IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 12, Issue 6 (Jul. - Aug. 2013), PP 87-90
www.iosrjournals.org

Implementation of Secure Cloud Storage Gateway using Symmetric Key Algorithm

Vijayalakshmi M.M.[1], Prof. Sharayu pradeep [2]
[1,2]*Dept. of CSE CIT, Gubbi, Tumkur, Karnataka, INDIA,*

Abstract: *Focusing on engineering computing and optimization tasks, this paper investigates secure outsourcing of widely applicable linear programming (LP) computations. In order to achieve practical efficiency, our mechanism design explicitly decomposes the LP computation outsourcing into public LP solvers running on the cloud and private LP parameters owned by the customer. The resulting flexibility allows us to explore appropriate security/ efficiency tradeoff via higher-level abstraction of LP computations than the general circuit representation. In particular, by formulating private data owned by the customer for LP problem as a set of matrices and vectors, we are able to develop a set of efficient privacy-preserving problem transformation techniques, which allow customers to transform original LP problem into some arbitrary one while protecting sensitive input/output information. To validate the computation result, we further explore the fundamental duality theorem of LP computation and derive the necessary and sufficient conditions that correct result must satisfy. Such result verification mechanism is extremely efficient and incurs close-to-zero additional cost on both cloud server and customers. Extensive security analysis and experiment results show the immediate practicability of our mechanism design.*
Keywords*: cloud customer, cloud server,fully homomorphic encryption (FHE), linearprogramming,*

I. Introduction

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.

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 prevent 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.

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 prevent 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.Fully homomorphic encryption (FHE) scheme, a general result of secure computation outsourcing has been shown viable in theory, where the computation is represented by an encrypted combinational Boolean circuit that allows to be evaluated with encrypted private inputs.

II. Problem Statement

2.1 System and Threat Model

We consider a computation outsourcing architecture involving two different entities, thecloud customer, who has large amount of computationally expensive LP problems to be outsourced to the cloud; the cloudserver (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_ (to be formally defined later) to be solved. However, due to the lack of computing resources, like processing power, memory, and storage etc., he 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 _, the customer first uses a secret K to map _ into some encrypted version _K and outsources problem _K to CS. CS then uses its public LP solver to get the answer of _K and provides a correctness proof , but it is supposed to learn nothing or little of the sensitive information contained in the original problem description _. After receiving the solution of encrypted problem _K, the customer should be able to first verify the answer via the appended proof. If it's correct, he then uses the secret K to map the output into the desired answer for the original problem _.

2.2. 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 his own. The computation burden on the cloud server should be within the comparable time complexity of existing practical algorithms solving LP problems.

2.3. 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, we study a more general form as follows,

Minimize $c^T x$ subject to $Ax = b, Bx \geq 0.$ (2)

In Eq. (2), we replace 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) degenerates to Eq. (1) when B is the identity matrix. Thus, the LP problem can be defined via the tuple _ = (A,B, b, c) as input, and the solution x as output.

III. Module Description

1. Mechanism Design Framework
2. Basic Techniques
3. Enhanced Techniques via Affine Mapping
4. Result Verification

3.1. Mechanism Design Framework:

We 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 ProofGen and the process on customer can be

organized into three algorithms (KeyGen, ProbEnc, and 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,_) → {_K}. This algorithm encrypts the input tuple _ into _K with the secret key K. According to problem transformation, the encrypted input _K has the same form as _, and thus defines the problem to be solved in the cloud.
• ProofGen (_K) → {(y, □)}. This algorithm augments a generic solver that solves the problem _K to produce both the output y and a proof □. The output y later
Decrypts to x, and □ is used later by the customer to verify the correctness of y or x.
• Result Dec (K, _, y, □) → {x,⊥}. This algorithm may choose to verify either y or x via the proof □. 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.

3.2. Basic Techniques

Before presenting the details of our proposed mechanism, we study in this subsection 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, we assume that the cloud server honestly performs the computation, and defer the discussion on soundness to a later section.
1) Hiding equality constraints (A, b): First of all, a randomly generated m × m non-singular matrix Q can be part of the secret key K. The customer can apply the matrix to Eq. (2) for the following constraints transformation,
$Ax = b \Rightarrow A'x = b'$
where $A' = QA$ and $b' = Qb$.

3.3. 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. We propose to encrypt the feasible region of _ by applying an affine mapping on the decision variables x. This design principle is based on the following observation: ideally, if we can arbitrarily transform the feasible area of problem _ from one vector space to another and keep the mappingfunction as the secret key, there is no way for cloud server to learn the original feasible area information. Further, such a linear mapping also serves the important purpose of output hiding.

3.4. Result Verification

Till now, we have been assuming the server is honestly performing the computation, while being interested learning information of original LP problem. However, such semihonest model is not strong enough to capture the adversary behaviors in the real world. In many cases, especially when the computation on the cloud requires a huge amount of computing resources, there exist strong financial incentives for the cloud server to be "lazy". They might either be not willing to commit service-level-agreed computing resources to save cost, or even be malicious just to sabotage any following up computation at the customers. Since the cloud server promises to solve the LP problem $_K = (A', B', b', c')$, we propose to solve the result verification problem by designing a method to verify the correctness of the solution y of _K. The soundness condition would be a corollary thereafter when we present the whole mechanism in the next section. Note that in our design, the workload required for customers on the result verification is substantially cheaper than solving the LP problem on their own, which ensures the great computation savings for secure LP outsourcing.

The LP problem does not necessarily have an optimal solution. There are three cases as follows.
• Normal: There is an optimal solution with finite objective value.
• Infeasible: The constraints cannot be all satisfied at the same time.
•Unbounded: For the standard form in Eq. (1), the objective function can be arbitrarily small while the constraints are all satisfied.

IV. Conclusion and future work

In this paper, for the first time, we formalize the problem of securely outsourcing LP computations in cloud computing, and provide 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 private data, our mechanism design is able to explore appropriate security/efficiency tradeoffs via higher level LP computation than the general circuit representation. We develop problem transformation techniques that

enable customers to secretly transform the original LP into some arbitrary one while protecting sensitive input/output information.

References

[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 incloud 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 customer trust in cloud computingwith 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.