Secure Order-Preserving Indexing Schemes for Outsourced Data

Somayeh Sobati Moghadam, Gérald Gavin, Jérôme Darmont

A Secure Order-Preserving Indexing Scheme for Outsourced Data

Somayeh Sobati Moghadam*, Gérald Gavin†, Jérôme Darmont‡
*Univ Lyon, Lumière, ERIC EA 3083
Email: Somayeh.Sobati-moghadam@univ-lyon2.fr
†Univ Lyon, Claude Bernard, ERIC EA 3083
Email:gerald.gavin@univ-lyon1.fr
‡Univ Lyon, Lumière, ERIC EA 3083
Email:jerome.darmont@univ-lyon2.fr

***Abstract*—Cloud computing offers the opportunity of data outsourcing as well as data management. However, because of various privacy issues, confidential data must be encrypted before being outsourced to the cloud. But query processing over encrypted data without decrypting data is a very challenging task. Property-preserving encryption schemes allow encrypting data while still enabling efficient querying over encrypted data. The inherent merits of property-preserving encryption schemes make them very suitable and efficient for cloud data outsourcing. However, the security of such schemes is still a challenge because they are vulnerable to statistical attacks. We present a new order-preserving scheme for indexing encrypted data, as an alternative to property-preserving schemes, which hides data frequency to achieve a strictly stronger notion of security. The proposed indexing method is secure against statistical attacks. Hence, data cannot be recovered from indexes. Moreover, our scheme is still efficient for query processing.**

***Keywords*-Cloud computing, Data privacy, Order-preserving indexing scheme.**

I. Introduction

Nowadays, data outsourcing become casual with the advent of cloud computing. Cloud computing appeals to many organizations because of a wide variety of benefits such as cost savings, higher availability, scalability and effective disaster recovery rather than in-house operations [1]. One of the most notable cloud outsourcing services is database outsourcing (Database-as-a-Service or DBaaS), where individuals and organizations outsource data storage and management to a Cloud Service Provider (CSP) [2]. Such services allow storing data on a remote CSP and querying data on demand [3].

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

A straightforward solution to preserve privacy is encrypting data before outsourcing to the cloud. Encryption protects the exposure of sensitive data even if the server is compromised and ensures that an adversary will be unable to interpret data.

However, when data are encrypted, query processing is not trivial. To overcome this problem, several solutions have been proposed [4], [5], [6]. These approaches implement cryptographic techniques, e.g., order-preserving encryption or homomorphic encryption, which allow computations to be carried out on encrypted data. In the context of relational databases, state-of-the-art solutions use property-preserving encryption (PPE) schemes. PPE schemes enable processing query over encrypted data without decryption. For instance, order-preserving encryption (OPE) preserves the order of plaintext in ciphertexts. Deterministic (DET) schemes encrypt the same plaintext into identical ciphertexts, thus the equality property is preserved [4]. PPE schemes are undoubtedly efficient, enabling queries to be directly processed over encrypted data [7].

The term efficient means here that all computation can be processed in time logarithmic in the size of the database, in contrast to performing linear work on each query, which is prohibitively slow for large databases [8]. Any practical PPE scheme inherently leaks some information about underlying data and are vulnerable to statistical attacks [9]. In statistical attacks, an adversary possesses some prior knowledge about the plaintext domain and its frequency distribution to gain access to encrypted data. Statistical attacks do not impose any threat when the underlying data has a uniform distribution [10]. CryptDB [5] is the first practical system that uses PPE schemes to support a wide range of queries to be processed over encrypted data. To the best of our knowledge, other systems such as BigQuery demo [11], Always Encrypted [12], CipherBase [13] and Relational Cloud [14] use PPE schemes, too. As a result, they are vulnerable to statistical attacks. Naveed et al. demonstrate that a large fraction of the records from PPE encrypted columns can be decrypted by statistical attacks [9].

Thus, we aim at proposing a secure and feasible scheme that is practical on cloud outsourced databases. In this paper, we focus on the problem of performing range and exact match queries over encrypted databases. We propose an Order-Preserving Indexing (OPI) scheme to address the vulnerability of PPE schemes. Our scheme bears good performance and leads to minimal change for existing software. The proposed indexing scheme is robust against statistical attacks through uniforming the distribution of underlying data. We extend the proposed scheme to introduce a new $w$rap-$a$round OPI scheme. waOPI provides higher level of security while hiding the

978-1-5090-1072-1/16/$31.00

original order of plaintexts.

In our scheme, the user does not need to model data distribution. Our scheme can be used along any encryption scheme to deal with range and exact match queries over encrypted data. Our scheme prevents the CSP from performing statistical analysis, even with prior knowledge about data.

Note that our scheme does not hide information such as the number of data items. One solution for applications that demand the privacy of data item numbers is injecting dummy data items into small data sets to build equally sized data sets [7].

The remainder of this paper is organized as follows. Section II gives some preliminaries. The overview of our system is described in section III. Section IV details our indexing scheme. We enhance the security of our scheme in section V, which is followed in section VI by a discussion. Section VII reviews related works. Finally, section VIII provides final conclusions and directions for future work.

II. Preliminaries

A. Property Preserving Encryption

In the context of DBaaS, state-of-the-art solutions rely on PPE schemes. A PPE scheme leaks certain properties of plaintexts by default, which enables an untrusted CSP to compute over encrypted data. Order-preserving encryption (OPE) and deterministic encryption (DET) are two PPE schemes that are used to handle range and exact match queries over encrypted data.

DET allows equality checking by encrypting a plaintext into the same ciphertexts when using the same key k. Thus:

$$\forall x, y: \quad x = y \Leftrightarrow EncDET_k(x) = EncDET_k(y).$$

DET allows performing `SELECT` with equality predicates, equality `JOIN`, `GROUP BY`, `COUNT`, `DISTINCT` queries [15]. Blowfish and AES+ECB are examples of widely used DET encryption schemes.
DET is secure when there is no data redundancy. In contrast to random encryption schemes (a random scheme encrypts the same plaintext into different ciphertexts using the same key), DET is not robust against statistical attacks.

OPE is a deterministic encryption scheme that preserves the order of plaintexts in ciphertexts, i.e., for any key k,

$$if \quad x \leq y, \quad then \quad EncOPE_k(x) \leq EncOPE_k(y).$$

The CSP can perform range queries when given encrypted constants $EncOPE_k(c_1)$ and $EncOPE_k(c_2)$ corresponding to range $[c_1, c_2]$. Aggregation queries `MIN`, `MAX`, `ORDER BY` and `SORT` can also be computed directly over encrypted data.

OPE is a weaker encryption scheme than DET because it reveals order. This weak form of encryption may provide sufficient security for some applications, e.g., when the adversary does not possess any prior knowledge, while increasing query processing efficiently [16].

B. Statistical Attacks against PPE Schemes

The most common adversarial model in existing solutions is ciphertext-only. It is assumed that the adversary has access only to ciphertexts without other background information. In this context, PPE schemes are vulnerable to statistical attacks [7]. In a statistical attack, the adversary has prior knowledge, e.g., the adversary possesses some statistical information about underlying plaintexts. The adversary can use such knowledge to infer about data or even launch an attack. For example, the adversary can map the distribution of ciphertext and plaintext to find all ciphertexts with high frequency.

The major problem with PPE schemes is that they are deterministic, i.e., plaintext and ciphertext have the same distribution. To address this drawback, a one-to-many mapping can be used to flatten the distribution of plaintexts. In one-to-many mappings, one plaintext is mapped into many ciphertexts belonging to a fixed size interval. As a result, highly frequent plaintexts induce dense intervals (figure 1). Hence, the distribution of plaintexts can easily be estimated by differential attacks [17].

C. Secure Indexes over Encrypted Attributes

A common technique to speed up the execution of queries in databases is to use a pre-computed index. The purpose of secure indexes is to retrieve requested data without decryption. An index is built for a specific attribute that needs to be accessed to evaluate a query. In fact, the biggest difference between a plain index and a secure index is that a secure index must not expose anything about underlying plaintexts.

A secure index for ordered data should support exact match, range and aggregation queries `MAX` and `MIN`. A secure index is built on plaintexts before encryption. Then, the index is stored in an auxiliary attribute along encrypted values at the CSP's.

III. Overview of System

In this section, we briefly discuss the system model where our scheme is applied, our adversary model and our system goals.

A. Basic Model

In our setting, there are three entities.

- *The Data Owner (DO)* has a large amount of data to be outsourced in the cloud to eliminate database maintenance. The DO accesses its data remotely without revealing the content of its database. The DO has limited computation and storage resources.
- *The CSP* provides data storage services and computational resources dynamically.
- *Data Users (DU)* are authorized users. DUs can retrieve encrypted data and decrypt them.

Our scheme is implemented in two software layers: a *user layer* that resides at the DO's and a *server layer* that resides at the CSP's.
When a query is issued, the user layer translates it into a query that can be executed directly over the encrypted

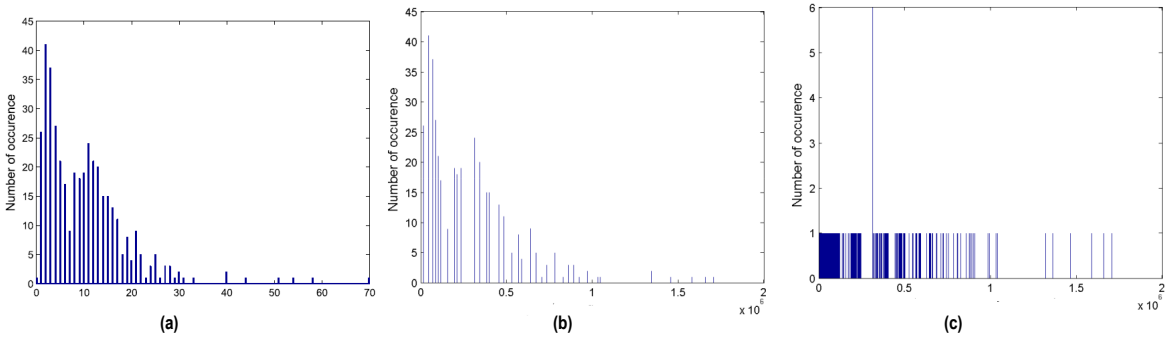

Figure 1. Comparison between plaintext and ciphertext distribution. (a) Plaintext distribution. (b) Ciphertext distribution of deterministic OPE. (c) Ciphertext distribution of one-to-many OPE [17].

database. When a query is received by the server, it is executed by the server layer. The computed result is shipped back to the client layer, which decrypts it (figure 2).

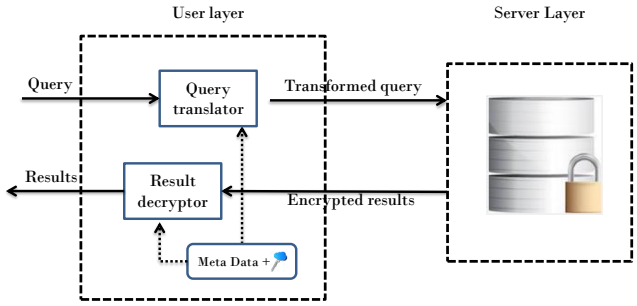

Figure 2. System model.

The user layer stores some metadata such as keys and database schemes to translate queries and decrypt query results.

For ease of presentation, we do not distinguish between the human user (DO or DU) and the user layer. When we say "user", we mean the human user who has access to the user layer and similarly, when we say "CSP", we mean the cloud service provider who has access to the server layer.

B. Security Threats

We assume the following threat model.

- The CSP is honest but curious (semi-honest) and has full access to encrypted data. We limit the attacks by the CSP to passive attacks. It is assumed that the CSP executes submitted queries honestly on encrypted data and sends complete and correct results. However, the CSP is curious and may attempt to increase its knowledge about confidential data.
- The CSP has some prior knowledge about outsourced data. For example, the CSP might possess some statistical information about data, such as the domain of plaintext values, minimum or maximum values, or some information about the frequency of values.
- We assume the user's machine is trusted. The communication channel between the user and the CSP uses a standard secure protocol such as SSL or IPsec.

C. System Goals

In order to preserve the privacy of outsourced data, we have the following goals:

- Data confidentiality: ensures that unauthorized parties and the CSP have no access to sensitive data.
- Index privacy: the proposed indexing method leaks no information about underlying plaintexts.
- Efficiency: the proposed method should be efficient for query processing.

IV. OPI

In our new order-preserving scheme, called OPI, the order of plaintext values is preserved in their corresponding ciphertexts, but not the frequency of values.

The intuition of our scheme is simple. First, we split the original plaintext's domain into successive intervals of equal lengths. Secondly, we extend the plaintext domain into a new domain. Finally, we use a mapping function to map the original values into the extended domain.

OPI satisfies two properties.

1) *Order-preserving:* Let D be the domain of plaintext values. The order of values must be preserved in their corresponding indexes, i.e., $\forall v, v' \in D : \quad if \quad v < v' \implies OPI(v) < OPI(v')$.
2) *Uniform distribution:* The distribution of plaintexts must be flattened in their corresponding indexes, i.e., two equal values have different indexes with a high probability.

The second property implies that the scheme is not deterministic and improves the robustness against statistical attacks.

We design our scheme for numerical values. Other data types should be translated into integers before indexing [18].

A. Notations

Consider a database consisting of a relation T. Suppose that T has one attribute, A. T stores N records. We denote by v_i the i^{th} value of attribute A. The goal here is storing the encryption of T, T', at the CSP's.

- Let D be the domain of plaintext values of A, consisting of t distinct values $\{v_1, \ldots, v_t\}$ with corresponding frequencies $\{f_1, \ldots, f_t\}$ where $v_1 < v_2 < \ldots < v_t$.
- Let D'=$[l', r')$ be an extension of D, the domain of indexes, where $l', r' \in \mathbb{Z}$. Without lose of generality, we consider $l' = 0$.
- Let $\mathcal{F}(i)$ be a function that outputs a random number in interval $d'_i = [l'_i, r'_i)$.

OPI consists of the following steps.

- **Splitting the domain of indexes:** The first step is splitting the domain of indexes D' into t intervals d'_i $(1 \leq i \leq t)$ [19] such that d'_i=$[l'_i, r'_i)$=$[(\sum_{k=1}^{i-1} f_k).\frac{|D'|}{N}, (\sum_{k=1}^{i} f_k).\frac{|D'|}{N})$.

  Such splitting has the following properties [7], [19]:
  1) $d'_i \subseteq D'$ $(1 \leq i \leq t)$
  2) $\bigcup_{i=1}^{t} d'_i = D'$
  3) $\forall d'_i, d'_j$ $i \neq j$: $d'_i \cap d'_j = \emptyset$.
- **Mapping:** Second, each plaintext value v_i $(1 \leq i \leq t)$ is mapped to an index I_i, where I_i=$\mathcal{F}(i)$. In other words, each time v_i is repeated, $\mathcal{F}(i)$ returns a random number in interval d'_i=$[l'_i, r'_i)$ as I_i.

B. Table Structure

To store relation T at the CSP's, column A is presented in two columns $Enc(A)$ and $SIdx(A)$ (Secure order-preserving Index) in encrypted table T'. The values of $Enc(A)$ are encrypted values of A with an additive homomorphic encryption scheme such as Paillier's cryptosystem [20]. Values I_i $i = 1, ..., N$ are stored in column $SIdx(A)$. When a range or exact match query is issued, it is rewritten and executed on column $SIdx(A)$. When an aggregation query is issued, it is directly computed over $Enc(A)$.

C. Query Processing

In this section, we discuss how to process queries.

Each query is transformed into a form that can be computed by the CSP. We classify queries into two main classes.

1) Range and Exact Match Queries:

- QUERY-I: "$A = v_i$"
  is an exact match query, e.g., SELECT * FROM T WHERE A=v$_i$. This query is translated into SELECT * FROM T' WHERE $SIdx(A) \geq l'_i$ AND $SIdx(A) < r'_i$, where l'_i and r'_i are the boundaries of interval d'_i.

- QUERY-II: "$A \leq v_i$"
  is a range query that is translated into SELECT * FROM T' WHERE $SIdx(A) < r'_i$.

- QUERY-III: "$A < v$"
  is a range query that is translated into SELECT * FROM T' WHERE $SIdx(A) < l'_i$.

- QUERY-IV: "$v_i \leq A \leq v_j$"
  is translated into SELECT * FROM T' WHERE $SIdx(A) \geq l'_i$ AND $SIdx(A) < r'_j$. l'_i and r'_j are the left and the right boundary of d'_i and d'_j, respectively.

2) Update Queries: In our scheme, the execution of update queries is straightforward. A deletion query such as DELETE FROM T WHERE $A = v_i$, is translated into DELETE FROM T' WHERE $SIdx(A) \geq l'_i$ AND $SIdx(A) < r'_i$.

To insert a new value v_i, the corresponding encryption and index, $Enc(v_i)$ and I$_i$ are computed and inserted into T'.

For an update query, tuples satisfying the query predicate are returned. Then, the values in the SET predicate are substituted with the new ones.

V. Enhancing Security

In this section, we introduce waOPI, a light modification of OPI that improves its security. waOPI reveals less information about underlying plaintexts without sacrificing efficiency. OPI Actually reveals the location of plaintext. waOPI is not order-preserving per se, but still allows range and exact match queries to be processed. This modification is simple and generic: we wrap-around the domain of indexes.

A. waOPI: wrap-around OPI

First, a random value, wa, is chosen randomly from D' such that $l' < wa < r'$. Then, in waOPI the values of wrap-around index waI, waI_i $i = 1, ..., N$, are defined based on algorithm1.

Algorithm 1 Calculating waI

Input: wa
Output: waI_i, $i = 1, ..., N$
  for $i = 1$ to N **do**
    if $(i > wa)$ **then**
      $waI_{i-wa} \leftarrow I_i$
    else
      $waI_{i+N-wa} \leftarrow I_i$
    end if
    return waI_i
  end for

B. Query Processing

waOPI processes range and exact match queries in the same way as OPI. The only difference is that when a query is issued, it must be transformed into a form that can be executed over wrap-around indexes.

Considering a query that asks all data in interval $[v_i, v_j)$. This interval is transformed into $[l'_i, l'_j)$, as described in

section IV-C1. Then, $[l'_i, l'_j)$ is transformed into $[c_i, c_j)$ where:

- if $l'_i \leq l'_j \leq wa$ then

$$[c_i, c_j) = [waI_{l'_i+N-wa}, waI_{l'_j+N-wa})$$

- if $wa \leq l'_i \leq l'_j$ then

$$[c_i, c_j) = [waI_{l'_i-wa}, waI_{l'_j-wa})$$

- if $l'_i \leq wa < l'_j$ then

$$[c_i, c_j) = [waI_{N-wa+l'_i}, waI_N] \cup (0, waI_{l'_j-wa}).$$

VI. Security Discussion

In OPI, we uniformly distribute original values across a domain into another broader domain. Having a uniform distribution of values minimizes information leakage to an adversary who would observe the outsourced data. Overhead comes from the new domain size, which is bigger than the original domain's. Theoretically, we need domain D' to be of size $|D'| \gg |D|$ to uniformly distribute all original values across D'. Choosing an appropriate domain size for D' is a trade-off between security and storage cost [7], [19].

waOPI provides a substantial improvement in security. Given the wrap-around indexes, the adversary cannot deduce any information about the real location of underlying plaintexts.

However, our schemes are possibly vulnerable when executing queries. The adversary or the CSP is able to infer the frequency of attribute values by observing frequent intervals accessed at query time. The adversary might also be able to infer random value wa. Yet, it has been proven that no scheme is secure against an adversary that has access to both plaintexts and prior knowledge about query workloads [21], [22]. To prevent such inference, Obvious RAM (ORAM) [23], [24], [25], [26] can be used. ORAM hides access patterns during query processing, but ORAM techniques are currently computationally prohibitive and cannot be used in practice. Another solution is sending fake queries. Fake queries must be frequently submitted. The goal is misleading the adversary to infer a wrong wa. Submitting fake queries makes waOPI more robust.

VII. Related Works

OPE was introduced in the database community by Agrawal et al. [27] as a tool to support efficient range queries over encrypted data [22]. This scheme maps each value of the plaintext domain to one value in the ciphertext domain, such that the relative ordering of plaintexts is preserved after encryption. This scheme bears weak privacy protection because the CSP can statistically estimate the original data values [7]. Boldyreva et al. define the first formal ideal-security definition of OPE, called indistinguishability under ordered chosen-plaintext attack (IND-OCPA) [8]. Informally, an IND-OCPA scheme reveals no additional information about the underlying plaintexts beside their order, which is the minimum requirement of the order-preserving property. IND-OCPA [8] introduces a random mapping that preserves order, but leaks at least half of the plaintext bits (i.e., more information than OPE) and shows poor efficiency [28].

Popa et al. introduce the first practical IND-OCPA scheme, mutable order-preserving encoding (mOPE) [28]. mOPE requires an interactive protocol for query processing. Additionally, mOPE relies on user defined functions (UDFs) for query processing, which makes it unsuitable for cloud outsourcing.

Liu et al. introduce an OPE scheme that randomly splits the original plaintext domain into successive intervals with different lengths [7]. Then, an extended ciphertext domain is selected and split into the same number of intervals. Finally, nonlinear mapping functions map the original plaintexts into ciphertexts in the extended domain.

While Liu et al.'s scheme is efficient for query processing, it partially destroys the distribution of original data because the splitting method is random and does not rely on data frequency.

Table I shows the comparison between existing OPE schemes. It highlights that mOPE has the lowest performance because it requires the user to interact with the server to retrieve requested data. Our schemes do not require such interaction and present the same efficiency as Liu et al.'s.

In IND-OCPA approach, only the order of data is revealed, while our scheme reveals nothing about the original data. Compared with Liu et al.'s scheme, waOPI can achieve security against statistical attacks, because we use a density and frequency-aware domain splitting method.

Table I
Comparison of OPE schemes

Scheme	Efficiency level	Security level	Order comparison
Agrawal	Medium	Low	Direct
Boldyreva	Low	Medium	Direct
mOPE	Low	High	UDFs
Liu	High	Medium	Direct
OPI	High	Medium	Direct
waOPI	High	High	Direct

VIII. Conclusion

In this paper, we propose an order-preserving indexing scheme, OPI, in which the ordering relation between plaintexts is preserved in their corresponding indexes. In OPI, the distribution of index values is different from the plaintext distribution. We extend our scheme in order to achieve robustness against statistical attacks. In waOPI, the security of plaintexts is indeed improved by hiding the rule of data distribution and data frequency, without sacrificing efficiency.

Nevertheless, we should prove the security of OPI and waOPI. We also plan to investigate secure indexing mechanisms with additive homomorphic property to efficiently support aggregation queries over encrypted data.

REFERENCES

[1] E. Damiani, S. D. C. di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati, "Key management for multi-user encrypted databases," in *Proceedings of the 2005 ACM Workshop On Storage Security And Survivability, USA, November 11*, 2005, pp. 74–83. [Online]. Available: http://doi.acm.org/10.1145/1103780.1103792

[2] L. Xiong, S. Chitti, and L. Liu, "Preserving data privacy in outsourcing data aggregation services," *ACM ACM Transactions on Internet Technology (TOIT)*, vol. 7, no. 3, 2007. [Online]. Available: http://doi.acm.org/10.1145/1275505.1275510

[3] G. Amanatidis, A. Boldyreva, and A. O'Neill, "Provably-secure schemes for basic query support in outsourced databases," in *21st Annual IFIP WG 11.3 Working Conference on Data and Applications Security, USA,*, 2007, pp. 14–30. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-73538-0_2

[4] M. Bellare, A. Boldyreva, and A. O'Neill, "Deterministic and efficiently searchable encryption," in *Advances in Cryptology - CRYPTO, 27th Annual International Cryptology Conference, USA, August 19-23*, 2007, pp. 535–552. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-74143-5_30

[5] R. A. Popa, C. M. S. Redfield, N. Zeldovich, and H. Balakrishnan, "Cryptdb: protecting confidentiality with encrypted query processing," in *Proceedings of the 23rd ACM Symposium on Operating Systems Principles 2011, SOSP 2011, Cascais, Portugal, October 23-26, 2011*, 2011, pp. 85–100. [Online]. Available: http://doi.acm.org/10.1145/2043556.2043566

[6] S. Tu, M. F. Kaashoek, S. Madden, and N. Zeldovich, "Processing analytical queries over encrypted data," *Proceedings of the VLDB Endowment, PVLDB*, vol. 6, no. 5, pp. 289–300, 2013. [Online]. Available: http://www.vldb.org/pvldb/vol6/p289-tu.pdf

[7] Z. Liu, X. Chen, J. Yang, C. Jia, and I. You, "New order preserving encryption model for outsourced databases in cloud environments," *J. Network and Computer Applications*, vol. 59, pp. 198–207, 2016. [Online]. Available: http://dx.doi.org/10.1016/j.jnca.2014.07.001

[8] A. Boldyreva, N. Chenette, Y. Lee, and A. Oneill, "Order-preserving symmetric encryption," *Advances in Cryptology-EUROCRYPT*, pp. 224–241, 2009.

[9] M. Naveed, S. Kamara, and C. V. Wright, "Inference attacks on property-preserving encrypted databases," in *SIGSAC*, 2015, pp. 644–655. [Online]. Available: http://doi.acm.org/10.1145/2810103.2813651

[10] T. Sanamrad, L. Braun, D. Kossmann, and R. Venkatesan, "Randomly partitioned encryption for cloud databases," in *DBSec*, 2014. [Online]. Available: http://dx.doi.org/10.1007/978-3-662-43936-4_20

[11] "Google encrytped big query," https://github.com/google/encrypted-bigquery-client, (accessed 2016).

[12] "Always encrypted," https://msdn.microsoft.com/enus/library/mt163865(v=sql.130).aspx., (accessed 2016).

[13] A. Arasu, K. Eguro, M. Joglekar, R. Kaushik, D. Kossmann, and R. Ramamurthy, "Transaction processing on confidential data using cipherbase," in *31st IEEE International Conference on Data Engineering, ICDE 2015, South Korea, April 13-17, 2015*, 2015, pp. 435–446. [Online]. Available: http://dx.doi.org/10.1109/ICDE.2015.7113304

[14] C. Curino, E. P. C. Jones, R. A. Popa, N. Malviya, E. Wu, S. Madden, H. Balakrishnan, and N. Zeldovich, "Relational cloud: a database service for the cloud," in *CIDR 2011, Fifth Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 9-12, 2011, Online Proceedings*, 2011, pp. 235–240. [Online]. Available: http://www.cidrdb.org/cidr2011/Papers/CIDR11_Paper33.pdf

[15] R. A. Popa, "Building practical systems that compute on encrypted data," Ph.D. dissertation, Massachusetts inistitute of technology, 2014.

[16] G. Özsoyoglu, D. A. Singer, and S. S. Chung, "Anti-tamper databases: Querying encrypted databases." in *DBSec*, 2003, pp. 133–146.

[17] K. Li, W. Zhang, C. Yang, and N. Yu, "Security analysis on one-to-many order preserving encryption-based cloud data search," *IEEE Trans. Information Forensics and Security*, vol. 10, no. 9, pp. 1918–1926, 2015. [Online]. Available: http://dx.doi.org/10.1109/TIFS.2015.2435697

[18] D. Liu and S. Wang, "Programmable order-preserving secure index for encrypted database query," in *2012 IEEE 5th International Conference on Cloud Computing (CLOUD),*. IEEE, 2012, pp. 502–509.

[19] M. A. H. R. Jalili, "Secure data outsourcing based on threshold secret sharing: Towards a more practical solution," in *VLDB PhD Workshop*, 2010, pp. 54–59.

[20] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Advances in cryptology EURO-CRYPT'99*. Springer, 1999, pp. 223–238.

[21] W. K. Wong, B. Kao, D. W. Cheung, R. Li, and S. Yiu, "Secure query processing with data interoperability in a cloud database environment," in *International Conference on Management of Data, SIGMOD 2014, USA, June 22-27, 2014*, 2014, pp. 1395–1406. [Online]. Available: http://doi.acm.org/10.1145/2588555.2588572

[22] A. Boldyreva, N. Chenette, and A. O'Neill, "Order-preserving encryption revisited: Improved security analysis and alternative solutions," in *Advances in Cryptology CRYPTO, 31st Annual Cryptology Conference, USA, August 14-18*, 2011, pp. 578–595. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-22792-9_33

[23] O. Goldreich and R. Ostrovsky, "Software protection and simulation on oblivious rams," *J. ACM*, vol. 43, no. 3, pp. 431–473, 1996. [Online]. Available: http://doi.acm.org/10.1145/233551.233553

[24] E. Shi, T. H. Chan, E. Stefanov, and M. Li, "Oblivious RAM with o((logn)3) worst-case cost," in *Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings*, 2011, pp. 197–214. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-25385-0_11

[25] E. Stefanov and E. Shi, "Oblivistore: High performance oblivious cloud storage," in *2013 IEEE Symposium on Security and Privacy, SP 2013, Berkeley, CA, USA, May 19-22, 2013*, 2013, pp. 253–267. [Online]. Available: http://dx.doi.org/10.1109/SP.2013.25

[26] E. Stefanov, M. van Dijk, E. Shi, C. W. Fletcher, L. Ren, X. Yu, and S. Devadas, "Path ORAM: an extremely simple oblivious RAM protocol," in *2013 ACM SIGSAC Conference on Computer and Communications Security, CCS'13, Berlin, Germany*, 2013, pp. 299–310. [Online]. Available: http://doi.acm.org/10.1145/2508859.2516660

[27] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, "Order-preserving encryption for numeric data," in *Proceedings of the ACM SIGMOD International Conference on Management of Data, Paris, France, June 13-18, 2004*, 2004, pp. 563–574. [Online]. Available: http://doi.acm.org/10.1145/1007568.1007632

[28] R. A. Popa, F. H. Li, and N. Zeldovich, "An ideal-security protocol for order-preserving encoding," in *IEEE Symposium on Security and Privacy (SP)*. IEEE, 2013, pp. 463–477.