International Journal of Current Trends in Engineering & Research (IJCTER)
e-ISSN 2455–1392 Volume 2 Issue 2, February 2016 pp. 8-11
Scientific Journal Impact Factor : 3.468
http://www.ijcter.com

CLOUD SECURITY IN PRACTICAL OUTSOURCING BY LINEAR PROGRAMMING

Sathishkumar J [1], Priya A[2]

[1,2] Assistant Professor , Department of Computer Science and Engineering
Sasurie College of Engineering, Vijayamangalam, Tiruppur

Abstract—Cloud Computing has great potential of providing robust computational power to the society at reduced cost. It enables customers 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 customers 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 customers must design mechanisms that not only protect sensitive information by enabling computations with encrypted data, but also protect customers from malicious behaviors by enabling the validation of the computation result. Such a mechanism of general secure computation outsourcing was recently shown to be feasible in theory, but to design mechanisms that are practically efficient remains a very challenging problem.

I. INTRODUCTION

Cloud Computing provides convenient on-demand network access to a shared pool of configurable computing resources that can be rapidly deployed with great efficiency and minimal management overhead. One fundamental advantage of the cloud paradigm is computation outsourcing, where the computational power of cloud customers is no longer limited by their resource-constraint devices. By outsourcing the workloads into the cloud, customers could enjoy the literally unlimited computing resources in a pay-per-use manner without committing any large capital outlays in the purchase of hardware and software and/or the operational overhead therein.

Despite the tremendous benefits, outsourcing computation to the commercial public cloud is also depriving customers' direct control over the systems that consume and produce their data during the computation, which inevitably brings in new security concerns and challenges towards this promising computing model .On the one hand, the outsourced computation workloads often contain sensitive information, such as the business financial records, proprietary research data, or personally identifiable health information etc. To combat against unauthorized information leakage, sensitive data have to be encrypted before outsourcing so as to provide end- to-end data confidentiality assurance in the cloud and beyond. However, ordinary data encryption techniques in essence pre- vent cloud from performing any meaningful operation of the underlying plaintext data, making the computation over encrypted data a very hard problem. On the other hand, the operational details inside the cloud are not transparent enough to customers. As a result, there do exist various motivations for cloud server to behave unfaithfully and to return incorrect results, i.e., they may behave beyond the classical semi- honest model. For example, for the computations that require a large amount of computing resources, there are huge financial incentives for the cloud to be "lazy" if the customers cannot tell the correctness of the output. Besides, possible software bugs, hardware failures, or even outsider attacks might also affect the quality of the computed results. Thus, argue that the cloud is intrinsically *not secure* from the viewpoint of customers. Without providing a mechanism for secure computation outsourcing, i.e., to protect the sensitive input and output information of the workloads and to validate the integrity of the computation

@IJCTER-2016, All rights Reserved

result, it would be hard to expect cloud customers to turn over control of their workloads from local machines to cloud solely based on its economic savings and resource flexibility. For practical consideration, such a design should further ensure that customers perform fewer amounts of operations following the mechanism than completing the computations by themselves directly. Otherwise, there is no point for customers to seek help from cloud.

However, applying this general mechanism to our daily computations would be far from practical, due to the extremely high complexity of FHE operation as well as the pessimistic circuit sizes that cannot be handled in practice when constructing original and encrypted circuits. This overhead in general solutions motivates us to seek efficient solutions at higher abstraction levels than the circuit representations for specific computation outsourcing problems. Although some elegant designs on secure outsourcing of scientific computations, se- quence comparisons, and matrix multiplication etc. have been proposed in the literature, it is still hardly possible to apply them directly in a practically efficient manner, especially for large problems. In those approaches, either heavy cloud-side cryptographic computations or multi-round interactive protocol executions or huge communication complexities, are involved. In short, practically efficient mechanisms with immediate practices for secure computation outsourcing in cloud are still missing.

II. PROBLEM STATEMENT

A. System and Threat Model

Consider a computation outsourcing architecture involving two different entities, the cloud customer, who has large amount of computationally expensive LP problems to be outsourced to the cloud; the cloud server (CS), which has significant computation resources and provides utility computing services, such as hosting the public LP solvers in a pay-per-use manner.

The customer has a large-scale linear programming problem Φ (to be formally defined later) to be solved. However, due to the lack of computing resources, like processing power, memory, and storage etc., it cannot carry out such expensive computation locally. Thus, the customer resorts to CS for solving the LP computation and leverages its computation capacity in a pay-per-use manner. Instead of directly sending original problem Φ, the customer first uses a secret K to map Φ into some encrypted version ΦK and outsources problem ΦK to CS. CS then uses its public LP solver to get the answer of ΦK and provides a correctness proof Γ, but it is supposed to learn nothing or little of the sensitive information contained in the original problem description Φ. After receiving the solution of encrypted problem ΦK, the customer should be able to first verify the answer via the appended proof Γ. If it's correct, then uses the secret K to map the output into the desired answer for the original problem Φ.

B. Design Goals

To enable secure and practical outsourcing of LP under the aforementioned model, our mechanism design should achieve the following security and performance guarantees.

1) **Correctness:** Any cloud server that faithfully follows the mechanism must produce an output that can be decrypted and verified successfully by the customer.
2) **Soundness:** No cloud server can generate an incorrect output that can be decrypted and verified successfully by the customer with non-negligible probability.
3) **Input/output privacy:** No sensitive information from the customer's private data can be derived by the cloud server during performing the LP computation.
4) **Efficiency:** The local computations done by customer should be substantially less than solving the original LP on its own. The computation burden on the cloud server should be within the comparable time complexity of existing practical algorithms solving LP problems

@IJCTER-2016, All rights Reserved

C. Background on Linear Programming

An optimization problem is usually formulated as a mathematical programming problem that seeks the values for a set of decision variables to minimize (or maximize) an objective function representing the cost subject to a set of constraints. For linear programming, the objective function is an affine function of the decision variables, and the constraints are a system of linear equations and inequalities. Since a constraint in the form of a linear inequality can be expressed as a linear equation by introducing a non-negative slack variable, and a free decision variable can be expressed as the difference of two non-negative auxiliary variables, any linear programming problem can be expressed in the following standard form,minimize $c^T x$ subject to $Ax = b, x \geq 0$. (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 $m \leq n$ and that A has full row rank; otherwise, extras rows can always be eliminated from A. In this paper, study a more general form as follows, minimize $c^T x$ subject to $Ax = b, Bx \geq 0$. (2). In Eq. (2), it replace the non-negative requirements in Eq. (1) by requiring that each component of Bx to be non-negative,where B is an $n \times n$ non-singular matrix, i.e. Eq. (2) degenerates to Eq. (1) when B is the identity matrix.

III. THE PROPOSED SCHEMES

This section presents the LP outsourcing scheme which provides a complete outsourcing solution for – not only the privacy protection of problem input/output, but also its efficient result checking. So it start from 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. Then it discusses effective result verification by leveraging the duality property of LP. Finally, it gives the full scheme description.

A. Mechanism Design Framework

Here propose to apply problem transformation for mechanism design. The general framework is adopted from a generic approach while our instantiation is completely different and novel. In this framework, the process on cloud server can be represented by algorithm Proof Gen and the process on customer can be organized into three algorithms (KeyGen, ProbEnc, ResultDec). These four algorithms are summarized below and will be instantiated later.

- KeyGen(1k) → {K }. This is a randomized key generation algorithm which takes a system security parameter k, and returns a secret key K that is used later by customer to encrypt the target LP problem.
- ProbEnc(K, Φ) → {ΦK }. This algorithm encrypts the input tuple Φ into ΦK with the secret key K. According to problem transformation, the encrypted input ΦK has the same form as Φ, and thus defines the problem to be solved in the cloud.
- ProofGen(ΦK) → {(y, Γ)}. This algorithm augments a generic solver that solves the problem ΦK to produce both the output y and a proof Γ. The output y later decrypts to x, and Γ is used later by the customer to verify the correctness of y or x.
- ResultDec(K, Φ, y, Γ) → {x, ⊥}. This algorithm may choose to verify either y or x via the proof Γ. In any case, a correct output x is produced by decrypting y using the secret K. The algorithm outputs ⊥ when the validation fails, indicating the cloud server was not performing the computation faithfully.

IV. CONCLUDING REMARKS

It formalizes the problem of securely outsourcing LP computations in cloud computing, and provides such a practical mechanism design which fulfills input/output privacy, cheating resilience, and efficiency. By explicitly decomposing LP computation outsourcing into public LP solvers and

@IJCTER-2016, All rights Reserved

private data, our mechanism design is able to explore appropriate security/efficiency tradeoffs via higher level LP computation than the general circuit representation. Hence to develop problem transformation techniques that enable customers to secretly transform the original LP into some arbitrary one while protecting sensitive input/output information. Such a cheating resilience design can be bundled in the overall mechanism with close-to-zero additional overhead. Both security analysis and experiment results demonstrate the immediate practicality of the proposed mechanism.

Plan to investigate some interesting future work as follows: 1) devise robust algorithms to achieve numerical stability; 2) explore the sparsity structure of problem for further efficiency improvement; 3) establish formal security framework; 4) extend our result to non-linear programming computation outsourcing in cloud.

REFERENCES

[1] 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.
[2] M. J. Atallah and J. Li, "Secure outsourcing of sequence comparisons,"Int. J. Inf. Sec., vol. 4, no. 4, pp. 277–287, 2005.
[3] C. Gentry, "Fully homomorphic encryption using ideal lattices," in Proc of STOC, 2009, pp. 169–178.
[4] D. Luenberger and Y. Ye, Linear and Nonlinear Programming, 3rd ed.Springer, 2008.
[5] R. Gennaro, C. Gentry, and B. Parno, "Non-interactive verifiable computing:Outsourcing computation to untrusted workers," in Proc. Of CRYPTO'10, Aug. 2010.
[6] Sun Microsystems, Inc., "Building customer trust in cloud computing with transparent security," 2009, online at https://www.sun.com/offers/ details/sun transparency.xml.

@IJCTER-2016, All rights Reserved