تاریخچه مقاله:

تاریخ ارسال: ۹۸/۰۹/۱۰
تاریخ اصلاحات: ۹۸/۱۱/۱۲
تاریخ پذیرش: ۹۸/۱۲/۱۵
تاریخ انتشار: ۹۸/۱۲/۲۰

Keywords:

Cloud Computing
Security
Privacy-Preserving
Additive Homomorphic
Order-Preserving
Scheme.

Lightweight Additive Homomorphic Schemes for Cloud Outsourcing Scenarios

Somayeh Sobati.M[1]*
[1] Hakim Sabzevari University, Sabzevar, Iran

Abstract

Data privacy is a major concern in a cloud computing environment that uses the Database-as-a-Service model. Nevertheless, the existing encryption schemes are only partly homomorphic and are designed to enable for one particular method of computation to be carried out on encrypted data. To address these concerns, we have recently proposed new partially homomorphic schemes that preserve data privacy in a cloud database while still enabling encrypted data to be processed. The proposed schemes allow the processing of queries with an order-preserving partial homomorphic encryption scheme. We've shown that our schemes are a realistic framework that protect data privacy with reduced overhead.

روش ارجاع به مقاله:

S.Sobati.M, Lightweight Additive Homomorphic Schemes for Cloud Outsourcing Scenarios, Journal of Distributed Computing and Systems (JDCS) Vol.2, No.2, PP.170-177, 2020

Email: s.sobati@hsu.ac.ir

1. Introduction

Nowadays, data outsourcing scenarios tremendously grow with the advent of cloud computing. Cloud computing appeals businesses and organizations because of a wide variety of benefits such as cost saving and service benefits. Moreover, cloud computing provides higher availability, scalability and more effective disaster recovery rather than in-house operations [1]. One of the most notable cloud outsourcing services is database outsourcing (Database-as-a-Service), where individuals and organizations outsource data storage and management to a cloud Service Provider (SP). Such services allow storing sensitive data on a remote, untrusted SP and querying data on demand [2].

Although cloud data outsourcing induces many benefits, it also brings out security and privacy concerns. Privacy issues arise since sensitive data are stored, maintained and processed by an external third party (honest but curious SP).

Encrypted Query Processing. A naive solution to preserve data privacy and confidentiality is encrypting data locally before sending them into an untrusted server. Encryption protects the exposure of sensitive data even the server is compromised and ensures that an adversary will be unable to interpret them because they are encrypted.

When data are encrypted, query processing is not trivial. To overcome this problem, several solutions have been proposed. These approaches implement cryptographic techniques (e.g., order-preserving encryption or homomorphic encryption) that allow computations to be carried out on encrypted data. CryptDB [2] is one of the first systems that integrates efficient query processing over encrypted data into the database management system. CryptDB executes queries over encrypted data using a collection of efficient query-aware encryption schemes. CryptDB relies on Paillier's encryption scheme [3] to sum encrypted data at the server-side. Paillier's encryption scheme is not suitable for cloud outsourcing scenarios, mainly because:

(i) It employs modular multiplication to add ciphertexts at the server-side which is prohibitively expensive and,

(ii) It operates over large ciphertext values which induces storage overhead and affects query processing.

(iii) And finally, Paillier's scheme is not order-preserving and cannot support range queries. Hence, we introduce an efficient additive homomorphic scheme which is suited for outsourced data.

Secure Secret Splitting Schemes In this paper, we present a new Secure Secret Splitting Scheme, S4, which enables efficient query processing at the server's.

S4 shares similar flavors with Shamir's secret sharing scheme in that sensitive data is decomposed into shares [4].

In Shamir's secret sharing, a secret is mathematically decomposed into several shares and are stored at n participants's, with $k \leq n$ participants being required to reconstruct the secret. Secret sharing is very flexible and scalable, and one can easily imagine participants being SPs. In terms of databases, each record in a database is split into n shares that are distributed to independent data servers [2], [1]. Although theoretically one can execute in this way SQL queries on sensitive data, there are two issues: (i) such solutions require at least k non-colluding parties (SP's). This incurs significant costs in outsourcing data, (ii) computing queries over shares is expensive, both in terms of computations at SPs's and communications among the user and SPs.

Like Shamir's scheme, S4 decomposes sensitive data into shares (splits). In contrast to Shamir's scheme, there are only two participants, a SP and a user. Computations are performed on splits at the SP's. The secret can only be reconstructed by the user. S4 eliminates expensive operations i.e., modular multiplication for aggregation queries and reduces the total size of the ciphertext. Hence, S4 significantly improves computation and storage cost on the SP's compared to the previous solutions.

2. Background

Shamir's secret sharing

Secret sharing divides a secret piece of data into so-called shares that are stored at n participants'. One single participant has no means to reconstruct the secret; a subset of $k \leq n$ participants is required to reconstruct it.

Shamir's secret sharing scheme is based on polynomial interpolation. It works by defining a polynomial of degree $k-1$ with the secret as the constant term. Then, $n \geq k$ points are built on the polynomial. Since at least k points are needed to uniquely reconstruct a polynomial of degree $k-1$, knowing k points, the secret can be reconstructed using Lagrange interpolation [3].

To share a secret v_j, first a random polynomial $P_{v_j}(x)$ of degree *k-1* is built. Theoretically, the points and coefficients in Shamir's scheme are taken from $\mathbb{F}_p$, where p is a prime. The owner of the secret, first chooses a prime number p such that $p > v_j$ then picks $k-1$ random numbers $a_1, a_2, \ldots, a_{k-1}$ from $\mathbb{F}_p$ and sets $a_0 = v_j$ $P_{v_j}(x)$ passes through the point $(0, v_j)$.

$$P_{v_j}(x) = a_{k-1}x^{k-1} + \ldots + a_1 x + a_0 \quad (1)$$

Formula-1

To build n points over $P_{v_j}(x)$, a set of n distinct random elements in $\mathbb{F}_p$, $X = \{x_1, x_2, \ldots, x_n\}$, is chosen such that $x_i \neq 0 \forall i = 1, \ldots, n$. For each participant i, the corresponding share is $v_{i,j} = P_{v_j}(x_i)$. For each secret v_j, there are n points $(x_i, v_{i,j})$ through which the polynomial $P_{v_j}(x)$ passes [2]. Note that X must be highly protected from untrusted parties and adversaries.

Given any k shares, form k points $(x_i, v_{i,j})$ $i = 1, \ldots, k$, which polynomial $P_{v_j}(x_i)$ can be reconstructed using Lagrange interpolation (Equation 2).

$$P_{v_j} = \sum_{i=1}^{k} v_{i,j}\, \ell_i(x) \quad mod \ p \quad (2)$$

Formula-2

Where $\ell_i(x)$ is the Lagrange basis polynomial:

$$\ell_i(x) = \prod_{1 \leq j \leq k_2 j \neq i} (x - x_j)(x_i - x_j)^{-1} \quad mod \ p \quad (3)$$

Formula-3

And $(x_i - x_j)^{-1}$ is the multiplicative inverse of $(x_i - x_j)$.

The secret is the constant term of P_{v_j} and can be recovered from:

$$v_j = P_{v_j}(0) = \sum_{i=1}^{k} v_{i,j}\, \ell_i(0) \quad mod \ p \quad (4)$$

Formula-4

3. S4

Secure Secret Splitting Scheme, S4, is a new secret sharing scheme to securely store data, while allowing summation queries to be computed at a SP's, efficiently. S4 is based on Shamir's secret sharing. In S4, each secret v_j is split into $n = k$ meaningless shares $v_{1,j}, v_{2,j}, \ldots, v_{k,j}$. Since shares are not actually distributed to several participants, we call shares splits.

k- 1 splits, $v_{1,j}, v_{2,j}, \ldots, v_{k-1,j}$, are stored at the SP's and $v_{k,j}$ is stored in a trusted machine, e.g., at the user's. In order to reduce storage overhead at the user's, $v_{k,j}$ is considered to be the same for all secrets.

Since $v_{k,j}$ is fixed for all secrets, we denote it by v_k.

Data model. Let us assume a relational table T consists of one attribute A (additional attributes, if any, can be processed similarly). Suppose T has N tuples. We denote by v_j the j^{th} record of A. Consider $D = [l, r)$ the domain of A, with t distinct values $v_1, v_2, \dots, v_t$. For attribute A in T, k *-1* attributes A_i, i *= 1, ..., k-1* are created in table T' at the SP's, where each column A_i stores the i^{th} splits. Without loss of generality, we assume integer data type for *A*. Other types, e.g., reals, characters, strings, can be transformed into integers before splitting.

Splitting and Reconstruction Processes. First, a large p is picked such that $p > v_j$ $\forall v_j \in D$, then x_k and v_k are set up randomly from $\mathbb{F}_p$. For any secret v_j, a random polynomial $P_{v_j}(x)$ is built that passes through $(0, v_j)$ and (x_k, v_k). To this end, $k - 2$ points (a_i, b_i), $\forall i = 1, \dots, k - 2$ are chosen randomly from $\mathbb{F}_p$such that $a_i \neq x_k$ and $a_i \neq 0$ $\forall i = 1, \dots, k - 2$. Given k points (a_1, b_1), $(a_2, b_2), \dots, (a_{k-2}, b_{k-2}), (0, v_j)$, and (x_k, v_k), polynomial $P_{v_j}(x)$ is built using 2. Note that it is not necessary to store $k - 2$ random points because they are not needed for secret reconstruction.

To split v_j into $k - 1$ splits (since (x_k, v_k) is already fixed), a set of $k - 1$ distinct elements of $\mathbb{F}_p$, $X = \{x_1, x_2, \dots, x_{k-1}\}$, is chosen randomly, with $x_i \neq 0$ and $x_i \neq x_k$ $\forall i = 1, \dots, k - 1$. Then, splits are $v_{i,j} = P_{v_j}(x_i)$. Note that $\mathcal{K} = (X, (x_k, v_k))$ is considered as a private key for S4 and must be highly protected and kept hidden from the SP.

As in Shamir's secret sharing, the key observation is that *k* points are required to build a unique polynomial of degree k – 1. In order to reconstruct secret v_j, its k – 1 splits must be retrieved from the SP. Given points $(x_i, v_{i,j})$, $i = 1, \dots, k - 1$ and (x_k, v_k), which is stored at the user's, polynomial $P_{v_j}(x)$ can be reconstructed using 2. Its constant term is v_j.

4. Order-Preserving Secret Splitting

The theoretic security of Shamir's scheme guarantees data confidentiality, keeping in mind that $\mathcal{K}$ is hidden from the SP [5]. Secret splitting scheme provides the highest level of data confidentiality because the data splits are produced using random points. As a result, random polynomials are generated and consequently two equal values have different splits. Hence, S4 leaks no information about the distribution or the frequency of original values.

In outsourcing scenario, it is assumed that a secure scheme must handle queries for retrieving the requested splits [6]. While the random points are not stored, data retrieval becomes a problem in S4. To overcome this problem, we propose a new Order-Preserving S4 (OPS4) which allows querying over splits while preserves data confidentiality. First, we extend and redefine our secret splitting scheme to preserve the order of values in their corresponding splits.

OPS4. The first line of S4 extension focuses on a splitting method that preserves the order of attribute values over only one split values, e.g., the i^{th} split where $1 \leq i \leq k - 1$. We refer the i^{th} split as order-preserving split and we demonstrate its values as $ordv_{i,j}$.

In our system, *i* could be chosen between 1 and *k- 1*. To ease of presentation, in the remaining of this paper we assume $i = 1$, in the other word, we aim at preserving order in the over first splits of original values.

Since the first split is order-preserving split, *OPS4* splits each attribute value v_j into $k - 1$ splits $ordv_{1,j}, v_{2,j}, \dots, v_{k-1,j}$ as shown in Figure 1.

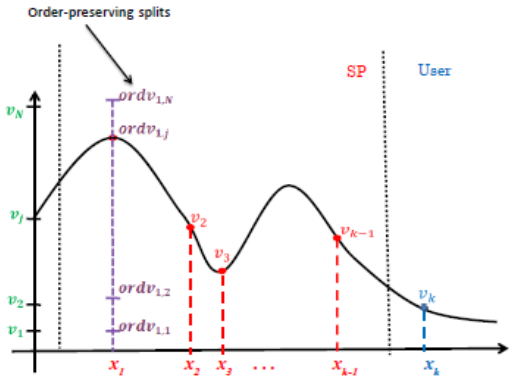

Figure 1. Order-preserving secret splitting

To this end, first, a new domain D' is considered $D' = [l', r')$ such that $l', r' \in \mathbb{Z}$ and $D' \gg D$. D' is partitioned into t intervals $d_l = [l_l, r_l) = [l' + (v_l - v_1)\frac{|D'|}{t}, l' + (v_l - v_1 + 1)\frac{|D'|}{t})$ $l = 1, \dots, t$. Then, each distinct value v_l $l = 1, \dots, t$ of A is assigned into interval d_l.

For splitting value v_j, a random value $ordv_{1,j}$ is chosen from the corresponding interval as the first split of v_j, which guarantees that the order of attribute values is preserved over the first splits. Moreover, $ordv_{1,j}$ is used as the coordinate of a point for polynomial construction.

In OPS4, random polynomial $P_{v_j}(x)$ should also be passed through a point $(x_1, ordv_{1,j})$ which guarantees that OPS4 preserves ordering over first splits.

Secret splitting in OPS4. First, x_1, x_k, and v_k are set up randomly from $\mathbb{F}_p$. For secret v_j, $ordv_{1,j}$ is chosen randomly from d_j. Then, a random polynomial $P_{v_j}(x)$ is built that passes through $(0, v_j)$, (x_k, v_k) and $(x_1, ordv1, j)$.

To this end, $k-3$ points $(a_i, b_i), i = 1, \dots, k-3$ are chosen randomly from $\mathbb{F}_p$, such that $a_i \neq x_k$, $a_i \neq x_1$ and $a_i \neq 0$ $\forall i = 1, \dots, k-3$.

Given the above $k-3$ points along $(x_1, ordv_{1,j})$, (x_k, v_k) and $(0, v_j)$ polynomial $P_{v_j}(x)$ of degree $k-1$ is built using 2.

To split v_j into $k-2$ splits (since (x_k, v_k) is already fixed and $(x_1, ordv_{1,j})$ is already set), a set $OPX = \{x_2, \dots, x_{k-1}\}$ of $k-2$ distinct elements in $\mathbb{F}_p$ is set, with $x_i \neq 0$, $x_i \neq x_1$ and $x_i \neq x_k$ $\forall i = 2, \dots, k-1$. Then, splits are $v_{i,j} = P_{v_j}(x_i)$.

Secret reconstruction in OPS4 is similar to S4. Note that $\mathcal{K} = (OPX, (x_k, v_k))$ together with x_1 are considered as a key and must be highly protected and kept hidden from the SP.

5. Analysis

In this section, we compare our schemes with the Paillier cryptosystem which is actually the most practical homomorphic encryption scheme, with respect to security and overhead.

In the proposed schemes, there is a trade-off between privacy and efficiency. In practical solutions, privacy requirements can be relaxed to provide more practical and efficient solutions [7]. Our schemes propose a classical trade-off with a lower level of security, but better storage and response time efficiency.

5.1. Security

We consider one security definition and two kinds of attacks based on ability and the knowledge of the SP or more generally an adversary may possess. We analyze the security of S4 against those definition and adversary's knowledge and abilities.

Database Knowledge attack (DBK): The adversary has only access to encrypted data stored at the SP's. This happens when the adversary hacks into the DBMS and gains access to disk-resident data.

Chosen Plaintext Attack (CPA): The adversary has access to a selected set of plaintexts and their associated ciphertexts. The adversary can gain information which reduces the security of encryption scheme.

Paillier is secure against both CPA and DBK attacks, i.e., Paillier's scheme is semantically secure. But, it is too expensive in terms of ciphertext storage space and query response time [8]. In our schemes, data privacy is preserved against DBK. In DBK attacks, an adversary has only access to splits stored at the SP's. More precisely, privacy in our secret splitting schemes, relies on the fact that a secret value is only retrievable by the user via private key $\mathcal{K}$. At least k splits and X are indeed necessary to reconstruct a secret, while the SP, or more generally an adversary with database knowledge, has access to only $k-1$ splits. Both X and the k^{th} split, i.e., $\mathcal{K}$, are stored at the user's.

The proposed secret splitting schemes are not secure against CPA. An adversary who has access to some plaintexts and their corresponding splits can use this information to infer more about data, which reduces the security of our schemes or even may reveal the secret key.

CPA is important in the context of public key cryptography, where the encryption key is public, hence the adversary can encrypt arbitrary plaintexts and see ciphertexts. CPA is not feasible in our schemes which are considered as a private key scheme, i.e., our schemes rely on private information for secret splitting/reconstructing [9]. Moreover, CPA is easy for public key cryptosystem, but it is much harder under the cloud data outsourcing, because the adversary typically does not access to the user's system and so it is difficult to find the association between plaintexts and ciphertexts.

However, regardless the feasibility of CPA, privacy is not guaranteed against CPA. An adversary who knows some plaintexts and their corresponding splits may infer about private information and even uses them to find the secret key.

5.2. Overhead

Our secret splitting schemes reduce storage costs comparing with Paillier's scheme. Paillier's scheme always produces ciphertexts of 2048 bits [10,11]. For example, for 32-bit integers, the storage overhead of Paillier is 64 times that of unencrypted data ($2048 = 32 \times 64$). In contrast S4 induces an overhead of $k-1$ times that of original data, because each integer is split into $k-1$ integers [12].

Since, choosing a large k increases storage cost while offering no more confidentiality, it is easy to set up k to allow a much lower storage overhead than Paillier [11].

In order to compute a summation query, Paillier's scheme requires multiplying the encryption of two values modulo a 2,048-bit public key [13]. Such modular multiplications are computationally expensive. The proposed schemes eliminate modular multiplication and thus significantly reduce computational overhead [14,11].

6. Conclusion

In this article, we presented a new privacy-preserving scheme, S4, for cloud outsourcing scenarios. The proposed scheme allows summation queries with the reduced cost comparing to the Paillier's scheme. In S4, computations are fully handled by the SP using a new secret-splitting scheme. Then, we extended S4 in a new way to propose an order-preserving homomorphic scheme to support more computations. The proposed schemes provide a trade-off for a fair lower level of privacy and more f lexibility in real-world scenarios. We plan to analyze the efficie ncy of our schemes by evaluating computation overhead, storage, and computing costs at the SP's. We also plan to strengthen the security of our schemes again st more powerful adversary model CPA.

7. References

[1] *Agrawal, Divyakant, Amr El Abbadi, Fatih Emekci, Ahmed Metwally, and Shiyuan Wang. "Secure data management service on cloud computing infrastructures."*

In New Frontiers in Information and Software as Services, pp. 57-80. Springer, Berlin, Heidelberg, 2011.

[2] *Popa, Raluca Ada, Catherine MS Redfield, Nickolai Zeldovich, and Hari Balakrishnan. "CryptDB: protecting confidentiality with encrypted query processing." In Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles, pp. 85-100. 2011.*

[3] *Paillier, Pascal. "Public-key cryptosystems based on composite degree residuosity classes." In International conference on the theory and applications of cryptographic techniques, pp. 223-238. Springer, Berlin, Heidelberg, 1999.*

[4] *Shamir, A. "How to Share a Secret Communications of the ACM, Vol. 22, n. 11." Nov 10 (1979): 359168-359176.*

[5] *Wong, Wai Kit, Ben Kao, David Wai Lok Cheung, Rongbin Li, and Siu Ming Yiu. "Secure query processing with data interoperability in a cloud database environment." In Proceedings of the 2014 ACM SIGMOD international conference on Management of data, pp. 1395-1406. 2014.*

[6] *Hadavi, Mohammad Ali, Rasool Jalili, Ernesto Damiani, and Stelvio Cimato. "Security and searchability in secret sharing-based data outsourcing." International Journal of Information Security 14, no. 6 (2015): 513-529.*

[7] *Dautrich, Jonathan L., and Chinya V. Ravishankar. "Security limitations of using secret sharing for data outsourcing." In IFIP Annual Conference on Data and Applications Security and Privacy, pp. 145-160. Springer, Berlin, Heidelberg, 2012.*

[8] *Kaaniche, Nesrine. "Cloud data storage security based on cryptographic mechanisms." PhD diss., 2014.*

[9] *Liu, Dongxi, and Shenlu Wang. "Nonlinear order preserving index for encrypted database query in service cloud environments." Concurrency and Computation: Practice and Experience 25, no. 13 (2013): 1967-1984.*

[10] *Moghadam, Somayeh Sobati, and Amjad Fayoumi. "Toward Securing Cloud-Based Data Analytics: A Discussion on Current Solutions and Open Issues." IEEE Access 7 (2019): 45632-45650.*

[11] *Naveed, Muhammad, Seny Kamara, and Charles V. Wright. "Inference attacks on property-preserving encrypted databases." In Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, pp. 644-655. 2015.*

[12] *Attasena, Varunya, Nouria Harbi, and Jérôme Darmont. "fvss: A new secure and cost-efficient scheme for cloud data warehouses." In Proceedings of the 17th International Workshop on Data Warehousing and OLAP, pp. 81-90. 2014.*

[13] *Omote, Kazumasa, and Tran Phuong Thao. "Sw-sss: Slepian-wolf coding-based secret sharing scheme." In Computational Intelligence in Security for Information Systems Conference, pp. 347-365. Springer, Cham, 2015.*

[14] *Moghadam, Somayeh Sobati, Jérôme Darmont, and Gérald Gavin. "Enforcing privacy in cloud databases." In International Conference on Big Data Analytics and Knowledge Discovery, pp. 53-73. Springer, Cham, 2017.*

Somayeh Sobati M. is an assistant professor in IT within the Electrical and Computer Engineering Faculty at Hakim Sabzevari University. She is also the head of security research laboratory at Hakim Sabzevari University. Dr. Sobati was awarded her Ph.D. degree in the information technology area from Lyon2 University, France in 2017. Her current research includes privacy, security issues in cloud computing, decision support systems,

data mining, IoT, and business information systems. Dr. Sobati is currently a member of a number of professional associations and she has taken roles in reviewing and editing research papers for several international conferences and journals.