LP problem, which usually requires more than O (N3) time. Evidently, the client will not spend more time to encrypt the problem and solve the problem in the cloud than to solve the problem on his own. Therefore, in theory, the proposed mechanism would allow the client to outsource their LP problems to the cloud and gain great computation savings. The asymmetric speedup captures the client efficiency gain via LP outsourcing. The cloud efficiency captures the overall computation cost on cloud introduced by solving encrypted LP problem, which should ideally be as closer to 1 as possible.

B. Experiment Results

The practical efficiency of the proposed secure and verifiable LP outsourcing scheme with experiments are assessed. We implement the proposed mechanism including both the client and the cloud side processes in Matlab and utilizes the MOSEK optimization through its Mat lab interface to solve the original LP problem Φ and encrypted LP problem ΦK. Both client and cloud server computations in our experiment are conducted on the same workstation with an Intel Core 2 Duo processor running at 1.86 GHz with 2 GB RAM. In this way, the practical efficiency of the proposed mechanism can be assessed without a real cloud environment. We also ignore the communication latency between the clients and the cloud for this application since the computation dominates the running time as evidenced by our experiments. Our randomly generated test benchmark covers the small and medium sized problems. All these benchmarks are for the normal cases with feasible optimal solutions. Since in practice the infeasible cases for LP computations are very rare, we do not conduct those experiments for the current preliminary work and leave it as one of our future tasks.

V. CONCLUSIONS

In this work, the difficulty of securely outsourcing LP computations in cloud computing, and provide such a practical method design which fulfills input/output privacy, cheating flexibility, and efficiency are formalized. By unambiguously decomposing LP computation outsourcing into public LP solvers and private data, our method design is able to explore appropriate security tradeoffs via higher level LP computation than the general circuit illustration. The problem conversion methods that enable clients to secretly transform the original LP into some random one while protecting sensitive input/output information are developed. Duality inspect theorem derives a set of necessary and sufficient situation for result verification. Such a cheating flexibility design can be bundled in the overall method with close-to-zero additional operating cost. Both security analysis and experiment results display the immediate practicality of the proposed method.

REFERENCE [1] P. Mell and T. Grance, "Draft nist working definition of cloud computing," Referenced on Jan. 23rd, 2010 Online at http://csrc.nist.gov/ groups/ SNS/cloud-computing/index.html, 2010. [2] Cloud Security Alliance, "Security guidance for critical areas of focus in cloud computing," 2009, online at http://www.cloudsecurityalliance.org. [3] C. Gentry, "Computing arbitrary functions of encrypted data," Commun. ACM, vol. 53, no. 3, pp. 97–105, 2010. [4] Sun Microsystems, Inc., "Building client trust in cloud computing with transparent security," 2009, online at https://www.sun.com/offers/ details/sun transparency.xml. [5] M. J. Atallah, K. N. Pantazopoulos, J. R. Rice, and E. H. Spafford, "Secure outsourcing of scientific computations," Advances in Computers, vol. 54, pp. 216–272, 2001. [6] S. Hohenberger and A. Lysyanskaya, "How to securely outsource cryptographic computations," in Proc. of TCC, 2005, pp. 264–282. [7] M. J. Atallah and J. Li, "Secure outsourcing of sequence comparisons," Int. J. Inf. Sec., vol. 4, no. 4, pp. 277–287, 2005. [8] D. Benjamin and M. J. Atallah, "Private and cheating-free outsourcing of algebraic computations," in Proc. of 6th Conf. on Privacy, Security, and Trust (PST), 2008, pp. 240–245. [9] R. Gennaro, C. Gentry, and B. Parno, "Non-interactive verifiable computing: Outsourcing computation to untrusted workers," in Proc. of CRYPTO'10, Aug. 2010. [10] M. Atallah and K. Frikken, "Securely outsourcing linear algebra com-putations," in Proc. of ASIACCS, 2010, pp. 48–59.