Secure Range Query Processing over Untrustworthy Cloud Services

Theodoros Tzouramanis
Department of Information and Communication Systems Engineering
University of the Aegean – Samos, 83200
Greece
ttzouram@aegean.gr

ABSTRACT

Database-as-a-service is a relatively new cloud computing service offered on a pay-per-use basis and providing on-demand access to a database. The way data has dramatically increased in volume explains its success, while security and privacy issues arise, leaving enterprises, in particular, exposed to the risk of leakage of the data which they entrust to specialized cloud service providers or to other parties in order to reduce storage and query processing costs. Since traditional encryption does not support the execution of queries on encrypted data, this paper focuses on the problem of secure computation on encrypted data and puts forward a cloud database model that supports secure range query processing and retrieval of multi-dimensional (i.e. multi-attribute) data. It proposes two schemes to resist practical attacks operating on the basis of powerful background knowledge. A performance and efficiency evaluation of these schemes is also carried out to confirm their efficiency and practicability.

CCS CONCEPTS

• Information systems~Data encryption • Security and privacy~Privacy-preserving protocols • Security and privacy~Management and querying of encrypted data • Theory of computation~Database query processing and optimization (theory) • *Theory of computation~Cryptographic protocols*

KEYWORDS

Data outsourcing, cloud database services, secure cloud computing, confidential multi-dimensional data retrieval, encryption.

ACM Reference format:

T. Tzouramanis. 2017. Secure Range Query Processing over Untrustworthy Cloud Services. In *Proceedings of the 20th ACM International Database Engineering & Applications Symposium, Bristol UK, July 2017 (IDEAS 2017)*, 10 pages.
DOI: 10.1145/3105831.3105872

IDEAS '17, July 12 - 14 2017, Bristol, United Kingdom.

ACM ISBN 978-1-4503-5220-8/17/07…$15.00
http://dx.doi.org/10.1145/3105831.3105872

1 INTRODUCTION

Cloud computing is a model which enables enterprises and individuals to benefit from ubiquitous, convenient and on-demand access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal effort in terms of management or service provider interaction. Among the services on offer is the Database-as-a-Service (DaaS). It is understood that the DaaS is related to traditional database services with no need to manage or control the underlying infrastructure, nor even take on some of the core database administration responsibilities. While there are clear benefits, some aspects of cloud technology are also marked by a lack of efficiency, a major example of which is the issue of the protection of the privacy of the data. Before cloud technology gained in popularity, the risks involved would have made it unthinkable for any enterprise or individual to surrender all their sensitive data to a third-party entity.

The most important threats to privacy in the cloud lie within two broad categories which are the "lack of control" over the data and the "absence of transparency", since there tends to be insufficient information available regarding the cloud service processing operation itself. These risks may expose the data to a security breach, possibly the most prominent issue in the domain of data outsourcing [1], the impact of which may be devastating for the owner of the data [2].

A straightforward solution to reduce the risk of a breach of data is encryption. And yet, recent market surveys [3, 4] reveal that almost a third of all highly sensitive corporate data stored in the cloud is not encrypted. This low percentage is partly due to the increased communication and computational cost involved in executing the client's queries. For this reason, a new class of encryption models has been proposed [5] offering protection for sensitive outsourced data in untrustworthy clouds while still supporting business workflows efficiently. Models and schemes have been proposed that either suggest a keyword search [6] or the execution of several kinds of well-known database queries such as the nearest-neighbor [7] or the top-k [8] queries.

This work focuses on the secure and efficient execution of range queries over multi-dimensional (i.e. multi-attribute) datasets in the cloud. The range query is defined as finding all the data with keys within a certain range in one or more attributes. It can be used either as a standalone query (e.g. in similarity search

applications) or as a core module of common data analytic tasks such as machine learning and data mining (e.g. classification and clustering). The first work of research to propose cryptographic techniques for evaluating range predicates directly over encrypted data are [9] and [10]. For example, Shi et al. in [10] propose a searchable encryption scheme that supports multi-dimensional range queries by utilizing an interval tree structure to form a hierarchical representation of intervals along every dimension. Since then several solutions [11 – 20] have appeared, some of which offer very strong security guarantees but not efficient performance. Other solutions offer more of a balance between confidentiality and efficiency.

This study focuses on encrypted data that are not supported by any specialized indexing method to compute the range query. A real-life application could be a cloud service providing storage and a processing environment in which an indexing mechanism may not be an option, or it may be offered at a disadvantageous cost. The paper suggests three schemes and identifies trade-offs between security and efficiency, which revolve around trading storage or processing overhead and potentially false positives for security. With the proposed schemes, a cloud server is able to correctly verify whether a data object is inside the boundary of a range in the encrypted data domain without breaching the privacy of the data or of the users' queries (i.e. preferences).

The paper will unfold with Section 2, which briefly surveys the advantages and limitations of some of the work that is closely related to range searching over encrypted data; Section 3, which discusses preliminaries and notations that are relevant to the proposed work; Section 4, which puts forward three schemes for range query processing in encrypted multi-dimensional databases; Section 5, which examines experimental results on the performance efficiency of both real and synthetic datasets; Section 6, in which conclusions are drawn and suggestions for future research are made.

2 RELATED WORK

The range query being one of the most popular query operations in SQL and in multi-dimensional databases, models based on four distinct methodologies were developed to offer secure range query processing in cloud databases: the hidden vector encryption (HVE), bucketization, order-preserving encryption (OPE), and special indexing methods traversal.

The HVE-based approaches [9, 11], which use asymmetric cryptography to encrypt the data by hiding its attributes in an encrypted vector, use bilinear groups equipped with bilinear maps and at a considerably expensive computational cost.

In response to this limitation, bucketization offered a more desirable balance between security and practical efficiency by grouping the multi-dimensional objects with spatial proximity into the same encrypted bucket: the range query retrieves all the objects in the buckets that overlap the range [12, 13], and this is possibly achieved by introducing false positives. Bucketization provides weak privacy protection since it discloses the distribution of the data to an observer because two encrypted datasets, with the same number of data items but different distributions, will cause the buckets to have different distributions in sizes as these two datasets will balance the number of items among buckets differently.

Similarly, OPE [14, 15] offers solutions that preserve the relative ordering of the data items even after encryption, which allow the direct translation of a range query from the original to the encoded domain, thus the trivial support of rectangular range search. By design, OPE inevitably leaks the ordering of the encoded data, allowing the cloud provider to statistically estimate the actual values of both the data items and the queries.

Some specialized indexing methods can offer the range query evaluation over encrypted data. For example [16] proposes a searchable encryption scheme that supports range query processing by utilizing an interval tree structure to form a hierarchical representation of intervals for every dimension. The method stores multiple ciphertexts corresponding to a single data value in the server, i.e. every one corresponds to a range. Elsewhere, [17, 18] make use of lightweight cryptography and follow the notion of searchable symmetric encryption to support the range query by utilizing fast inverted indices. Both methods leak the search pattern (i.e. reveal which queries are the same). Furthermore, the approach proposed by [17] cannot support updates. In other recent research, the authors of [19] propose a secure hardware-based construction of the popular single-dimensional B^+-tree for the support of exact match and range queries. However, the extension of this secure access method for handling multi-dimensional data is yet to be explored.

The index-based solution in [20], which introduces a hierarchical encrypted variation of the R-tree [21, 22], encrypts the query ranges along a methodology similar to this paper's, while the data objects themselves can be encrypted in any other way. To avoid revealing the data proximity in every query, the method returns all the data stored in the data nodes overlapping a query range, thus by possibly introducing false positives. The main difference is that this paper proposes a solution that neither relies on a pre-computed index nor needs computational and storage power to pre-process and index the dataset, and that it avoids limitations such as the well-known curse of dimensionality in the case of the R-tree.

Privacy-preserving range queries can be supported with optimal security *via* powerful theoretical cryptographic tools such as the Oblivious Random Access Memory (ORAM, [23]), which enables access to an encrypted memory space without disclosing which memory location is accessed, thus hiding both the data and the access patterns of the queries, or such as the fully homomorphic encryption [24]. Both these tools are prohibitively costly for database applications.

This paper extends the secure computation model presented in [7] for nearest neighbor search, in order to also handle the range query, while shielding it with stronger security guaranties. The main difference between nearest neighbor and range search is that nearest neighbor search pre-defines the number of effective search results (i.e. to 1 or k) in the generation of each query without providing a particular range on the data space; while range search specifically defines an interval of values in every dimension of the

data space without considering the number of effective search results. Therefore, nearest neighbor search is different from range search, even in the plaintext domain and the model proposed in [7] is not applicable to range search.

3 PRELIMINARIES AND NOTATIONS

3.1 Design Goals

The proposed model is based on the DaaS cloud services model in which two entities form the two parts of the system: the client, which is the legitimate owner of the data, and the cloud server. The protocol considers the participation of one client, although more clients may exist. The client outsources the data to the server in an encrypted form with the expectation that s/he will be able to carry out a remote search without breach of privacy. It is assumed that the client is capable of protecting the secret key for the data decryption process. The client's device to access the data is assumed to have a minimum degree of computation power to be ableto execute the encryption and decryption processes or to perform simple calculations in order to refine the results of the queries. This paper does not consider issues of control of access and accountability or threats to data integrity and availability: other mechanisms can handle these.

On the basis of the above requirements, the design of the proposed models on the cloud should achieve the following main security and performance goals.

- *Confidential data storage and retrieval:* A goal of the model is to prevent the cloud provider or any outsider from obtaining access to the plaintext of the protected content or from obtaining any amount of useful information about it, besides what can be derived from the legitimate client's encrypted search results.
- *Client authorization*: The authorized client alone should be able to contribute data or obtain access to the plaintext of the relevant encrypted content.
- *Scalability and data upload/search efficiency*: The system should aim at high scalability, i.e. low key management overhead; the client should not be obliged to store permanently any plaintext or ciphertext of the data content in her/his local device. The system should support efficient encrypted range search functionality. These goals should be achieved with a low communication and computation overhead.
- *Simplicity and extensibility*: The protocol should be simple enough to be implementable on top of existing commercial cloud APIs.
- *Secure communication*: Communication between the client and the server should be secure and it should happen without the need for the intervention of an intermediary entity such as a trustworthy authority.

3.2 The Threats Model

The predator is assumed to be honest-but-curious, with the intention of obtaining full access to the plaintext of the encrypted stored data without altering any data that is communicated between the client and the server. The predator might know all the procedures involved, such as the encryption and decryption algorithms. The predator should not be able to get access to any part of the plaintext database. Besides having access to the encrypted data, the predator might possess additional knowledge about the original data. The attacks can be classified according to the different amounts of knowledge that the predator might possess [7].

- *Level* 1 - *Known ciphertext sample attack*: the simplest class of attacks in which the predator may have access to a sample or even to the whole set of the encrypted data.
- *Level* 2 - *Known ciphertext and plaintext samples attack*: Apart from the ciphertext, in this class of attacks the predator may be aware of the values of a sample of tuples of the plaintext, without having any knowledge the corresponding ciphertext of these tuples in the encrypted dataset.
- *Level* 3 - *Known link between a ciphertext and a plaintext sample attack (or known input-output attack)*: The predator may now be aware of which ciphertext tuples in the encrypted dataset correspond to which tuples of the known plaintext sample. The predator may accordingly be aware of the plaintext and ciphertext of a sample of the client's queries.

It is now clear that a higher-level attack is more powerful than a lower-level attack, so if an encryption model is secure against an attack of a higher-level, it follows that it will be secure against a lower-level attack as well.

3.3 Notations

The study considers every data tuple as a multi-dimensional point by modeling its attributes as dimensions and their values as their coordinates. Therefore, the data point is of the form p (p_1, p_2, ..., p_d) where d is the number of data dimensions. The data points construct a dataset P which is encrypted and stored on the cloud. The range query is defined with a hyper-rectangle which is provided by its d-dimensional centre point q (q_1, q_2, …, q_d) and

Table 1: Symbols and notations

Symbol	Definition
d	The number of data dimensions
P	A set of d-dimensional point objects
P'	The encrypted version of P
p, r	d-dimensional data points
p (p_1, p_2, …, p_d)	A d-dimensional point p with coordinates p_1, p_2, …, and p_d
p' (p'_1, p'_2, …, p'_d)	The encrypted version of a point p (p_1, p_2, …, p_d)
$E_p(p)$	The encrypted version of a point p using the encryption function E_p
q (q_1, q_2, …, q_d)	The d-dimensional centre-point of the hyper-rectangular predicate of a range query
l_1, l_2, …, l_d	The length in every dimension of the hyper-rectangular predicate of a range query

the length l_1, l_2, …, l_d of the hyper-rectangle's side in every dimension. Table 1 lists the most commonly used symbols in the paper.

4 The Proposed Encryption Schemes

4.1 The Basic ASPE Scheme

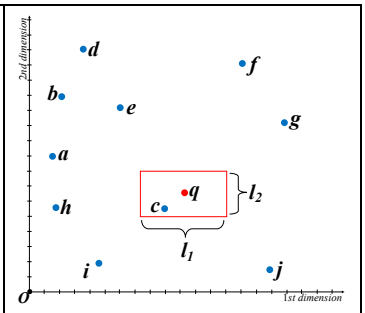
Given a d-dimensional point p (p_1, p_2, …, p_d) and a hyper rectangle H with a center point q (q_1, q_2, …, q_d) and a length side in every dimension l_1, l_2, …, l_d (an example in two dimensions is illustrated in Fig. 1), the point p lies within H if all the following d distance comparison operations truly hold:

$$\begin{pmatrix} d(q_1,p_1) \le l_1/2 \\ d(q_2,p_2) \le l_2/2 \\ \dots \\ d(q_d,p_d) \le l_d/2 \end{pmatrix} \Leftrightarrow \begin{pmatrix} \sqrt{{q_1}^2 - 2q_1p_1 + {p_1}^2} \le l_1/2 \\ \sqrt{{q_2}^2 - 2q_2p_2 + {p_2}^2} \le l_2/2 \\ \dots \\ \sqrt{{q_d}^2 - 2q_dp_d + {p_d}^2} \le l_d/2 \end{pmatrix} \Leftrightarrow$$

$$\Leftrightarrow \begin{pmatrix} {q_1}^2 - 2q_1p_1 + {p_1}^2 - {l_1}^2/4 \le 0 \\ {q_2}^2 - 2q_2p_2 + {p_2}^2 - {l_2}^2/4 \le 0 \\ \dots \\ {q_d}^2 - 2q_dp_d + {p_d}^2 - {l_d}^2/4 \le 0 \end{pmatrix}$$

object	coordinates	
p	p_1	p_2
a	1.5	9
b	2	13
c	9	5.5
d	4	17
e	6	12
f	14	15
g	17	11
h	2	5.5
i	4.5	2
j	16	1.5

(a)



(b)

Figure 1: A 2-dimensional sample dataset and the range query defined by a rectangle with a center-point q (q_1, q_2) and length l_1, l_2 for each of its sides.

The above inequalities are decomposed into a number of product computations, from which $\forall\, i \in \{1, ..., d\}$ the products ${p_i}^2$ and ${q_i}^2$ are fixed and, thus, can be pre-computed and be given to the cloud upon the insertion of p and, accordingly, upon the creation of q, so that they will be available for the secure range query processing. The only products that need to be computed by the server during the query processing are the ones between the coordinates of every database point p and the corresponding coordinates on the same dimensions of the query point q. By representing p and q by column vectors, the products between the coordinates of p and q can be computed by a scalar product between p and q in the form $p^T * q$, where p^T is the transpose of p. Therefore, the proposed encryption scheme needs to preserve only this type of scalar products for the range query processing, and as [7] has proved the scheme will be then resistant to level-2 attacks. Therefore the scheme has to be built on top of the notion of *asymmetric scalar-product-preserving encryption* as it has been defined in [7].

Definition 1 - Asymmetric scalar-product-preserving encryption (ASPE) scheme: Let E_p / E_q be the function for encrypting data / query points, K_p / K_q be the corresponding secret key and $E_p(p, K_p)$ / $E_q(q, K_q)$ be the encrypted version of a data / query point p / q. The encryption scheme is an ASPE scheme if, and only if, it preserves only the scalar product between p and q, i.e., $p^T * q = E_p(p, K_p) * E_q(q, K_q)$.

The query point q should be encrypted differently from the data points (i.e. by using a different encryption function E_q) to guarantee that the encrypted version of q should not be equal to the encrypted version of any point r in the database when q coincides with r. In such a case, the encryption process would preserve the scalar product between p and q, thus the scalar product between p and r which is not a desirable property because it can be proved (see Theorem 2 in [7]) that it reveals the distance between p and r.

The scalar product $p^T * q$ between p and q can be written as $p^T * I_d * q$, in which I_d is the $d \times d$ identity matrix. The I_d matrix can be decomposed into $M * M^{-1}$ for any invertible matrix M, i.e. $p^T * q = p^T * I_d * q = p^T * (M * M^{-1}) * q = (p^T * M) * (M^{-1} * q)$. If we set $p' = E_p(p, K_p) = M^T * p$ and $q' = E_q(q, K_q) = M^{-1} * q$, where p' and q' are the encrypted versions of p and q respectively, then $p^T * q = p^T * M * M^{-1} * q = p'^T * q'$, i.e. the scalar product between any database point p and the query point q is preserved. Additionally it is not possible for someone to determine the values of p and q using the values of p' and q' without knowing M. Also supposing that p' and r' are the encrypted versions of two data points p and r, then $p'^T * r' = p^T * M * M^T * r$, which is not equal to $p^T * r$ in general. Therefore, the encryption scheme does not preserve the scalar product between two data points, or between a data point and itself, while it indeed preserves the scalar product between a data point and a query point. This analysis shows that the ASPE scheme can be implemented by using M and M^{-1} as the encryption keys for the data points and the queries, respectively.

The products ${p_1}^2$, ${p_2}^2$, …, ${p_d}^2$ of the coordinates of the data point p can be computed by the scalar product $p^T * p$. Wong et al. [7] have proved that the preservation of this product will reveal to the predator that p lies on a hyper-sphere that is centered at the origin of the space with a radius $\sqrt{{p_1}^2 + {p_2}^2 + \dots + {p_d}^2}$. Although the exact location of p will be unknown, the information revealed partially compromises security. In order to keep this information hidden, the idea in our ASPE scheme is to transform the $d \times 1$ column vector $p = [p_1, p_2, \dots, p_d]^T$ into the corresponding $3d \times 1$ column vector $\hat{p} = [s_{p1}{p_1}^2, -2s_{p1}p_1, s_{p1}, s_{p2}{p_2}^2, -2s_{p2}p_2, s_{p2}, \dots, s_{pd}{p_d}^2, -2s_{pd}p_d, s_{pd}]^T$, in which the pre-computed products ${p_1}^2$, ${p_2}^2$, …, ${p_d}^2$ are also included. The s_{p1}, s_{p2}, …, s_{pd} parameters are random positive numbers which are used to increase the number of unknown parameters in the possible system of linear equations that a predator might construct and, as will be shown later, their existence does not affect the correctness of the range query computation. The transformed data point $\hat{p}$ is then encrypted using the proposed function E_p for encrypting the data.

For a similar reason, the $d \times 1$ column vector query point $q = [q_1, q_2, \ldots, q_d]^T$ is transformed into the $3d \times d$ matrix:

$$\hat{q} = \begin{bmatrix} s_{q1} & & 0 \\ s_{q1}q_1 & \cdots & 0 \\ s_{q1}({q_1}^2 - {l_1}^2/4) & & 0 \\ 0 & & \vdots \\ \vdots & \ddots & 0 \\ 0 & & s_{qd} \\ 0 & \cdots & s_{qd}q_d \\ 0 & & s_{qd}({q_d}^2 - {l_d}^2/4) \end{bmatrix}, \quad (1)$$

in which the pre-computed products ${q_1}^2$, ${q_2}^2$, …, ${q_d}^2$ and ${l_1}^2$, ${l_2}^2$, …, ${l_d}^2$ are spread and hidden in the $\hat{q}$ vector using the corresponding s_{q1}, …, s_{qd} random positive numbers. The transformed query point $\hat{q}$ is then encrypted using the proposed function E_q for encrypting the queries.

Fig. 2 summarizes the proposed Basic ASPE Scheme process.

***Theorem* 1.** Suppose p' is the encrypted version of the d-dimensional point p ($p_1, p_2, \ldots, p_d$) and q' is the encrypted version of the d-dimensional query point q ($q_1, q_2, \ldots, q_d$) that is the center-point of a hyper-rectangle H with a length side in every dimension $l_1, l_2, \ldots, l_d$. The Basic ASPE Scheme correctly determines whether p is inside the hyper-rectangular area H by evaluating ${p'}^T * q' \leq 0$.

Private Key: a $3d \times 3d$ invertible matrix M.

Data encryption function: $E_p(p) = M^T * \hat{p}$, where p ($p_1, p_2, \ldots, p_d$) is a d-dimensional data point, $\hat{p}$ is a $3d$-dimensional data vector of the form $\hat{p} = [s_{p1}{p_1}^2, -2s_{p1}p_1, s_{p1}, \ldots, s_{pd}{p_d}^2, -2s_{pd}p_d, s_{pd}]^T$ and $s_{p1}, s_{p2}, \ldots, s_{pd}$ are random positive numbers.

Query encryption function: $E_q(q) = M^{-1} * \hat{q}$, where q ($q_1, q_2, \ldots, q_d$) is a d-dimensional query point and $\hat{q}$ is a $3d \times d$ matrix that is defined as in Equation (1).

Range enclosure operation: assuming $p' = E_p(p)$ and $q' = E_q(q)$, in order to determine whether p lies within a hyper-rectangle with centre-point q and length side in every dimension $l_1, l_2, \ldots, l_d$, the server needs to check whether ${p'}^T * q' \leq 0$.

Data decryption function: assuming an encrypted point p', a preliminary step of the decryption process is to extract the random positive numbers $s_{p1}, s_{p2}, \ldots, s_{pd}$ using the function $\pi * (M^T)^{-1} * p'$, where π is a $d \times 3d$ binary matrix in which $\forall\, i \in \{1, d\}$ and $\forall\, j \in \{1, 3d\}$ if $j = 3i$ then π_{ij} is set to be equal to 1, otherwise π_{ij} is set to be equal to 0. The coordinates of the data point p are then decrypted using the decryption function $E_p^{-1}(p') = \sigma * (M^T)^{-1} * p'$ where σ is a $d \times 3d$ matrix in which $\forall\, i \in \{1, d\}$ and $\forall\, j \in \{1, 3d\}$ if $j = 3i - 1$ then σ_{ij} is set to be equal to $\frac{-1}{2s_{pi}}$, otherwise σ_{ij} is set to be equal to 0.

Figure 2: The Basic ASPE Scheme.

Proof: Starting with ${p'}^T * q'$ we get $(M^T * \hat{p})^T * M^{-1} * \hat{q} = \hat{p}^T * M * M^{-1} * \hat{q} = \hat{p}^T * \hat{q} = [s_{q1}s_{p1}{p_1}^2 - 2s_{q1}s_{p1}q_1p_1 + s_{q1}s_{p1}({q_1}^2 - {l_1}^2/4), \ldots, s_{qd}s_{pd}{p_d}^2 - 2s_{qd}s_{pd}q_dp_d + s_{qd}s_{pd}({q_d}^2 - {l_d}^2/4)] = [s_{q1}s_{p1}(d^2(q_1, p_1) - {l_1}^2/4), \ldots, s_{qd}s_{pd}(d^2(q_d, p_d) - {l_d}^2/4)]$

Therefore, the evaluation ${p'}^T * q' \leq 0$ is equivalent to the evaluation:

$$\begin{pmatrix} s_{q1}s_{p1}(d^2(q_1,p_1) - {l_1}^2/4) \leq 0 \\ s_{q2}s_{p2}(d^2(q_2,p_2) - {l_2}^2/4) \leq 0 \\ \ldots \\ s_{qd}s_{pd}(d^2(q_d,p_d) - {l_d}^2/4) \leq 0 \end{pmatrix} \Leftrightarrow \begin{pmatrix} d(q_1,p_1) \leq l_1/2 \\ d(q_2,p_2) \leq l_2/2 \\ \ldots \\ d(q_d,p_d) \leq l_d/2 \end{pmatrix} \square$$

With regard to attacks, the following theorem holds.

Theorem 2: The Basic ASPE Scheme is not secure against level-3 attacks.

Proof: Assuming that the predator knows both the plaintext p and the ciphertext p' of a data point (the case of the knowledge of a plaintext of a query point q and its corresponding encrypted version q' is similar), the predator can use the equality $p' = M^T * \hat{p}$ in order to construct a set of $3d$ equations, in which s/he will have $3d \times 3d = 9d^2$ unknown parameters in the encryption key M^T and d unknown parameters in the vector $\hat{p}$ (i.e. the random positive numbers $s_{p1}, s_{p2}, \ldots, s_{pd}$).

By knowing, in total, n pairs of plaintext data points and their corresponding ciphertexts, the predator can construct a set of $n3d$ equations with $9d^2 + nd$ unknown parameters. It is thus clear that if $n3d \geq 9d^2 + nd \Rightarrow n \geq 9d/2$, then the number of equations will be larger than the number of unknown parameters. Therefore the predator will be able to solve the system of linear equations to eventually find the secret key M and the plaintext of every ciphertext. □

4.2 The Enhanced Scheme 1

To make the Basic ASPE Scheme secure against level-3 attacks, this section discusses a solution inspired from an analogous scheme introduced in [7] for the k-nearest neighbor query. The new solution suggests the random splitting of all the values in every $3d \times 1$ column vector $\hat{p}$, in order to generate two random shares $\widehat{pa}$ and $\widehat{pb}$, such that $\forall\, i \in \{1, \ldots, 3d\}$ we will have $\hat{p}_i = \widehat{pa}_i + \widehat{pb}_i$. Therefore the product $\hat{p}^T * \hat{q}$ will be equal to $\widehat{pa}^T * \hat{q} + \widehat{pb}^T * \hat{q}$. The vector $\widehat{pa}$ will be then encrypted with the secret key Ma and the vector $\widehat{pb}$ with the secret key Mb. Additionally every query point q will be encrypted twice, using the matrices Ma^{-1} and Mb^{-1}, respectively. The final range enclosure evaluation will be performed using the equation ${pa'}^T * qa' + {pb'}^T * qb' = (Ma^T * \widehat{pa})^T * Ma^{-1} * \hat{q} + (Mb^T * \widehat{pb})^T * Mb^{-1} * \hat{q} = \widehat{pa}^T * \hat{q} + \widehat{pb}^T * \hat{q} = \hat{p}^T * \hat{q}$.

Instead of splitting the values in the column vector $\hat{p}$ the method can alternatively split the values in the matrix $\hat{q}$, or the values in some of the rows of $\hat{p}$ and the values in some other rows of $\hat{q}$ (however, the method cannot split the values in the same

rows in both $\hat{p}$ and $\hat{q}$ at the same time). Therefore, during the encryption phase, the client will need to store a configuration bit string S, which is a $3d$-bits vector, with every entry in it indicating whether p-splitting or q-splitting is used for the corresponding row in $\hat{p}$ and $\hat{q}$. Since there are 2^{3d} possible configurations, the Enhanced ASPE Scheme 1 is more secure against attacks as compared to the Basic ASPE Scheme.

Private Key: two $3d' \times 3d'$ invertible matrices Ma and Mb, a configuration $3d'$-bits string S and $3d' - 3d$ pre-generated random numbers $w_{3d+1}, w_{3d+2}, \ldots, w_{3d'}$.

Data encryption function: $E_p(p) = Ma^T * \widehat{pa} + Mb^T * \widehat{pb}$, where p $(p_1, p_2, \ldots, p_d)$ is a d-dimensional data point and $\hat{p} = \widehat{pa} + \widehat{pb}$ is a $3d'$-dimensional vector in which the first $3d$ dimensions are the same as the corresponding vector $\hat{p}$ in the Basic ASPE Scheme in Fig. 2 and for the remainder $i = 3d + 1$ to $i = 3d'$ dimensions if $S_i = 1$ then $\hat{p}_i = w_i$, otherwise $\hat{p}_i$ is set to be equal to a random number. For the last dimension with which $S_i = 0$, $\hat{p}_i$ is given a value so that the scalar product over the artificial attributes $3d + 1$ to $3d'$ is 0 (see [7] for more details). Additionally, for $i = 1$ to $i = 3d'$, if $S_i = 1$, then the value of $\hat{p}_i$ is randomly split into $\widehat{pa}_i$ and $\widehat{pb}_i$. If $S_i = 0$ then $\widehat{pa}_i$ and $\widehat{pb}_i$ are both set to be equal to $\hat{p}_i$.

Query encryption function: $E_q(q) = Ma^{-1} * \widehat{qa} + Mb^{-1} * \widehat{qb}$, where q $(q_1, q_2, \ldots, q_d)$ is a d-dimensional query point and $\hat{q}$ is a $3d' \times d'$ matrix, in which the first $3d \times d$ cells are defined as in Equation (1) and for the rest $i = 3d + 1$ to $i = 3d'$ of the rows of the $j = 1$ to $j = d$ left-most columns, if $S_i = 0$ then $\hat{q}_{ij} = w_i$, otherwise $\hat{q}_{ij}$ is set to be equal to a random number. For the last dimension with which $S_i = 1$, $\hat{q}_{ij}$ is given a value so that the scalar product over the artificial values $i = 3d + 1$ to $i = 3d'$ is 0 (see [7] for more details). All the other cells in the matrix are set to be equal to a random number. Additionally, $\forall\, j \in \{1, d\}$ for $i = 1$ to $i = 3d'$, if $S_i = 0$ then the value of $\hat{q}_{ij}$ is randomly split into $\widehat{qa}_{ij}$ and $\widehat{qb}_{ij}$. If $S_i = 1$ then $\widehat{qa}_{ij}$ and $\widehat{qb}_{ij}$ are both set to be equal to $\hat{q}_{ij}$.

Range enclosure operation: assuming $p' = E_p(p) = pa' + pb'$ and $q' = E_q(q) = qa' + qb'$, in order to determine whether p lies within a hyper-rectangle H with centre-point q and a length side in every dimension $l_1, l_2, \ldots, l_d$, the server needs to check whether $pa'^T * qa' + pb'^T * qb' \leq 0$.

Data decryption function: assuming an encrypted point $p' = pa' + pb'$, a preliminary step of the decryption process is to extract the random positive numbers $s_{p1}, s_{p2}, \ldots, s_{pd}$ using the function $\pi * (Ma^T)^{-1} * pa' + \pi * (Mb^T)^{-1} * pb'$, where π is a $d \times 3d'$ binary matrix in which $\forall\, i \in \{1, d\}$ and $\forall\, j \in \{1, 3d\}$ if $j = 3i$ then π_{ij} is set to be equal to 1, otherwise π_{ij} is set to be equal to 0. The coordinates of the d-dimensional data point p are then decrypted using the decryption function $E^{-1}{}_p(p') = \sigma * (Ma^T)^{-1} * pa' + \sigma * (Mb^T)^{-1} * pb'$ where σ is a $d \times 3d'$ matrix in which $\forall\, i \in \{1, d\}$ and $\forall\, j \in \{1, 3d'\}$ if $j = 3i - 1$ then σ_{ij} is set to be equal to $\frac{-1}{2s_{pi}}$, otherwise σ_{ij} is set to be equal to 0.

Figure 3: The Enhanced ASPE Scheme 1.

To boost security further, the solution suggests the increase of the d number of dimensions by adding artificial attributes to both $\hat{p}$ and $\hat{q}$. This can be achieved by extending the $3d$ column vector $\hat{p}$ to a $3d'$ column vector, and the $3d \times d$ matrix $\hat{q}$ to a $3d' \times d'$ matrix, by padding artificial values such that the scalar product over the added attribute values will be 0.

Fig. 3 summarizes the procedures implementing the proposed Enhanced ASPE Scheme 1.

Theorem 3: The Enhanced ASPE Scheme 1 is secure against level-3 attacks.

Proof: Assuming that $d' = d$, i.e. that no artificial dimensions have been added (the addition of artificial attributes will only increase the security of the scheme) and assuming that the predator knows both the plaintext p and the parts pa' and pb' of the ciphertext p' of a data point (the case of the knowledge about a plaintext of a query point q and its corresponding encrypted version q' is similar), the predator can use the equalities $pa' = Ma^T * \widehat{pa}$ and $pb' = Mb^T * \widehat{pb}$ in order to construct a set of $2*3d$ equations, in which s/he will have $2(3d*3d) = 18d^2$ unknown parameters in the encryption keys Ma^T and Mb^T, $3d$ unknown parameters in the vector $\widehat{pa}$ (since s/he does not know how this vector has been created, i.e. s/he does not know the configuration bit string S) and $3d$ unknown parameters in the vector $\widehat{pb}$ (for the same reason as in the case of $\widehat{pa}$).

By knowing in total n pairs of plaintext data points and their corresponding ciphertexts, the predator can construct a set of $n6d$ equations with $18d^2 + n6d$ unknown parameters. It is thus evident that independently of the value of n there would always be $18d^2$ more unknown parameters than there are equations. Therefore, the predator would not be able to solve the system of linear equations in order to find the secret keys Ma and Mb. Thus the scheme can guarantee protection against level-3 attacks. □

4.3 The Enhanced Scheme 2

This section proposes a secure solution against level-3 attacks that does not need to split the data and query points nor to add artificial dimensions, both of which actions may burden the processing cost for executing the range query. As the proof in Theorem 2 has shown, the drawback of the Basic ASPE Scheme is that if the predator knows the coordinates of a data point p $(p_1, p_2, \ldots, p_d)$, then the $3d$-dimensional vector $\hat{p}$ that is defined in Fig. 2 has only d unknown parameters, i.e. the random positive numbers $s_{p1}, s_{p2}, \ldots, s_{pd}$. Accordingly, if the predator knows the coordinates of the centre-point q $(q_1, q_2, \ldots, q_d)$ and the length $l_1, l_2, \ldots, l_d$ in every dimension of the hyper-rectangular predicate of a range query, then the $3d \times d$ matrix $\hat{q}$ that is defined in Equation (2) has only d unknown parameters, i.e. the random positive numbers $s_{q1}, s_{q2}, \ldots, s_{qd}$.

To remediate to the above drawback, the Enhanced ASPE Scheme 2 suggests that the vector $\hat{p}$ should be transformed into a $3d \times d$ matrix as follows:

$$\hat{p} = \begin{bmatrix} s_{p1}(p_1{}^2 - m_{11}) & & -s_{pd}m_{1d} \\ -s_{p1}(2p_1 + m_{21}) & \cdots & -s_{pd}m_{2d} \\ s_{p1}m_{31} & & -s_{pd}m_{3d} \\ \vdots & \ddots & \vdots \\ -s_{p1}m_{[3d-2]1} & & s_{pd}(p_d{}^2 - m_{[3d-2]d}) \\ -s_{p1}m_{[3d-1]1} & \cdots & -s_{pd}(2p_d + m_{[3d-1]d}) \\ -s_{p1}m_{[3d]1} & & s_{pd}m_{[3d]d} \end{bmatrix}, \quad (2)$$

in which the parameters s_{p1}, s_{p2}, …, s_{pd} are random positive numbers and $\forall\, i \in \{1, 3d\}$ and $\forall\, j \in \{1, d\}$ and the parameter m_{ij} is a random positive number, with the exception of two cases: (a) if $i = 3j$ then $m_{ij} = 1$ and (b) if $i = 3j - 1$ then m_{ij} is a pseudo-random positive number, which is calculated as a function of the encryption key M and s_{pi}, for example $m_{ij} = \theta * \min(Hash(M_{ij}), Hash(s_{pi})) / \max(Hash(M_{ij}), Hash(s_{pi}))$, where θ is a user-predefined positive number and *Hash* is a cryptographically secure hash function.

Also the $3d \times d$ matrix $\hat{q}$ should be defined as follows:

$$\hat{q} = \begin{bmatrix} s_{q1}e_{11} & & s_{qd}e_{1d} \\ s_{q1}(q_1 + e_{21}) & \cdots & s_{qd}e_{2d} \\ s_{q1}(q_1{}^2 - l_1{}^2/4 - e_{31}) & & s_{qd}e_{3d} \\ \vdots & \ddots & \vdots \\ s_{q1}e_{[3d-2]1} & & s_{qd}e_{[3d-2]d} \\ s_{q1}e_{[3d-1]1} & \cdots & s_{qd}(q_d + e_{[3d-1]d}) \\ s_{q1}e_{[3d]1} & & s_{qd}(q_d{}^2 - l_d{}^2/4 - e_{[3d]d}) \end{bmatrix}, \quad (3)$$

in which the parameters s_{q1}, s_{q2}, …, s_{qd} are random positive numbers and $\forall\, i \in \{1, 3d\}$ and $\forall\, j \in \{1, d\}$, and the parameter e_{ij} is a random positive number as well, with the exception of the case when $i = 3j - 2$ in which $e_{ij} = 1$.

Therefore, the range enclosure operation is $p'^{T} * q' \leq 0 \Leftrightarrow \hat{p}^{T} * \hat{q} \leq 0$, where:

$$p'^{T} * q' \leq 0 \Leftrightarrow \hat{p}^{T} * \hat{q} \leq 0 \Leftrightarrow \begin{pmatrix} a_{11} \leq 0 & \cdots & a_{1d} \leq 0 \\ \vdots & \ddots & \vdots \\ a_{d1} \leq 0 & \cdots & a_{dd} \leq 0 \end{pmatrix}$$

in which $\forall\, i \in \{1, d\}$ the cell a_{ii} represents the equation:

$$a_{ii} = s_{qi}s_{pi}\left(q_i{}^2 - 2q_ip_i + p_i{}^2 - \frac{l_i{}^2}{4}\right) -$$

$$-s_{qi}s_{pi}\left(m_{[3i-1]i}q_i + 2e_{[3i-1]i}p_i + \sum_{j=1}^{3d} m_{ji}e_{ji}\right) =$$

$$= s_{qi}s_{pi}(d^2(q_i, p_i) - \frac{l_i{}^2}{4} - a_i)$$

in which:

$$a_i = m_{[3i-1]i}q_i + 2e_{[3i-1]i}p_i + \sum_{j=1}^{3d} m_{ji}e_{ji}$$

and, without loss of generality, by assuming that all the coordinate values p_1, p_2, …, p_d and q_1, q_2, …, q_d are non-negative numbers, then evidently $\forall\, i \in \{1, d\}$ we have $a_i \geq 0$. By evaluating the inequality $a_{ii} \leq 0$ we then have:

$$s_{qi}s_{pi}(d^2(q_i, p_i) - \frac{l_i{}^2}{4} - a_i) \leq 0 \Leftrightarrow$$

$$\Leftrightarrow d^2(q_i, p_i) \leq \frac{l_i{}^2}{4} + a_i \quad (4)$$

Equation (4) is equivalent to the equation $d^2(q_i, p_i) \leq \frac{l_i'^2}{4}$, where $l_i'^2 = l_i{}^2 + 4a_i$ is the length of the side in the i-th dimension of a hyper-rectangle H' which encloses the range query hyper-rectangle H. Therefore, the Enhanced Scheme 2 in the range enclosure operation evaluates every encrypted data point against a hyper-rectangle that encloses the one which has been defined by the client. Thus, the execution of the client's query may introduce false positives, which can subsequently be discarded by the client after s/he receives the results from the server. However, the me-

Private Key: a $3d \times 3d$ invertible matrix M.

Data encryption function: $E_p(p) = M^{T} * \hat{p}$, where p (p_1, p_2, …, p_d) is a d-dimensional data point, $\hat{p}$ is a $3d \times d$ matrix that is defined as in Equation (2).

Query encryption function: $E_q(q) = M^{-1} * \hat{q}$, where q (q_1, q_2, …, q_d) is a d-dimensional query point and $\hat{q}$ is a $3d \times d$ matrix that is defined as in Equation (3).

Range enclosure operation: assuming $p' = E_p(p)$ and $q' = E_q(q)$, in order to determine whether p lies within a hyper-rectangle with centre-point q and length side in every dimension l_1, l_2, …, l_d, the server needs to check whether $p'^{T} * q' \leq 0$. The result might contain some false positives that will be discarded by the client after receiving and decrypting the results.

Data decryption function: assuming an encrypted point p', a preliminary step of the decryption process is to extract the random positive numbers s_{p1}, s_{p2}, …, s_{pd} using the function $\pi * (M^{T})^{-1} * p'$ where π is a $d \times 3d$ binary matrix in which $\forall\, i \in \{1, d\}$ and $\forall\, j \in \{1, 3d\}$ if $j = 3i$ then π_{ij} is set to be equal to 1, otherwise π_{ij} is set to be equal to 0. The coordinates of the data point p are then decrypted using the decryption function $E^{-1}{}_p(p') = \sigma * (M^{T})^{-1} * p'$ where σ is a $d \times 3d$ matrix in which $\forall\, i \in \{1, d\}$ and $\forall\, j \in \{1, 3d\}$ if $j = 3i - 1$ then σ_{ij} is set to be equal to $\frac{-1}{2s_{pi}}$, otherwise σ_{ij} is set to be equal to 0. It should be kept in mind that if $i = 3j - 1$ then m_{ij} is a pseudo-random positive number which is calculated as a function of the encryption key M and s_{pi}, for example $m_{ij} = \theta * \min(Hash(M_{ij}), Hash(s_{pi})) / \max(Hash(M_{ij}), Hash(s_{pi}))$, where θ is a user-predefined positive number and *Hash* is a cryptographically secure hash function.

Figure 4: The Enhanced ASPE Scheme 2.

thod does not provide false negatives, i.e. all the data points in the requested hyper-rectangle by the client are included in the results set. Fig. 4 summarizes the procedures that build the proposed Enhanced ASPE Scheme 2.

Theorem 4: The Enhanced ASPE Scheme 2 is secure against level-3 attacks.

Proof: Assuming that the predator knows both the plaintext p and the ciphertext p' of a data point (the case of knowledge about a plaintext of a query point q and its corresponding encrypted version q' is similar), the predator can use the equality $p' = M^T * \hat{p}$ in order to construct a set of $3d$ equations, in which s/he will have $3d * 3d = 9d^2$ unknown parameters in the encryption key M^T and $3d^2$ unknown parameters in the matrix $\hat{p}$ (i.e. $\forall\ i \in \{1, d\}$ the parameters s_{pi} and $\forall\ i \in \{1, 3d\}$ and $\forall\ j \in \{1, d\}$ with $i \neq 3j$ the parameters m_{ij}).

By having knowledge of, in total, n pairs of plaintext data points and their corresponding ciphertexts, the predator can construct a set of $n3d$ equations with $9d^2 + n3d^2$ unknown parameters. It is thus clear that, independently of the value of n, there would always be $9d^2 + n3d(d - 1)$ more unknown parameters than there are equations. Therefore the predator would not be able to solve the system of linear equations in order to find the secret key M. Thus the scheme can guarantee protection against level-3 attacks. □

5 EXPERIMENTAL EVALUATION

This section presents an experimental performance evaluation of a prototype implementation of the three proposed ASPE schemes in Java. The workstation that was used for simulating the client-side was equipped with Intel Core i5 CPU running at 2.70GHz with 8GB RAM and Microsoft Windows 7 Professional 64-bit OS. With regard to the cloud service, its implementation is based on the Okeanos Infrastructure as a Service [25] using a Dual Core virtual machine with 6 GB RAM running Microsoft Windows Server 2012 and Microsoft SQL Server 2012. The experiments were conducted using two datasets, the 'Statlog (Shuttle)' dataset from the UCI repository [26], which is a real-life dataset dealing with the positioning of radiators in the space shuttle counting 58,000 9-dimensional points, as well as a synthetic dataset containing an equal number of 58,000 uniformly distributed 100-dimensional points.

Every experiment was repeated 10 times and the average value of the measured parameters was calculated. When measuring the time cost performance for processing the range query on the server-side, a different randomly located range was chosen in the repetition of the experiment. In the case of the Enchanced Scheme 1 the number of artificial dimensions $d' - d$ was set to be equal to twice the d number of dimensions of the actual data points, therefore $d' = 3d$. If the findings of the performance investigation of the proposed ASPE schemes are comparable, irrespectively of whether the real or the synthetic data are used, then only half of them (*i.e.* either with the real or with the synthetic data) are depicted.

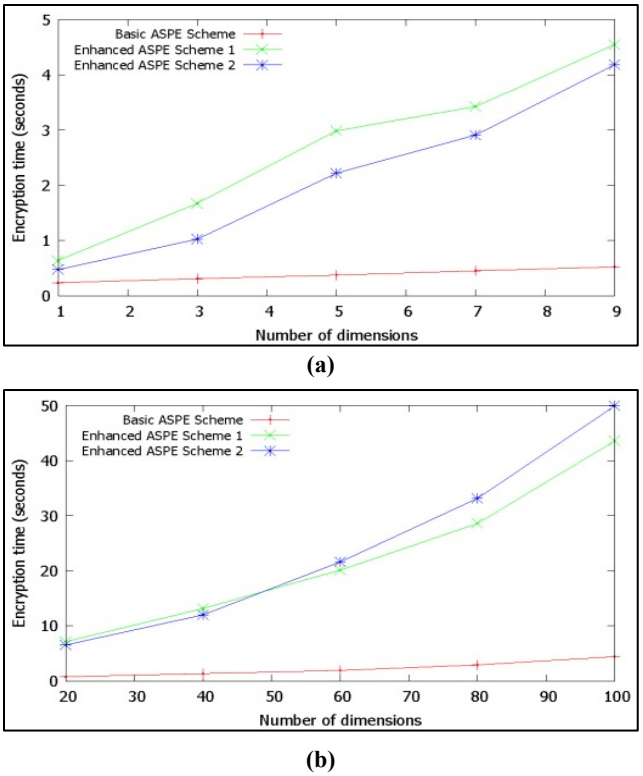
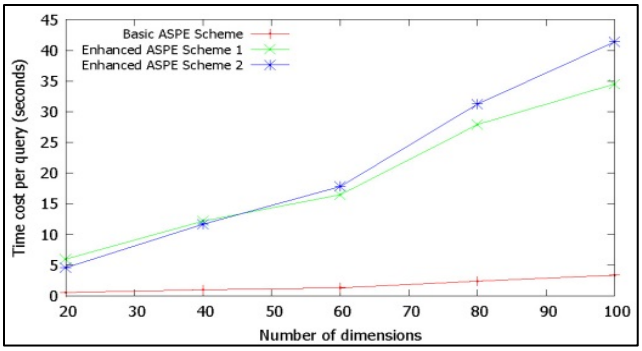


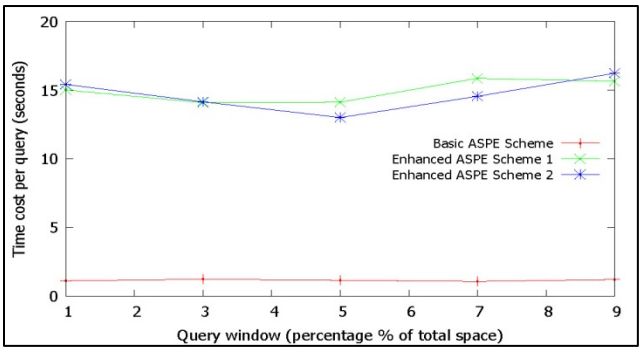
Figure 5: Data encryption: The impact of the number of dimensions on the data encryption time using (a) real data and (b) synthetic data.

The graphs of the first experiment in Fig. 5(a) / Fig. 5(b) study the impact of the d number of data dimensions on the Statlog / synthetic dataset encryption time for all the proposed ASPE schemes. The graphs show an expected growth of the time cost as the number of data dimensions increases. The two figures also indicate that the performance of the Enchanced Scheme 1 can be better than the corresponding performance of the Enchanced Scheme 2 only above dimensions $d = 50$, while in lower dimensions of roughly $d = 3$ to 7 its performance rates about 20% to 30% worse than the performance of the Enchanced Scheme 2. Fig. 5(b) also shows that the performace of the enchanced schemes indicates a sharper increase on the encryption time in a rather high number of dimensions.

The next two graphs in Fig. 6 illustrate the execution time of the range query on the server-side for the synthetic dataset. The results in Fig. 6(a) show an expected growth of the time cost in relation to the growth of the number of data dimensions (in this experiment the query hyper-rectangle covers an area equal to 5% of the workspace). The results in Fig. 6(b) show that the processing time cost is not affected by the query window size (defined as a percentage of the area of the rectangular workspace) since, in every query evaluation the proposed schemes access all the points in the dataset. In this experiment d is considered to be equal to 50.

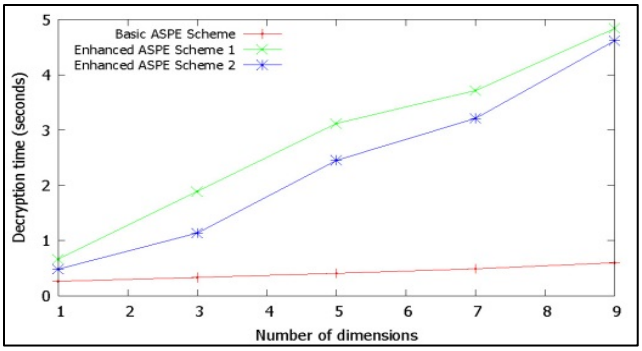


(a)

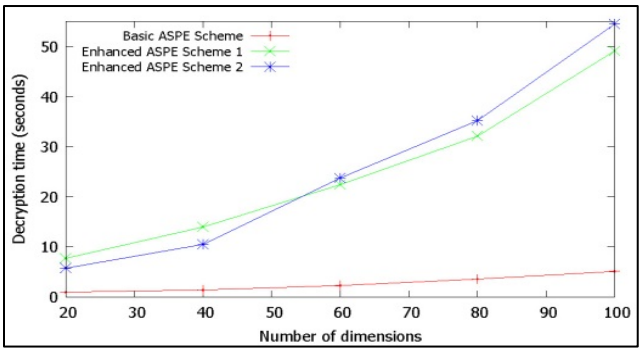


(b)

Figure 6: Range query: (a) The impact of the number of dimensions, and (b) the impact of the query window size, in both cases, on the query execution time, using the synthetic data.



(a)



(b)

Figure 7: Data decryption: The impact of the number of dimensions on the data decryption time using (a) real data and (b) synthetic data.

The graphs of the third experiment in Fig. 7(a) / Fig. 7(b) illustrate the impact of the d number of dimensions on the Statlog / synthetic dataset decryption time for all the proposed schemes. The graphs indicate that the decryption time cost is clearly higher that the corresponding encryption time cost for all the schemes, which is because the decryption function performs slightly more operations per data tuple. As expected, the performance behavior of the three ASPE schemes that is shown in Fig. 7 is analogous to the corresponding behavior in Fig. 5 for the encryption time cost.

Finally the graph relating to the last experiment in Fig. 8 studies the false positives rate for the Enhanced ASPE Scheme 2 as a function of the values of the random positive numbers m_{ij} and e_{ij}, $\forall\, i \in \{1, 3d\}$ and $\forall\, j \in \{1, d\}$ in the definition of the $\hat{p}$ and $\hat{q}$ parameters in Equations (2) and (3). It should be remembered that the Enhanced ASPE Scheme 2 is the only scheme introducing false positives in the query response. The underlying dataset is the synthetic dataset and the query window covers an area that is equal to 1% of the workspace. The x-axis in the graph shows four different domain values from which the random parameters draw their values, as a ratio to the corresponding coordinate value of the data or the query point, respectively, on the same dimension, i.e. for uniformly distributed random values that are equal to or smaller than 0.1%, 1%, 5% and 10% of the coordinate values of the data or of the query point on the same dimension. The figure shows that, as the values of the random parameters increase, the number of false positives increases as well.

False positives increase privacy, therefore a rather large number of false positives is not necessarily a drawback for a secure query processing scheme. The conclusions drawn from this experiment are that the owner of the data can tune the Enhanced Scheme 2 for different tradeoffs between security and false positives.

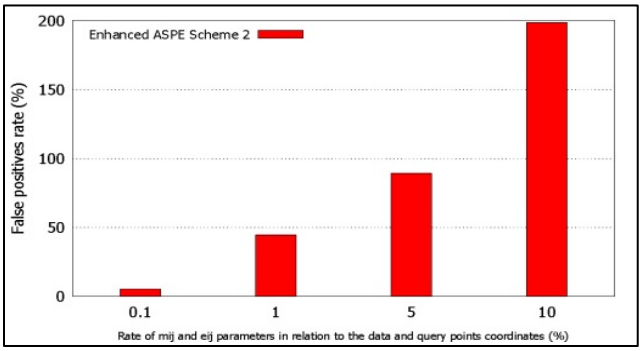


Figure 8: Range query: The impact of the values of the random parameters in Equations (2) and (3) on the false positives rate for the Enhanced ASPE Scheme 2, using the synthetic dataset.

6 CONCLUSIONS AND FUTURE RESEARCH

To address the security concerns relating to the use by enterprises and individuals of relational data management services on the cloud, the paper studies the problem of multi-dimensional (i.e. multi-attribute) range searching over encrypted data on an untrustworthy server, without the need for the mobilization of specialized pre-computed indexing methods or for the intervention of any intermediary trustworthy authority between the client and the server. Prior techniques in the field are either secure but entail prohibitive performance costs, or efficient but involve privacy leakages. The paper introduces two schemes with realistic security and efficiency tradeoffs. More specifically, the security analysis formally determines that the proposed schemes achieve data confidentiality and preserve data and query privacy under the known input-output attack. A prototype implementation and experimental evaluation of the proposed schemes indicate that they provide in practice efficient query processing costs for multi-dimensional cloud database applications.

As regards future plans of research, the proposed schemes will be examined for further optimizations, towards improving the speed of their efficiency without undermining their security. The construction of models for supporting other well-known queries for multi-dimensional data will also be examined.

ACKNOWLEDGMENT

The author would like to thank Mr. Konstantinos Vogiatzoglou for his contribution in respect of the implementation and evaluation processes of this work.

REFERENCES

[1] J.-M. Brook, S. Field, D. Shackleford, et al. 2016. The Treacherous Twelve: Cloud Computing Top Threats in 2016, *Cloud Security Alliance*. Retrieved June 15, 2017 from: https://cloudsecurityalliance.org/group/top-threats/

[2] J. Lewin. 2016. Cyber-attack Cost TalkTalk up to £60m and 101k Customers. *Financial Times,* February 2, 2016.

[3] CipherCloud. 2016. The Global Cloud Data Security Report – The 2016 edition. Retrieved June 15, 2017 from: http://pages.ciphercloud.com/rs/830-ILB-474/images/Q2-Global-Cloud-Data-Security-Report.pdf

[4] HyTrust, Inc. 2016. Cloud Adoption Survey, White Paper. Retrieved June 15, 2017 from: https://www.hytrust.com/uploads/cloud-adoption-survey-report.pdf

[5] H. Hacıgümüs, B. Iyer, C. Li, and S. Mehrotra. 2002. Executing SQL over Encrypted Data in the Database-As-a-Service Model. In *Proceedings of the ACM International Conference on Management of Data* (SIGMOD 2002). 216-227.

[6] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano. 2004. Public Key Encryption with Keyword Search. In *Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques*. 506-522.

[7] W.K. Wong, D.W.L. Cheung, B. Kao, and N. Mamoulis. 2009. Secure kNN Computation on Encrypted Databases. In *Proceedings of the ACM International Conference on Management of Data* (SIGMOD 2009). 139-152.

[8] S. Zerr, D. Olmedilla, W. Nejdl, and W. Siberski: Zerber+r: Top-k Retrieval from a Confidential Index. In *Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology*. 439-449.

[9] D. Boneh, and B. Waters. 2007. Conjunctive, Subset, and Range Queries on Encrypted Data. In *Proceedings of the Theory of Cryptography Conference*. 535-554.

[10] E. Shi, J. Bethencourt, H.T.-H. Chan, D.X. Song, and A. Perrig. 2007. Multi-dimensional Range Query over Encrypted Data. In *Proceedings of the IEEE Symposium Security and Privacy*. 350-364.

[11] M. Wen, R. Lu, K. Zhang, J. Lei, X. Liang, and X. Shen: PaRQ. 2013. A Privacy-preserving Range Query Scheme over Encrypted Metering Data for Smart Grid. *IEEE Transactions on Emerging Topics in Computing*, 1, 1 (2013) 178-191.

[12] B. Hore, S. Mehrotra, and G. Tsudik: A Privacy-Preserving Index for Range Queries. 2004. In *Proceedings of the International Conference on Very Large Data Bases* (VLDB 2004). 720-731.

[13] B. Hore, S. Mehrotra, M. Canim, and M. Kantarcioglu. 2012. Secure Multidimensional Range Queries over Outsourced Data. *VLDB Journal*, 21, 3 (2012) 333-358.

[14] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. 2004. Order Preserving Encryption for Numeric Data, In *Proceedings of the ACM International Conference on Management of Data* (SIGMOD 2004). 563-574.

[15] R.A. Popa, F. H. Li, and N. Zeldovich. 2013. An Ideal-Security Protocol for Order-Preserving Encoding, In *Proceedings of the IEEE Symposium Security and Privacy*. 463-477.

[16] E. Shi, J. Bethencourt, T. Chan, D. Song, and A. Perrig. 2007. Multidimensional Range Query over Encrypted Data, In *Proceedings of the IEEE Symposium Security and Privacy*. 350-364.

[17] R. Li, A.X. Liu, A.L. Wang, and B. Bruhadeshwar. 2014. Fast Range Query Processing with Strong Privacy Protection for Cloud Computing. *Proceedings of the VLDB Endowment*, (2014), 1953-1964.

[18] B. Wang, M. Li, and H. Wang. 2016. Geometric Range Search on Encrypted Spatial Data. *IEEE Transactions on Information Forensics and Security*, 11, 4 (2016), 704-719.

[19] B. Fuhry, R. Bahmani, F. Brasser, F. Hahn, F. Kerschbaum, & A.R. Sadeghi. 2017. HardIDX: Practical and Secure Index with SGX. arXiv preprint arXiv:1703.04583. Retrieved June 15, 2017 from: https://arxiv.org/pdf/1703.04583.pdf

[20] P. Wang, and C.V. Ravishankar. 2013. Secure and Efficient Range Queries on Outsourced Databases using R-trees. In *Proceedings of the IEEE International Conference on Data Engineering* (ICDE 2013). 314-325.

[21] A. Guttman. 1984. R-trees: a Dynamic Index Structure for Spatial Searching, In *Proceedings of the ACM International Conference on Management of Data* (SIGMOD 1984). 47-57.

[22] Y. Manolopoulos, A. Nanopoulos, A.N. Papadopoulos, and Y. Theodoridis. 2010. *R-trees: Theory and Applications*. Springer Science & Business Media.

[23] R. Ostrovsky. 1990. Efficient Computation on Oblivious RAMs. In *Proceedings of the STOC*, 1990.

[24] C. Gentry. 2010. Computing Arbitrary Functions of Encrypted Data. *Communications of the ACM*, 53, 3 (2010), 97-105.

[25] Okeanos Infrastructure as a Service (IaaS), Available at https://okeanos.grnet.gr/home/, valid as of June 2017.

[26] University of California: Machine Learning Repository, Retrieved June 15, 2017 from:http://archive.ics.uci.edu/ml/datasets.html