RASP: Efficient Multidimensional Range Query on Attack-Resilient Encrypted Databases

Keke Chen, Ramakanth Kavuluru, Shumin Guo
Department of Computer Science and Engineering
Wright State University
Dayton, Ohio 45435, USA
{keke.chen,ramakanth.kavuluru, guo.18}@wright.edu

ABSTRACT

Range query is one of the most frequently used queries for online data analytics. Providing such a query service could be expensive for the data owner. With the development of services computing and cloud computing, it has become possible to outsource large databases to database service providers and let the providers maintain the range-query service. With outsourced services, the data owner can greatly reduce the cost in maintaining computing infrastructure and data-rich applications. However, the service provider, although honestly processing queries, may be curious about the hosted data and received queries. Most existing encryption based approaches require linear scan over the entire database, which is inappropriate for online data analytics on large databases. While a few encryption solutions are more focused on efficiency side, they are vulnerable to attackers equipped with certain prior knowledge. We propose the Random Space Encryption (RASP) approach that allows efficient range search with stronger attack resilience than existing efficiency-focused approaches. We use RASP to generate indexable auxiliary data that is resilient to prior knowledge enhanced attacks. Range queries are securely transformed to the encrypted data space and then efficiently processed with a two-stage processing algorithm. We thoroughly studied the potential attacks on the encrypted data and queries at three different levels of prior knowledge available to an attacker. Experimental results on synthetic and real datasets show that this encryption approach allows efficient processing of range queries with high resilience to attacks.

Categories and Subject Descriptors

H.2.0 [**Database Management**]: General—*Security, integrity, and protection*; E.3 [**Data Encryption**]

General Terms

Security, Algorithms

Keywords

Multidimensional Range Query, Random Space Encryption, Attack Analysis, Outsourced Databases

1. INTRODUCTION

With the increasing popularity of web-based applications and the support from widely available cloud infrastructures, service-based computing has become a major computing paradigm. Service providers take advantage of low cost cloud infrastructures, while service users enjoy convenient services without worrying about the cost of maintaining hardware and software. On the other hand, large datasets have been collected, stored, and analyzed in business intelligence and scientific computing for several years. It was reported that maintaining data and supporting query-based services incur much higher cost than initial data acquisition [12]. An appealing solution is to delegate data services to a service provider, which, however, raises the question: how to protect the private information in the outsourced data, considering the service provider might be curious about the data.

Range query is the most frequently used query in online data analytics (OLAP) that requires the service provider to quickly respond to concurrent user queries. To efficiently process range queries, indexing is a necessary step. However, most existing encryption approaches [30, 4, 5, 29] require linear scan over the entire database, thus, impractical for OLAP. Fully homomorphic encryption [13] in theory allows any operation on encrypted data that can be traced back to an equivalent operation on the corresponding plaintexts. However, as the author of [13] mentioned, this is still too expensive to be practical even for a simple application like encrypted keyword search.

Several methods that consider different tradeoffs between data security and efficiency of query processing were proposed in the recent years. Both Crypto-index [20, 21] and order-preserving encryption (OPE) [1, 3] assume the attacker does not have sufficient prior knowledge about the data; thus powerful attacks cannot be conducted. Specifically, they assume the attacker knows only the ciphertext. However, we have found that if the attacker has some prior knowledge, such as the attribute domains (maximum and minimum values), the attribute distributions, and even a few pairs of plaintext and ciphertext, these encryption methods will be vulnerable to attacks. Therefore, although they can allow the service provider to build indices on encrypted data and perform efficient query processing, they can only be applied to very restricted applications. Wang and Lakshmanan [31] use OPE in querying encrypted XML database and address the prior-knowledge enhanced attacks on OPE with duplicated fake index entries that point to the same data item in the encrypted data block. However, their approach requires the data owner to build indices for the server, which is expensive and not convenient when the database is large and frequently updated.

CODASPY'11, February 21–23, 2011, San Antonio, Texas, USA.

Challenge: Therefore, the challenge is to provide an encryption scheme that allows efficient, index based query processing, and is also resilient to prior knowledge enhanced attacks on both data and queries. Our goal is to develop such an encryption scheme.

1.1 Our Contributions and Scope of Research

We propose the RAndom SPace encryption (RASP) approach for efficient range query processing on encrypted data. We assume the outsourced data are multidimensional data and thus the data records can be treated as vectors (or points) in the multidimensional space. The RASP method randomly transforms the multidimensional space, while preserving the convexity of datasets, which allows indexing and query processing with the encrypted multidimensional data. The framework assumes a secure proxy server at the client side that handles data encryption/decryption and query encryption. The data owner and authorized users submit the original data and queries to the proxy server; the proxy server then sends the encrypted data/queries to the service provider. The service provider is able to index the encrypted data and use it to efficiently process encrypted queries.

Our approach has several important features: (1) The RASP approach uses a random space transformation method that allows the service provider to build indices and process queries with multidimensional indices. With the support of indices, the proposed two-stage query processing algorithm can achieve much better performance than linear scan. (2) The existing indexable encryption schemes hold strong assumptions on attacker's lack of prior knowledge on the data; thus they are vulnerable to many attacks enhanced with prior knowledge. Our work categorizes attacker's prior knowledge into three levels and the proposed schemes are resilient to these knowledge-enhanced attacks. To increase resilience against known plain text attacks we use a straightforward composition of Agarwal et al.'s OPE [1] with RASP. (3) Attacks based on queries were rarely discussed in existing schemes. We show that with prior knowledge, attacks on queries can seriously undermine the encryption. We design certain methods to enhance the resilience to the query-based attacks. (4) Some approaches, such as crypto-index, may return a lot of encrypted records irrelevant to the query and burden the client side to filter out these irrelevant records. Our approach always returns the exact result to the client, eliminating the unnecessary additional costs.

We also conduct a number of experiments on synthetic and real datasets to evaluate the performance and the attack resilience. The experimental results show that the proposed method is efficient and resilient to the knowledge enhanced attacks.

The rest of this paper is organized as follows. Section 2 briefly describes range queries and the privacy problems with outsourced databases. Section 3 gives the definition of random space perturbation. In section 4 we present the algorithms for query transformation on the client side and efficient query processing on the server side. In Section 5, we formally analyze the security of the scheme, describe various attacks and discuss the resilience of our scheme to these attacks. The algorithms outlined in Section 4 and Section 5 are summarized at the end of Section 5. The cost of encryption, the resilience to attacks, and the efficiency of query processing are further evaluated through extensive experiments in Section 6.

2. PRELIMINARIES

First, we establish the notation used in this paper. A database table consists of n records and k searchable attributes. We also frequently refer to an attribute as a dimension or column. These three names are exchangeable in our context. Each record can be represented as a vector, and notated with bold lower cases, while lower cases are used to represent scalars. Each column is defined on a domain. For categorical domain, we use integers to represent the categorical values. A table is also represented as a $k \times n$ matrix, notated with capital characters. In the following, we briefly describe the definition of range queries and the importance of indexability to the performance of query processing.

Range Queries: Range query is an important type of query for many data analytic tasks from simple aggregation to more sophisticated machine learning tasks. Let T be a table and X_i, X_j, and X_k be the real valued attributes in T, and a and b be some constants. Take the counting query for example. A typical SQL-style range query looks like

select count(*) from T
where $X_i \in [a_i, b_i]$ *and* $X_j \in (a_j, b_j)$ *and* $X_k = a_k$,

which calculates the number of records in the range defined by conditions on X_i, X_j, and X_k. Range queries may be applied to arbitrary number of attributes and conditions on these attributes combined with conditional operators "and"/"or". We call each part of the query condition that involves only one attribute as a *simple condition.* A simple condition like $X_i \in [a_i, b_i]$ can be described with two half space conditions $X_i \le b_i$ and $-X_i \le -a_i$. Without loss of generality, we will discuss how to process half space conditions like $X_i \le b_i$ in this paper. A slight modification will extend the discussed algorithms to handle other conditions like $X_i < b_i$ and $X_i = b_i$.

3. RANDOM SPACE ENCRYPTION

In this section, we propose the basic Random Space Encryption (RASP) approach for secure range query processing on the encrypted outsourced data. First, we give the system framework and assumptions held for the attack models. Second, we present the definition of the basic random space encryption method and distinguish it from order preserving encryption methods. Finally, we describe how to generate outsourced data and answer queries with the encrypted data.

3.1 System Framework and Attack Models

System Framework. We assume the outsourced data are multidimensional data and thus the data records can be treated as vectors (or points) in the multidimensional space. Figure 1 shows the framework for processing range query services on outsourced data. In the client side, the data owner has all rights to upload/query data, and may also grant the query right to the trusted users. The proxy server receives original data and queries, encrypts and submits them, and decrypts the query results. It keeps the security key, the encryption functions $E_T(), E_Q()$, the decryption function $D()$, and controls the access rights. The traffic between the proxy server and the service provider contains only the encrypted data and queries. Although the proxy server does not handle the large dataset and process queries, it might still become a bottleneck for a large number of users and frequent query submissions. However, the cost to scale the proxy server should be much lower than that to host the entire query processing service.

This framework includes several key components. (1) *Encrypted auxiliary data generation*. This approach will generate the auxiliary data encrypted with the proposed scheme for indexing purpose through the encryption function $E_T()$ in Figure 1. It applies a type of multiplicative perturbation [9, 27] on the searchable attributes in the original database to generate the auxiliary data. The goal is to keep the topology of original data vectors in the auxiliary data but

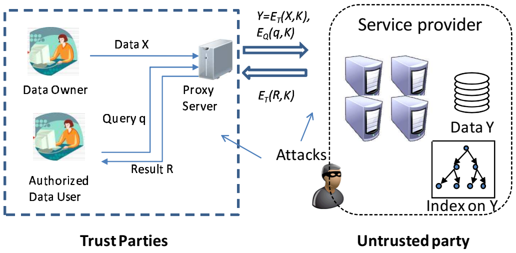

Figure 1: A framework for hosting range query services.

obscure the original data values so that they cannot be possibly inferred from the auxiliary data. (2) *Query Encryption.* A submitted query should also be appropriately transformed so that the server can use the index on the encrypted auxiliary data to process the query. This query transformation should be secure, not reveal any information that helps curious service providers breach privacy. We denote it as the $E_Q()$ function. (3) *Server side indexing and query processing.* The service provider is able to build multidimensional index on the auxiliary data. However, processing the transformed queries requires algorithms different from the existing ones. Our framework also includes the algorithms for query processing.

Attack Models. In our framework, we study the attack models based on the popular honest-but-curious service provider assumption. We assume the service provider will honestly provide the services and perform the computations following the protocol (e.g., public cloud providers). However, the provider might be curious about the data and the users' queries. Also, we assume that the attacker knows the algorithms used to encrypt data and queries (i.e., the algorithms for $E_T(), E_Q()$). Active attackers will also try to obtain as much prior knowledge as possible to help break the encryption, or estimate the encrypted data and queries. To better evaluate the strength of an encryption scheme, we categorize attacks into different levels based on the prior knowledge the attacker may have.

- Level 1: the attacker observes only the encrypted data and queries, without any additional knowledge. This corresponds to the ciphertext-only attack (COA) in cryptography [15].
- Level 2: Apart from the encrypted data, the attacker also knows some dimensional distribution information about the original data, including the attribute domain (the maximum and minimum values) and distribution (e.g., the Probability Density Function (PDF) or histogram). In practice, for some applications such as hosting query services for census data, the dimensional distribution might be known by the public.
- Level 3: In addition to Level 2 knowledge, the attacker observes a small set of plaintext tuples X and the corresponding encrypted tuples Y in the outsourced data. This corresponds to the known-plaintext attack (KPA) in cryptography. In our special context, the attacker may also observe a small number of plain queries and the corresponding encrypted queries.

The three levels also correspond to the difficulty level of obtaining the required prior knowledge. Both Level 2 and Level 3 knowledge are difficult to obtain and it is possible only when the attacker, e.g., the curious service provider, resorts to social engineering or gains temporary unauthorized access to some user accounts at the data owner location. For example, the related database applications may reveal some knowledge about the domain; the compromised user may use queries to probe the encrypted database. The prior knowledge based attacks have also been used in attack analysis for privacy preserving data mining [2, 22, 9] and kNN queries [33] on outsourced data. We will study attacks under these three different levels after we present the basic encryption methods.

3.2 Definition of Random Space Encryption

Random Space Encryption (RASP) is one type of multiplicative perturbation [8], with relaxed constraints on the encryption parameters. Let's consider the multidimensional data are numeric and in multidimensional vector space − For categorical attributes, we use a simple mapping of integers to categorical values to convert them to integers[1]. Assume the database has k searchable dimensions and n records, i.e., a $k \times n$ matrix X. Let $\mathbf{x}$ represent a k-dimensional record. Note that in the k-dimensional vector space, the range query conditions are represented as half space conditions and a range query is correspondingly translated to finding the point set in a hyper-cube [6]. The RASP encryption involves two steps. For each k-dimensional input vector $\mathbf{x}$,

1. the vector is first extended to $k+2$ dimensions as $(\mathbf{x}^T, 1, v)^T$, where the $(k+1)$-th dimension is always a 1 and v is drawn from $(0, \alpha]$ using a random number generator R_α, with some private α and distribution [2].
2. After this extension the $(k+2)$-dimensional vector is then subjected to the transformation

$$E_T(\mathbf{x}, K = \{A, R_v\}) = A \begin{pmatrix} \mathbf{x} \\ 1 \\ v \end{pmatrix}, \qquad (1)$$

   where A is a $(k+2) \times (k+2)$ randomly generated invertible matrix (see Appendix for matrix generation) with $a_{ij} \in \mathbb{R}$ such that there are at least two non-zero values in each row of A and the last column of A is also non-zero.

Note that A is shared by all vectors in the database, but v is randomly generated with the random number generator R_α for each individual vector. Note that only A is needed in the trusted proxy server for decryption (and hence forms the key) - we don't need to keep v in the proxy server. The design of extended data vector $(\mathbf{x}^T, 1, v)^T$ is to address the query-based attacks: the $k+1$ dimension is a homogeneous dimension for hiding the query content; the $k+2$ dimension is used to counter the inherent linearity in transforming the queries. The rationale behind different aspects of this encryption will be discussed clearly in later sections. Also, the structure of A will be slightly changed to withstand known plain text attacks in Section 5.

RASP has two important features. First, we want to show that RASP does not preserve the ordering of dimensional values, which distinguishes itself from order preserving encryption schemes, and thus does not suffer from the bucket-based attack (details in Section 7). Second, we show that RASP is *convexity preserving*, which allows range queries on the encrypted data.

RASP is Not Order Preserving. In the following, we show that RASP is not order preserving; thus the attacks on OPE schemes cannot be applied to RASP. An OPE scheme maps a set of dimensional values to another, while keeping the value order unchanged.

[1] For a categorical attribute X_i, the values $\{c_1, \ldots, c_m\}$ in the domain are mapped to $\{1, \ldots, m\}$. A query condition $X_i = c_j$, is converted to $j - \delta \leq X_i \leq j + \delta$, where $\delta \in (0, 1)$

[2] We used $\alpha = 1$ and uniform random distribution for the experiments

Let $\mathbf{x}$ be any record in the dataset, and $\mathbf{f}^i$ be the selection vector $(0, \dots, 1, \dots, 0)$ i.e., only the i-th dimension is 1 and other dimensions are 0. For simplicity we use unextended vectors $-$ the extended dimensions are not related to this discussion and can be safely removed. Then, $(\mathbf{f}^i)^T\mathbf{x}$ will return the value at dimension i of $\mathbf{x}$.

PROPOSITION 1. *Let A be an invertible matrix with at least two non-zero entries in each row. For any vector $\mathbf{s}$, let $\mathbf{s}' = A\mathbf{s}$. Then for any $i \in \{1, \dots, k\}$ there exist vectors $\mathbf{x}$, $\mathbf{y}$ for which A preserves the order of dimension i (that is, $(x_i - y_i)(x'_i - y'_i) > 0$) and there exist vectors $\mathbf{u}$, $\mathbf{v}$ for which A reverses the order of dimension i. That is, RASP does not preserve order for arbitrary input vector pairs.*

PROOF. Using the same dimensional selection vector, we have $s'_i = (\mathbf{f}^i)^T A\mathbf{s}$ and $t'_i = (\mathbf{f}^i)^T A\mathbf{t}$. Thus, we get

$$\begin{aligned} (s_i - t_i)(s'_i - t'_i) &= (s_i - t_i)(\mathbf{f}^i)^T A(\mathbf{s} - \mathbf{t}) \\ &= (s_i - t_i)\sum_{j=1}^{k} a_{i,j}(s_j - t_j), \end{aligned} \qquad (2)$$

where $a_{i,j}$ is the i-th row j-th column element of A. Without loss of generality, let's assume $s_i > t_i$ (for $s_i < t_i$ the same proof applies). It is straightforward to see that given the fixed values of A, the values of s_j and t_j for all $j \neq i$ can be chosen so that $(s_i - t_i)\sum_{j=1}^{k} a_{i,j}(s_j - t_j)$ is either negative or positive. Note that since each row of A has at least two non-zero entries, even if $a_{i,i}(s_i - t_i)^2 > 0\,(or < 0)$, using the other non-zero value in the i-th row of A, say $a_{i,k}$, the sign of $(s_i - t_i)\sum_{j=1}^{k} a_{i,j}(s_j - t_j)$ can be adjusted to either positive or negative by appropriately choosing the values s_k and t_k. □

RASP is Convexity Preserving. Let's treat data records as points in a real multidimensional space. In the following, we will show that although RASP does not preserve ordering, it preserves convexity, which forms the basis of our query processing strategy. The following definitions of convex set and convexity preserving function can be found in most textbooks on convex optimization, e.g., [6].

DEFINITION 1. *A set S is a convex set, if and only if for $\forall \mathbf{x}_1$, $\mathbf{x}_2 \in S$, and $\forall \theta \in [0, 1]$,*

$$\theta\mathbf{x}_1 + (1 - \theta)\mathbf{x}_2 \in S$$

DEFINITION 2. *A convexity preserving function $E()$ preserves the convexity of sets. Concretely, if S is a convex set in the original data space, the function $E()$ always transforms S to another convex set $E(S)$.*

The following proposition cited from [6] is critical to the proof of convexity preserving property of RASP encryption and our query processing algorithm.

PROPOSITION 2. *(1) Every convex set is a (possibly infinite) intersection of halfspaces, i.e., $\bigcap H_i$, where H_i defines a halfspace; and (2) the intersection of (possibly infinite) convex sets is also convex.*

With this proposition we can prove that

PROPOSITION 3. *RASP encryption is convexity preserving.*

PROOF. We assume an original convex set in $\mathbb{R}^k$ (that is closed) is the intersection of a set of halfspaces $\bigcap H_i$, where a halfspace H_i can be represented as $\mathbf{w}_i^T\mathbf{x} \le a_i$ ("=" for the closed set), and $\mathbf{w}_i \in \mathbb{R}^k$ and $a_i \in \mathbb{R}$ are parameters for the halfspace. By replacing $\mathbf{x}$ with a column vector $\mathbf{z} = (\mathbf{x}^T, 1, v)^T$ and $\mathbf{w}_i$ with $\mathbf{u}_i = (\mathbf{w}_i^T, -a_i, 0)^T$, the set enclosed by H_i is transformed to the set enclosed by the halfspace H_i^{ext}: $\mathbf{u}_i^T\mathbf{z} \le 0$. With the RASP function, we have $\mathbf{y} = A\mathbf{z}$, and thus this halfspace H_i^{ext} can be further transformed to H'_i as follows

$$\mathbf{u}_i^T A^{-1}\mathbf{y} \le 0. \qquad (3)$$

Each of the halfspace conditions, H'_i, in the transformed space represents a convex set. Thus, the intersection of them is convex as well. Therefore, the RASP encryption is convexity preserving. □

Since a range query defines a convex set, the transformation method (Eq. 3) gives a basic method for transforming the range query for the RASP encrypted data - we name it RASP query transformation method. The following proposition shows that by searching with the transformed conditions in the encrypted data space, we can get the exact set of points that is the image of the query result in the original data space.

PROPOSITION 4. *Let H_i and H'_i be halfspaces defined as in the proof of Proposition 3. The RASP query transformation* uniquely *maps the convex set S enclosed by halfspaces $\bigcap H_i$ to the convex set S' enclosed by $\bigcap H'_i$.*

It is straightforward to show that by using the RASP query transformation, any point in S cannot be mapped to a point outside S' and any point not in S cannot be mapped to a point in S'. So we skip the details of the proof.

Note that duplicate records in the original set might be mapped to different records in the encrypted space due to the randomly generated additional dimension $k+2$. However, this query transformation method guarantees all of such records are still exactly found in the encrypted space. This proposition forms the basis for the proposed query processing strategy, which will be discussed in details.

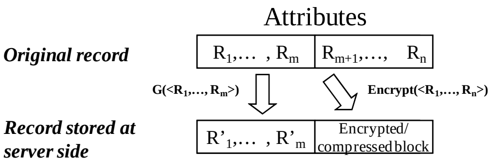

Figure 2: The records stored on the server.

Generating Auxiliary Vectors for Outsourcing. With the RASP encryption function, we generate the outsourced data as follows. First, we normalize each attribute to avoid the attacks based on the value ranges. The normalization process is briefly described as follows. Let the mean of the attribute distribution be μ_i and the variance be σ_i^2. For any value x of the i-th attribute, the transformation $(x - \mu_i)/\sigma_i$ is applied. For the sake of simplifying presentation, we assume the data columns are already normalized - when we say the vector $\mathbf{x}$ we mean it is the normalized version. Second, we assign an unintelligible name to each attribute, e.g. "X_1" for the first attribute. Finally, Eq. 1 is applied to the searchable dimensions $\mathbf{x}$ to generate the encrypted auxiliary record $\mathbf{y}$. $\mathbf{y}$ and the original record that is compressed and encrypted with any existing methods are used for outsourcing (as shown in Figure 2). The service provider may build indices or perform linear scan on the auxiliary data vectors to answer queries. The cost for generating an outsourced record consists of one RASP encryption perturbation

($\mathcal{O}(k^2)$ multiplications; k is the number of searchable dimensions) and the compression/encryption operations applied to the whole record. Here note that the RASP transformation is applied only to those attributes that are actually queried. Thus like mentioned earlier, conventional encryption can be used on compressed values of attributes that will not be queried.

4. EFFICIENT RANGE QUERY PROCESSING WITH RASP

We have shown that the RASP encryption is convexity preserving. This result is closely related to how a query can be transformed and processed. A range query can be represented as a convex set query. Thus, in the encrypted space there is a unique convex set that is the answer to the query. However, there are challenges in (1) efficiently processing it, and (2) making sure query processing does not reveal significant information about the encryption key and the original data. One may already notice that the simple query transformation method described in this section is vulnerable to attacks. However, in this section, we will focus on the first challenge. It will be revisited and significantly improved in security analysis in Section 5.

In the encrypted space, a simple dimensional condition in the original space is transformed to a general halfspace condition (as Figure 4 shows). It would be straightforward to scan each auxiliary vector with the transformed conditions and return the result. We want to explore more efficient index-based processing methods in this section. The normal processing strategies are based on multidimensional index trees, such as R-Tree [28], that handles axis-aligned minimum bounding boxes (MBR). If we still depend on multidimensional tree indexing to process the transformed queries, the processing algorithm should be slightly modified to handle arbitrary convex areas, the boundaries of which are not necessarily axis-aligned. We will start with the method of query transformation, briefly discuss the normal range query processing algorithms using multidimensional indices, and then present the proposed solution for processing the transformed queries.

4.1 Query Transformation

Since the auxiliary vectors are in the encrypted space, to query on this space, range queries should also be appropriately transformed. We have mentioned that the transformation method used in proving Proposition 3 can be used for query transformation. In this section, we discuss how to transform an original range query into the encrypted space in details.

First, let's look at the general form of a range query condition. Let X_i be an attribute in the database. A simple condition in a range query involves only one attribute and is of the form "X_i <*op*> a_i", where a_i is a constant in the normalized domain of X_i and *op* $\in \{<, >, =, \leq, \geq, \neq\}$ is a comparison operator. For convenience we will only discuss how to process $X_i \leq a_i$, while the proposed method can be slightly changed for other conditions. Any complicated range query can be transformed into the disjunction of a set of conjunctions, i.e., $\bigcup_{j=1}^{n}(\bigcap_{i=1}^{m} C_{i,j})$, where m, n are some integers depending on the original query conditions and $C_{i,j}$ is a simple condition about X_i. Again, to simplify the presentation we restrict our discussion to single conjunction condition $\cap_{i=1}^{m} C_i$. A simple condition $X_i \leq a_i$ is a halfspace condition. Following the previous discussion, $X_i \leq a_i$ is converted to the extended vector representation first: $\mathbf{u}^T\mathbf{z} \leq 0$, where $\mathbf{u}$ is a $k+2$ dimensional vector with $u_i = 1, u_{k+1} = -a_i$, and $u_j = 0$ for $j \neq i, k+1$, (for $X_i \geq a_i$, $u_i = -1, u_{k+1} = a_i$), and $\mathbf{z} = (\mathbf{x}^T, 1, v)^T$. Then, let $\mathbf{y}$ be the auxiliary vector, i.e., $\mathbf{y} = A\mathbf{z}$. The original condition is transformed to the form of Eq. 3 in the encrypted space.

As Proposition 4 shows, searching with the transformed queries on the auxiliary vectors is equivalent to searching with the original queries and data. Note that this simple query transformation is vulnerable to attacks as shown in Section 5.2; we will eventually use slightly different query transformation method. (In the Appendix, we also give the details of the transformed query.) Next, we show how to efficiently process these transformed queries.

4.2 A Two-Stage Query Processing Strategy with Multidimensional Index Tree

With the transformed queries, the first important task is to process queries efficiently. A commonly used method is to use tree indices to improve the search performance. However, multidimensional tree indices are normally used to process axis-aligned "bounding boxes"; whereas, the transformed queries are in arbitrary convex shape, not necessarily aligned to axes. In this section, we propose a two stage query processing strategy to handle such irregular shape queries in the encrypted space. First, we briefly introduce the query processing algorithm based on multidimensional index trees. Then, we describe the two stage processing algorithm.

Multidimensional Index Tree. Most multidimensional indexing algorithms are derived from R-tree like algorithms [19], where the minimum bounding region (MBR) is the construction block for the multidimensional data. For 2D data, an MBR is a rectangle. For higher dimensions, the concept of a MBR is extended to a hypercube. Figure 3 shows the MBRs in the R-tree for a 2D dataset, where each node is bounded by a node MBR.

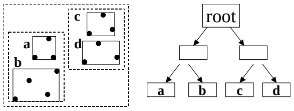

Figure 3: R-tree index.

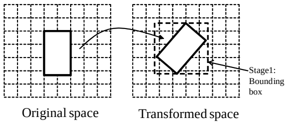

Figure 4: Illustration of the two-stage processing algorithm.

Range query processing with a multidimensional indexing tree can be described as follows. The conjunction of a set of simple range conditions can be represented as a query MBR. The goal is to find the MBRs in the tree that are contained by or intersected with the search MBR. If the query MBR contains a node MBR, all points in the subtree should be included in the query result. If the query MBR intersects a node MBR, further checking should be performed for the children nodes. If the query MBR intersects a leaf MBR, each point included by the leaf node should be checked and only those inside the query MBR are selected.

The Two-Stage Processing Algorithm. The transformed query describes an irregular convex shape that cannot be directly processed by multidimensional tree algorithms. New tree search algorithms can be designed to use arbitrary polyhedron conditions, i.e., the transformed query, directly for search. However, we use a simpler two-stage solution that keeps the existing tree search algorithms unchanged.

At the first stage, the proxy in the client side finds the MBR of polyhedron (as a part of the submitted transformed query) and the server uses the MBR to find the initial result set. We use the simple *vertex-based algorithm* for finding the MBR of the polyhedron. The original query condition constructs a hyper-cube shape. With the query transformation, the vertices of the hyper cube are also transformed to vertices of the polyhedron. Therefore, we can calculate the MBR with only the transformed vertices. Figure 4 illustrates the relationship between the vertices and the MBR. There are a maximum number of 2^k vertices for one conjunctive range query condition on k dimensions, i.e., each dimension has its lower and upper bounds. It is straightforward to construct these vertices based on the dimensional bounds. In practice, the MBR of the polyhedron needs to be calculated by the proxy server for security reason, and then sent to the server together with the transformed queries.

At the second stage, the server uses the transformed halfspace conditions [3] to filter the initial result and find the final result. In most cases, the initial result set will be reasonably small so that it can be filtered in memory with linear scan. In the worst case, the MBR of the polyhedron will possibly enclose the entire dataset and the second stage is reduced to linear scan of the entire dataset.

Cost Analysis. Assume the query ranges are selected uniformly at random. For small ranges the first stage average cost is $\mathcal{O}(\log_B N)$ index block accesses plus a few of data block accesses [28], where N is the number of records and B is the number of children an index node has. Due to the randomness associated with the RASP transformation, the data distribution, and the unpredictable query ranges, the cost to get the initial result could vary, which will be investigated in experiments. If the initial result has n records, a linear scan at the 2nd stage with $2k$ simple conditions will cost $\leq 2kn$ checks of the form in Eq. 3. In Section 6, we study the cost distribution between the two stages and experimentally demonstrate that this two-stage processing is efficient and orders of magnitudes faster than the linear scan approach.

5. ATTACK ANALYSIS

We categorize the possible attacks into two types: (1) Attacks on auxiliary vectors; (2) Attacks based on range queries. There has been some related work on attack analysis methods for similar encryption methods, e.g., geometric data perturbation for data mining [9], which can be migrated to analyze the first type of attacks. However, attacks on range queries are entirely new for our approach.

5.1 Attacks on Auxiliary Vectors

According to the three levels of knowledge the attacker may have, we categorize the attacks into three classes: (1) Naive estimation; (2) Distributional Attacks; and (3) Known Input/Output Attacks. Due to the random component in the RASP encryption, some attacks are actually estimation attacks, i.e., the goal of the attack is to estimate the original values. If the estimation result is sufficiently accurate, we say the encryption is broken.

5.1.1 *Attack Description and Analysis*

Naive Estimation. With the level 1 knowledge, the attacker observes only the encrypted data. The only attack is to blindly guess the matrix A. It has been discussed to find a matrix A to maximize the difference between the encrypted data and the original data [9]. However, since there is no way to verify how accurate a random guess is, this type of attack is ineffective, in general.

Distributional Attack. With the level 2 knowledge, the attacker also knows column domains and distributions. This knowledge can be possibly used to perform more effective attacks. In particular, when the original data have independent columns and no more than one column having Gaussian distribution, an attack called Independent Component Analysis (ICA) [23] can be applied to effectively recover the original data from the perturbed data. Originally developed for signal processing, ICA is used to discover components A (the mixing matrix) and X (the original signals) from the mixed data $Y = AX$. Since ICA recovers columns in an arbitrary order, it has to rely on the known distributional information to distinguish the columns and order them correctly. Furthermore, the effectiveness of ICA heavily depends on the independence of the columns and the number of columns having non-Gaussian distributions . In practice, since the independence condition and the Gaussian distribution condition are often not satisfied, the ICA attack can only result in approximate estimation to the original data. However, the previous study [9] shows that if the matrix A is not carefully selected, the ICA attack can still result in serious damage.

Another distributional attack is to enumerate the matrix A and then check the column distributions of $A^{-1}Y$ to find the best match between the known column distributions and the distributions of the estimated columns. However, since there is no constraint on the elements of A, with uniformly discretized domains, the number of candidate matrices will be extremely large. Concretely, if the discretized domain has d values, the total candidate matrices will be $d^{(k+2)^2}$, where k is the number of dimensions. Even for extremely low dimensionality, e.g., k=2, this attack could be computationally intractable.

Known Input/Output Attack. With the level 3 knowledge, the attacker knows a number of input/output (plaintext/ciphertext) record pairs. Concretely, let $P_{k\times m}$ be the known m k-dimensional original records $(\mathbf{x}_1, \ldots, \mathbf{x}_m), m > k+2$, that include $k+2$ linearly independent records, and $Q_{k+2\times m}$ be the corresponding perturbed $k+2$-dimensional records $(\mathbf{y}_1, \ldots, \mathbf{y}_m)$. The typical method is to use the linear regression method to get an estimate of the key and then recover the entire original data. In the following, we show how to use the regression method to attack the encryption. Let A decomposed into blocks $A = (A_1, A_2, A_3)$, where A_1, A_2 and A_3 have block sizes $(k+2)\times k$, $(k+2)\times 1$ and $(k+2)\times 1$, respectively, and the extended data be $\begin{pmatrix} P \\ \mathbf{1} \\ \mathbf{v} \end{pmatrix}$ where $\mathbf{1}$ is the row vector with '1' and $\mathbf{v}$ is a row vector with random positive values, corresponding to the two additional dimensions in RASP. Thus, the encryption can be represented as

$$Q = (A_1, A_2, A_3)\begin{pmatrix} P \\ \mathbf{1} \\ \mathbf{v} \end{pmatrix} = A_1P + A_2\mathbf{1} + A_3\mathbf{v}, \qquad (4)$$

where $A_2\mathbf{1}$ is a translation matrix that adds the vector A_2 to each of the column vectors in A_1X; $A_3\mathbf{v}$ is a random noise matrix. With sufficient number of known record pairs ($m > k+2$), first, the translation component can be canceled out; then, the regression method can be applied to estimate A_1. With the estimate $\hat{A}_1$ of A_1, A_2 can be estimated as well. Therefore, for an encrypted dataset Y, the estimate of the original data X is $\hat{X} = (\hat{A}_1^T\hat{A}_1)^{-1}\hat{A}_1^T(Y - \hat{A}_2\mathbf{1})$.

5.1.2 *Countering Attacks on Auxiliary Data*

Countering ICA-based Distributional Attack. Since the enumeration based attack is computationally intractable, we focus on the ICA-based attack. We propose two approaches to increase the resilience to the attack. The first approach is to simulate the ICA attack in sufficient rounds to find a statistically resilient A matrix

[3] The final form of the security-enhanced transformed query is represented with the matrices Θ_is that are described in Section 5.2.

as the previous work does [9]. However, a more attack-resilient approach is using the composition encryption scheme (CES) that consists of two steps: transforming the original data with an order preserving encryption scheme E_o first; then followed by the basic RASP encryption, which can be represented as follows.

$$E(X, K, K_o) = A \begin{pmatrix} E_o(X, K_o) \\ \mathbf{1} \\ \mathbf{v} \end{pmatrix} \tag{5}$$

We use the OPE scheme by Agarwal et al. [1] that allows us to change all column distributions to normal distributions. Thus, the requirement of non-Gaussian distribution is not satisfied, which renders ICA ineffective. Since the composition scheme is not order preserving, the attacks on OPE schemes will not be applicable either. However, can the two-stage query processing strategy still be applied? The following proposition indicates that it can still be applied.

PROPOSITION 5. *Order preserving encryption functions transform a hyper-cubic query range to another hyper-cubic query range.*

PROOF. Assume the original range query condition consists of simple conditions like $b_i \le X_i \le a_i$ for each dimension. Since the order is preserved, each simple condition is transformed as follows: $E_o(b_i) \le E_o(X_i) \le E_o(a_i)$, which means the transformed range is still a hyper-cubic query range. □

When processing a query, the proxy server needs to transform the ranges to OPE encrypted ranges first, and then apply the query transformation method. We will show in the experiments how the resilience to the ICA attack is improved with the composition scheme.

Countering Known Input/Output Attacks. As we have mentioned earlier, the random noise matrix $A_3\mathbf{v}$ in Eq. 4 determines how effective the linear regression estimation can be done. The more intense (in terms of variance) the noise component is, the less accurate the estimation can be. A randomly generated matrix A does not allow us to control the noise intensity, however. We propose to use the following method to generate A_3 that satisfying a specified noise intensity. (1) generate a $k+2$ by β random matrix Ψ according to the required noise distribution and intensity, with sample size β that is sufficiently large, or larger than the maximum number of known Input/Output records that an attacker may have access to; (2) generate β positive random values $\mathbf{v}$; (3) apply $A_3 = \Psi\mathbf{v}^T/(\mathbf{v}\mathbf{v}^T)$ to get A_3. The last column of the randomly generated A is then replaced with A_3. Note here that having a value of zero for say the i-th entry in A_3 would mean that the noise values are never added to i-th attribute. So the random generation process is repeated until all elements of A_3 are not zero. Previous study shows that Principle Component Analysis (PCA) can be used to possibly filter out the noise component or reduce the effect of noise in some circumstances [22]. We will investigate the relationship between the noise component and the accuracy of estimation, and study whether PCA can help improve the estimation in experiments.

A more attack-resilient solution is using the previous discussed composition encryption scheme. If the OPE scheme uses a nonlinear transformation, the composition of OPE and RASP will create a nonlinear mapping from the original data to the encrypted data. Therefore, the linear regression attack will not be effective by simply using the known pairs of input and output records, if the OPE key is unknown.

5.2 Attacks on Transformed Queries

As we have discussed, in query processing, the proxy server will submit the MBR and the transformed query to the server. We refer to the original query as the *input* query, and the transformed query that is submitted to the server as the *output* query. With the knowledge of a number of pairs of input/output queries, the following attacks can be performed on the current query conditions, e.g., $\mathbf{u}^T A^{-1}\mathbf{y} \le 0$, to break the encryption.

5.2.1 Attack Description and Analysis

First, we will show that the row vectors of A^{-1} can be probed if the attacker has the level 3 knowledge on query conditions. Second, we show a more serious attack that can reveal columns of data if the attacker has the level 2 knowledge about dimension distributions.

A^{-1} Probing Attack. With the level 3 knowledge, the attacker knows a pair of input query conditions on the same dimension, say $X_i \le a_i$ and $X_i \le b_i$, and their output forms, $\mathbf{u}_{a_i}^T A^{-1}\mathbf{y} \le 0$ and $\mathbf{u}_{b_i}^T A^{-1}\mathbf{y} \le 0$, respectively. Then, $\mathbf{u}_{a_i}^T A^{-1}\mathbf{y} - \mathbf{u}_{a_i}^T A^{-1}\mathbf{y} = \mathbf{u}_{a_i-b_i}^T A^{-1}\mathbf{y}$, where $\mathbf{u}_{a_i-b_i}^T = (0, \dots, a_i - b_i, 0)$, only the nonzero $(k+1)$-th element remains. Let $\mathbf{r}_j$ be the j-th row of A^{-1}. The constant part of the condition represents $(a_i - b_i)\mathbf{r}_{k+1}$, thus revealing $\mathbf{r}_{k+1}$. While the single condition like $\mathbf{u}_{a_i}^T A^{-1}\mathbf{y} \le 0$ has the constant part $\mathbf{r}_i + a_i\mathbf{r}_{k+1}$, with known a_i and $\mathbf{r}_{k+1}$, $\mathbf{r}_i$ can be revealed. Repeating this process for all dimensions with known input/output conditions, the attacker can recover $k+1$ rows of the matrix A^{-1}, which leaves very weak security.

Dimensional Selection Attack. With the level 2 knowledge, i.e., the column domains and the column distributions, the attacker can perform a *dimensional selection attack*. Assume the condition is applied to some unknown dimension i. Applying the query parameters $\mathbf{u}_i^T A^{-1}$ to each record $\mathbf{y}$ in the server, the attacker can get $\mathbf{u}_i^T A^{-1}\mathbf{y} = x_i - a_i$, where x_i is the i-th dimension of the corresponding original record $\mathbf{x}$. After getting all the values, the attacker can build up a histogram to compare with the known column distributions. It is thus easy to identify what dimension is queried. With the knowledge of the column domain, the constant a_i can be easily removed, which leads to complete breach of the entire column i.

In summary, the original query transformation method can be exploited to construct very effective attacks. It needs to be carefully redesigned to address these attacks.

5.2.2 Countering Query-based Attacks

The additional dimension X_{k+2} is used to construct secure query conditions. Instead of processing a half space condition $X_i \le a_i$, we use $(X_i - a_i)X_{k+2} \le 0$ instead. These two conditions are equivalent because the additional dimension X_{k+2} satisfies $X_{k+2} > 0$. Using the extended vector form $\mathbf{z}^T = (\mathbf{x}^T, 1, v)$, we have $X_i - a_i = \mathbf{z}^T\mathbf{u}$ and $X_{k+2} = \mathbf{w}^T\mathbf{z}$, where $u_i = 1$, $u_{k+1} = -a_i$, $u_j = 0$, for $j \ne i, k+1$; $w_{k+2} = 1$ and $w_j = 0$, for $j \ne k+2$. With the transformation $\mathbf{y} = A\mathbf{z}$, we get the transformed quadratic query condition

$$\mathbf{y}^T(A^{-1})^T\mathbf{u}\mathbf{w}^T A^{-1}\mathbf{y} \le 0. \tag{6}$$

Let $\Theta = (A^{-1})^T\mathbf{u}\mathbf{w}^T A^{-1}$. In the two-stage processing strategy, the bounding box of the transformed query area is calculated in the proxy server as we discussed earlier. Then, this bounding box and the parameters Θ_i for each condition i, are submitted to the server. Thus, assume each dimension is represented with two half space conditions, the encrypted query E_Q is represented as {MBR,$\{\Theta_1, \dots, \Theta_{2k}\}$}. The server will use the bounding box to get the first-stage results and then use the conditions, e.g., $\mathbf{y}^T\Theta_i\mathbf{y} \le 0$, to filter out the results. We now show that this query transformation is resilient to both query-based attacks.

Assume the attacker knows two pairs of input/output query conditions, e.g., for $X_i \le a_i$ and $X_i \le b_i$. We use the same method used in the A^{-1} probing attack to find the difference of the two conditions. Let Θ_a and Θ_b notate the parameters for these two conditions, respectively. The simplified form for a single condition, e.g., Θ_a, is $(\mathbf{r}_i^T - a_i\mathbf{r}_{k+1}^T)\mathbf{r}_{k+2}$, where $\mathbf{r}_i, \mathbf{r}_{k+1}$, and $\mathbf{r}_{k+2}$ are the row vectors of matrix A^{-1}. Thus, the result of $\Theta_a - \Theta_b$ is $(b_i - a_i)\mathbf{r}_{k+1}^T\mathbf{r}_{k+2}$. But knowing Θ_a, Θ_b, a_i, b_i does not help find the unknown vectors $-$ in fact there are an infinite number of solutions because $\mathbf{r}_{k+1} = \frac{(\Theta_a - \Theta_b)\mathbf{r}_{k+2}}{(b_i - a_i)||\mathbf{r}_{k+2}||}$. Therefore, knowing pairs of input/ouput queries does not help probing A^{-1}.

This quadratic query transformation method counters the dimensional selection attack as well. For any perturbed record $\mathbf{y}$, $\mathbf{y}^T\Theta_a\mathbf{y}$ recovers $(x_i - a_i)x_{k+2}$, where x_i and x_{k+2} are the dimensional values of the corresponding original vector $\mathbf{x}$. Since x_{k+2} is a randomly generated positive value, knowing X_i's domain and distribution does not help recover x_i. Therefore, dimensional selection attack does not work either.

In the appendix, we put together all the algorithms after considering the attacks discussed in this section.

6. EXPERIMENTS

In this section, we present three sets of experimental results to investigate the following questions: (1) How costly are the RASP encryption scheme and the composition scheme involving OPE scheme? (2) How effective are the ICA attack and the known input/output attack, if the composition scheme is not applied? (3) How efficient is the two-phase query processing?

6.1 Setup

Two datasets are used in experiments: (1) a synthetic dataset that draws samples from uniform distribution in the range [0, 1]; and (2) the Adult dataset from UCI machine learning database[4]. For the adult dataset, we assign numeric values to the categorical values using a simple one-to-one mapping scheme. For each dataset, we generate multiple versions with different numbers of records by using sampling with replacement. We also change the dimensionality of the datasets by randomly selecting a number of dimensions of the data. All experiments were done in a quad-core AMD Opteron server (2.5GHz CPU and 120GB memory).

6.2 Cost of Encryption

In this experiment, we study the cost of the components in the composition scheme. We implement the OPE scheme [1] by mapping original column distributions to normal distributions. The OPE algorithm partitions the target distribution into buckets, first. Then, the sorted original values are proportionally partitioned according to the target bucket distribution to create the buckets for the original distribution. With the aligned original and target buckets, an original value can be mapped to the target bucket and appropriately scaled. Therefore, the encryption cost mainly comes from the bucket search procedure (proportional to $\log D$, where D is the number of buckets). Both encryption schemes are implemented with Matlab. Figure 5 shows the cost distribution for 20K records at different number of dimensions of data for the two components in the composite scheme. The dimensionality has less effect on the cost of RASP than on that of OPE.

6.3 Resilience to Estimation Attacks

We have discussed the methods for countering the estimation attacks, primarily the ICA attack and the known input/output attack. In this set of experiments, we explore the resilience of both the RASP-Only scheme and the composition scheme to the estimation attacks. Although the composition scheme is more resilient to attacks, it incurs the additional cost that might not be favored by some applications. Therefore, it is worth looking at how resilient the RASP-Only scheme is to the attacks.

Metric for Evaluating Estimation Attacks. The accuracy of estimation attacks can be evaluated with the well-known mean square error (MSE). Let the number of records be n and the value of the i-th attribute in the j-th record be $x_{i,j}$ and the corresponding estimated value be $\hat{x}_{i,j}$. Model the i-th attribute with a random variable X_i and its estimate as $\hat{X}_i$. The estimation error can be estimated with the root of mean square error (RMSE): $m_i = \sqrt{1/n\sum_{j=1}^{n}(x_{i,j} - \hat{x}_{i,j})^2}$, i.e., $X_i = \hat{X}_i \pm m_i$. $2m_i$ can be used to roughly represent the effective estimation range. Apparently, m_i has different meaning for different value range. For example, ± 10 means ineffective estimation for an attribute "age" (in the range [0,100]), while very effective for "salary" (often $>$ 10000). One of the common methods is to normalize all attributes to approximately the same range. For large data, the assumption that each attribute has approximately normal distribution would be reasonable [25]. Therefore, standardization can be used to normalize all attributes to normal distribution with mean zero and standard deviation one. For a standardized domain, four times of the standard deviation (i.e., $4\sigma = 4$) covers the majority of records ($> 95\%$). Then, we can use the rate $p_i = 2m_i/(4\sigma) = m_i/2$ to represent the relative effectiveness of the estimation attack. The larger the m_i, the less effective the estimation is. To evaluate the resilience across all attributes, we also define dataset-wise metrics, such as the minimum security guarantee $p^{min} = \min\{p_i, 1 \le i \le k\}$, which is used in our experiment.

Results. We simulate the ICA attack for randomly chosen matrices A. The data used in the experiment is the 10-dimension Adult data with 10K records. The x-axis in Figure 6 represents the sequence number of randomly chosen matrix A and the y-axis represents the minimum security guarantee among all dimensions. The label "Best" means the most resilient A to the ICA attack; "Worst" means the A shows the weakest resilience; "Average" is the progressive average resilience for the generated A matrices. Figure 6 shows that the effectiveness of the ICA attack can vary with different matrices A and we can find some ones that are more resilient to the attack. In addition, if applied is the composition method that uses the OPE scheme to change column distributions to Gaussian distributions, the resilience of a randomly chosen A is significantly increased.

We also simulate the known input/output attack with a number of randomly selected input/output records pairs (10% of the entire dataset). The original data is generated with the method mentioned in Section 5.2.2 by generating the noise matrix with standard normal distribution $N(0, 1)$. Due to the randomness, we repeat the experiment 10 times and record the variance of the estimation. The PCA based noise filtering technique [22] is also applied as a part of the attack. Let Y be the encrypted data. The PCA method finds the eigenvalue decomposition of YY^T. Let Q be the eigenvectors corresponding to the largest p preserved eigenvalues (i.e., the principal components). The noise filtering algorithm uses YQQ^T to represent Y. Figure 7 shows the result for the known I/O attack

[4] http://archive.ics.uci.edu/ml/

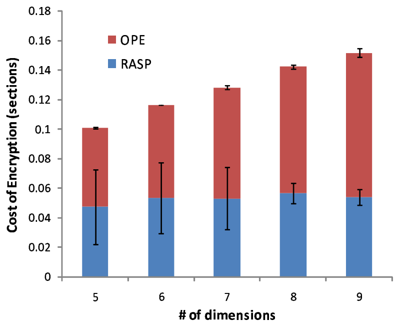

Figure 5: The cost distribution of the composition encryption scheme. Data: Adult (20K records,5-9 dimensions)

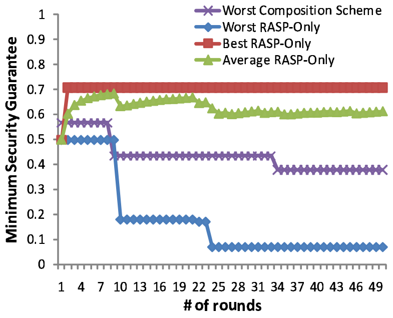

Figure 6: Randomly generated matrix A and the resilience to ICA attack. Data: Adult (10dimensions, 10K records)

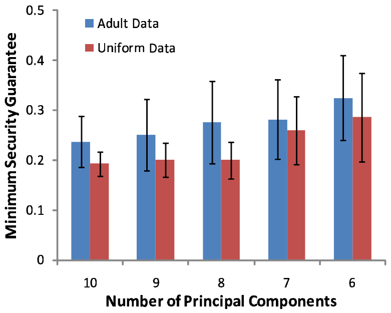

Figure 7: Known input/output attacks on both Adult and Uniform data.

with the PCA noise filtering step. The x-axis represents the number of principal components preserved. Since the data dimensionality is 10, 10 principal components means no noise reduction is applied. For both datasets, the average estimation errors are higher than 0.2. Also, the PCA noise filtering does not help much – when the number of principle components is reduced (trying to remove the noises), the estimation error does not reduce. This result shows that with appropriately set noise component, the known I/O attack is not effective either.

6.4 Performance of Two-stage Range Query Processing

In this set of experiments, we study the performance aspects of polyhedron range query processing. We use the two-stage processing strategy described in Section 4, and explore the additional cost incurred by this processing strategy. We implement the two-stage query processing based on an R*-tree implementation provided by Dr. Hadjieleftheriou at AT&T Lab http://www2.research.att.com/ marioh/spatialindex/. The block size is 4KB and we allow each block to contain only 20 entries to mimic a large database with many disk blocks. Samples from the three databases in different size (10,000 – 50,000 records, i.e., 500-2500 data blocks) are encrypted as the auxiliary data and then indexed for query processing. Another set of indices are also built on the original data for setting up the performance baseline of query processing on non-encrypted data. We will use the number of disk block accesses, including index blocks and data blocks, to assess the performance to avoid the possible variation caused by other parts of the computer system. In addition, we also show the wall-clock time for some results for comparison.

Recall the two-stage processing strategy: (1) calculate the MBR of the transformed query and use the MBR to search the indexing tree; (2) filter the returned result with the transformed query. We will study the performance of the first stage by comparing it to two additional methods: (1) the original queries with the index built on the original data, which is used to identify how much additional cost is paid for querying the MBR of the transformed query; (2) the linear scan approach, which is the worst case cost. Range queries are generated randomly within the domain of the datasets, and then transformed with the method described in the Section 4.1. We also control the range of the queries to be [10%,20%,30%,40%,50%] of the total range of the domain, to observe the effect of the scale of the range to the performance of query processing.

Results. The first pair of figures (the left subfigures of Figure 8 and 9) shows the number of block accesses for 10,000 queries on different sizes of data with different query processing methods. For clear presentation, we use $\log_{10}$(# of block accesses) as the y-axis. The cost of linear scan is simply the number of blocks for storing the whole dataset. The data dimensionality is fixed to 5 and the query range is set to 30% of the whole domain. Obviously, the first stage with MBR for polyhedron has a cost much cheaper than the linear scan method and only moderately higher than R*tree processing on the original data. Interestingly, different distributions of data result in slightly different patterns. The costs of R*tree on transformed queries are very close to those of original queries for Adult data, while the gap is larger on uniform data. The costs over different dimensions and different query ranges show similar patterns.

	Linear Scan	R*Tree-Orig	Stage-1	Stage-2	rpq	purity
Uniform	6.32	0.132	0.805	0.041	60	1.3%
Adult	5.42	0.091	0.20	0.017	24	2.2%

Table 1: Wall clock cost distribution and comparison.

We also studied the cost of the second stage. We use "purity" to represent the rate (final result count)/(1st stage result count), and records per query (RPQ) to represent the average number of records per query for the first stage results. The quadratic filtering conditions are used in experiments. Table 1 compares the average wall-clock time (milliseconds) per query for the two stages, the RPQ values for stage 1, and the purity of the stage-1 result. The tests are run with the setting of 10K queries, 20K records, 30% dimensional query range and 5 dimensions. Since the 2nd stage is done in memory, its cost is much lower than the 1st-stage cost. Overall, the two stage processing is much faster than linear scan and comparable to the original R*Tree processing.

7. RELATED WORK

We review the two most related methods: OPE and crypto-index first, and then give other related work.

OPE. As the name indicates, order preserving encryption (OPE) [1] preserves the dimensional value order after encryption. It can be described as a function $y = F(x), \forall x_i, x_j, x_i < (>, =) x_j \Leftrightarrow y_i < (>, =) y_j$. A well-known attack is based on attacker's prior knowledge of original distributions of attribute values. If the attacker knows the original distributions and manages to identify the mapping between the original attribute and its encrypted counterpart, the following bucket-based attack can be performed to break the encryption for the attribute. (1) Model the original distribution for the attribute with a histogram of a number of buckets; (2)

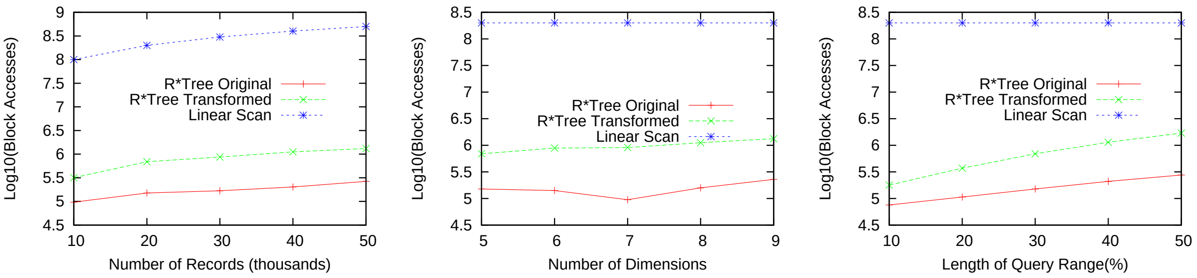

Figure 8: Performance comparison on Uniform data. Left: data size vs. cost of query; Middle: data dimensionality vs. cost of query; Right: query range (percentage of the domain) vs. cost of query

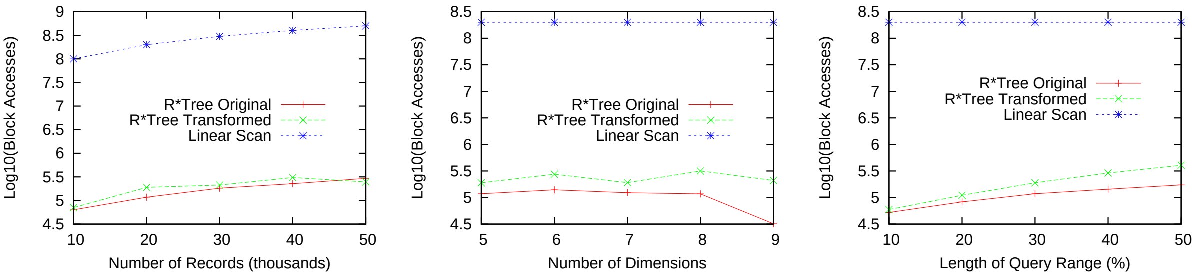

Figure 9: Performance comparison on Adult data. Left: data size vs. cost of query; Middle: data dimensionality vs. cost of query; Right: query range (percentage of the domain) vs. cost of query

Calculate the percentage of each bucket to the entire distribution; (3) Sort the encrypted values; (4) According the bucket's percentages, sequentially partition the sorted encrypted values to generate buckets; (4) Sequentially map the encrypted buckets to the original buckets and get the estimate of the encrypted value. The precision of estimation is determined by the width of the buckets - the narrower the buckets are the higher the precision will be. Since the number of buckets can be arbitrarily chosen, the bucket width can be very small. It is also not difficult to get the mapping between the original attribute and the encrypted attribute, if the attacker knows a number of plain queries and their encrypted queries. In developing our schemes, we have carefully studied whether the known query patterns will also damage the proposed encryption scheme in our framework.

Crypto-Index. Crypto-Index is also based on column-wise bucketization. It assigns a random ID to each bucket; the values in the bucket are replaced with the bucket ID to generate the auxiliary data for indexing. To utilize the index for query processing, a normal range query condition has to be transformed to a set-based query on the bucket IDs. For example, $X_i < a_i$ might be replaced with $X'_i \in [ID_1, ID_2, ID3]$. If the attacker manages to know the mapping between the input original query and the output bucket-based query, the range that a bucket ID represents could be estimated. The width of the bucket determines how precise the estimation could be done. A bucket-diffusion scheme [21] was proposed to address this problem, which, however, has to sacrifice the precision of query results. Another drawback of this method is that the client, not the server, has to filter out the query result. Low precision results raise large burden on the network and the client system. Furthermore, due to the randomized bucket IDs, the index built on bucket IDs is not so efficient for processing range queries as the index on OPE encrypted data is.

Other Related Work. Private information retrieval (PIR) [10, 24] tries to fully preserve the privacy of access pattern, while the data may not be encrypted. PIR schemes are normally very costly. Focusing on the efficiency side of PIR, Williams et al. [32] use a pyramid hash index to implement efficient privacy preserving data-block operations based on the idea of Oblivious RAM [16]. It is different from our setting of high throughput range query processing. Another line of research [5, 29] facilitates authorized users to access only the portion of data in the authorized range with a public key scheme. The underlying identity based encryption used in these schemes does not produce indexable encrypted data. Also the setting for which Shi et al. [29] propose the multidimensional range query is different from ours. The untrusted service provider in our setting is responsible for both indexing and query processing. Secure keyword search on encrypted documents [30, 17, 14, 4, 11] scans each encrypted document in the database and finds the documents containing the keyword, which is more like point search in database. The research on privacy preserving data mining has discussed multiplicative perturbation methods [7, 9, 27, 26, 18], which are similar to the RASP encryption, but with more emphasis on preserving the utility for data mining.

8. CONCLUSION AND FUTURE WORK

In this paper we propose the random space encryption approach to efficient range queries over encrypted data and analyze the unique attacks to this approach. Our approach uses a random space transformation to generate indexable auxiliary data. The auxiliary data is exported to the service provider, indexed and used for processing range queries. We present an efficient server-side two-stage query processing strategy. Experimental results show that this processing strategy is highly efficient. In addition, we analyzed the attacks on encrypted data and queries. Experiments are performed to show the resilience of the encryption to estimation attacks. Note that this attack analysis is just the first step to rigorous analysis of security. We will continue to explore more attacks and formally study the security. As an important extension to our approach, we would like to further investigate how database update operations, such as record deletions, insertions, and updates, affect data utility and security. The goal is to allow the data owner and authorized users update the encrypted data without undermining the security.

9. REFERENCES

[1] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, "Order preserving encryption for numeric data," in *Proceedings of ACM SIGMOD Conference*, 2004.

[2] R. Agrawal and R. Srikant, "Privacy-preserving data mining," in *Proceedings of ACM SIGMOD Conference*. Dallas, Texas: ACM, 2000.

[3] A. Boldyreva, N. Chenette, Y. Lee, and A. O'Neill, "Order preserving symmetric encryption," in *Proceedings of EUROCRYPT conference*, 2009.

[4] D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, "Public-key encryption with keyword search," in *Proceedings of Advances in Cryptology, (EUROCRYPT)*. Springer, 2004.

[5] D. Boneh and B. Waters, "Conjunctive, subset, and range queries on encrypted data," in *the Theory of Cryptography Conference (TCC*. Springer, 2007, pp. 535–554.

[6] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[7] K. Chen and L. Liu, "A random rotation perturbation approach to privacy preserving data classification," in *Proceedings of International Conference on Data Mining (ICDM)*. Houston, TX: IEEE, 2005.

[8] K. Chen and L. Liu, "A survey of multiplicative data perturbation for privacy preserving data mining," *Privacy-Preserving Data Mining: Models and Algorithms, Edited by Charu C. Aggarwal and Philip S. Yu*, 2008.

[9] K. Chen, L. Liu, and G. Sun, "Towards attack-resilient geometric data perturbation," in *SIAM Data Mining Conference*, 2007.

[10] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan, "Private information retrieval," *ACM Computer Survey*, vol. 45, no. 6, pp. 965–981, 1998.

[11] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, "Searchable symmetric encryption: improved definitions and efficient constructions," in *Proceedings of the 13th ACM conference on Computer and communications security*. New York, NY, USA: ACM, 2006, pp. 79–88.

[12] I. Gartner, "Server storage and raid worldwide," *Technical Report*, 1999.

[13] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *STOC '09: Proceedings of the 41st annual ACM symposium on Theory of computing*. New York, NY, USA: ACM, 2009, pp. 169–178.

[14] E.-J. Goh, "Secure indexes," Cryptology ePrint Archive, Report 2003/216, 2003, http://eprint.iacr.org/2003/216/.

[15] O. Goldreich, *Foundations of Cryptography*. Cambridge University Press, 2001.

[16] O. Goldreich and R. Ostrovsky, "Software protection and simulation on oblivious ram," *Journal of the ACM*, vol. 43, pp. 431–473, 1996.

[17] P. Golle, J. Staddon, and B. Waters, "Secure conjunctive keyword search over encrypted data," in *ACNS 04: 2nd International Conference on Applied Cryptography and Network Security*. Springer-Verlag, 2004, pp. 31–45.

[18] S. Guo and X. Wu, "Deriving private information from arbitrarily projected data," in *Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD07)*, Warsaw, Poland, Sept 2007.

[19] A. Guttman, "R-trees: A dynamic index structure for spatial searching," in *SIGMOD'84, Proceedings of Annual Meeting, Boston, Massachusetts, June 18-21, 1984*, B. Yormark, Ed. ACM Press, 1984, pp. 47–57.

[20] H. Hacigumus, B. Iyer, C. Li, and S. Mehrotra, "Executing sql over encrypted data in the database-service-provider model," in *Proceedings of ACM SIGMOD Conference*, 2002.

[21] B. Hore, S. Mehrotra, and G. Tsudik, "A privacy-preserving index for range queries," in *Proceedings of Very Large Databases Conference (VLDB)*, 2004.

[22] Z. Huang, W. Du, and B. Chen, "Deriving private information from randomized data," in *Proceedings of ACM SIGMOD Conference*, 2005.

[23] A. Hyvarinen, J. Karhunen, and E. Oja, *Independent Component Analysis*. Wiley, 2001.

[24] E. Kushilevitz and R. Ostrovsky, "Replication is not needed: Single database, computationally-private information retrieval," in *In Proc. of the 38th Annu. IEEE Symp. on Foundations of Computer Science*, 1997, pp. 364–373.

[25] E. L. Lehmann and G. Casella, *Theory of Point Estimation*. Springer-Verlag, 1998.

[26] K. Liu, C. Giannella, and H. Kargupta, "An attacker's view of distance preserving maps for privacy preserving data mining," in *European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, Berlin, Germany, September 2006.

[27] K. Liu, H. Kargupta, and J. Ryan, "Random projection-based multiplicative data perturbation for privacy preserving distributed data mining," *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, vol. 18, no. 1, pp. 92–106, 2006.

[28] Y. Manolopoulos, A. Nanopoulos, A. Papadopoulos, and Y. Theodoridis, *R-trees: Theory and Applications*. Springer-Verlag, 2005.

[29] E. Shi, J. Bethencourt, T.-H. H. Chan, D. Song, and A. Perrig, "Multi-dimensional range query over encrypted data," in *IEEE Symposium on Security and Privacy*, 2007.

[30] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *IEEE Symposium on Security and Privacy*. Washington, DC, USA: IEEE Computer Society, 2000, p. 44.

[31] H. Wang and L. V. S. Lakshmanan, "Efficient secure query evaluation over encrypted xml databases," in *VLDB '06: Proceedings of the 32nd international conference on Very large data bases*. VLDB Endowment, 2006, pp. 127–138.

[32] P. Williams, R. Sion, and B. Carbunar, "Building castles out of mud: Practical access pattern privacy and correctness on untrusted storage," in *ACM Conference on Computer and Communications Security*, 2008.

[33] W. Wong, D. W. Cheung, B. Kao, and N. Mamoulis, "Secure knn computation on encrypted databases," in *Proceedings of ACM SIGMOD Conference*, 2009.

Appendix: the Attack-Resilient Algorithms

There are four key algorithms in the proposed research $-$ two deployed at the proxy server: (1) Data Encryption and Decryption; (2) Query Transformation and Encryption; and two at the service provider: (3) Data Indexing; (4) Query Processing. We will use the existing multidimensional tree algorithms for data indexing, thus we skip the procedure (3). For simplicity, we process the conditions like $X_i \le a_i$ or $X_i \ge b_i$. The algorithms can be slightly changed to handle other types of conditions.

In Algorithm 1, the key matrix A is generated with the resilience to known input/output attack in mind. First, A is randomly generated with elements drawn from a random real number generator R_A (We used the normal distribution N(0,1) for R_A in experiments). Second, according to the desired noise distribution (i.e., $N(0, \sigma^2)$) for enhancing the resilience to known input/output attack, the algorithm described in Section 5.1 is used to find the last column of A (i.e., A_3) and A_3 replaces the last column of the generated A. The invertibility of A is checked to make sure decryption can be done. The data encryption function extends each original data vector $\mathbf{x}$ to the $(k+2)$ dimensional vector with the homogeneous $(k+1)$-th dimension and the random positive $(k+2)$-th dimension with the random real number generator R_α. Then, it uses an OPE scheme E_o to transform the original k dimensions, followed by the RASP encryption with the key matrix A. Note that the proxy server needs only A and the key for OPE in decryption.

Algorithm 1 Data encryption and decryption algorithms

```
**Encrypt**$(X, R_\alpha, R_A, K_o, \alpha, \beta, \sigma)$
Input: $X$: $k \times n$ data records, $R_\alpha$ and $R_A$ : random real value generators for generating the $(k+2)$-nd dimension (i.e., $\mathbf{v}$) and the invertible $(k+2) \times (k+2)$ matrix $A$ with at least two non-zero values in each row, $K_o$: key for OPE $E_o$, $\alpha$: the upper bound for $\mathbf{v}$, $\beta$:sample size, $\sigma$: noise intensity; Output: the matrix $A$

$A \leftarrow 0$;
**while** $A$ is not invertible **do**
    generate the elements in $A$ with $R_A$;
    $\mathbf{v} = (v_1, \ldots, v_\beta) \leftarrow$ generate $\beta$ random positive values in range (0,$\alpha$) with $R_\alpha$;
    $A_3 \leftarrow 0$;
    **while** $A_3$ contains zero elements **do**
        generate the $(k+2) \times \beta$ noise matrix $\Psi$ use $N(0, \sigma^2)$;
        $A_3 \leftarrow \Psi \mathbf{v}^T / (\mathbf{v}\mathbf{v}^T)$;
    **end while**
    Replace the last column of $A$ with $A_3$;
    Check the invertibility of the matrix $A$;
**end while**
**for** each record $\mathbf{x}$ in $X$ **do**
    $v \leftarrow$ random positive value in range $(0, \alpha)$ with $R_\alpha$
    $\mathbf{y} \leftarrow A(E_o(\mathbf{x}^T, K_o), 1, v)^T$;
    submit $\mathbf{y}$ to the server;
**end for**
return $A$;
```

```
**Decrypt**$(Y, A, K_o)$
Input: $Y$: $k \times n$ matrix, the encrypted records, $A$:the RASP key, $K_o$: the OPE key; Output: the decrypted records $X$

$X \leftarrow A^{-1}Y$;
$X' \leftarrow$ the first $k$ dimensions of $X$;
return OPE decryption $D_o(X')$
```

In Algorithm 2, the query transformation and encryption function takes the $2k$ simple conditions (assume two for each dimension: the upper and lower bounds for each conjunction clause) and the key matrix A as the input, and transforms each condition with the method described in Section 5.2. The MBR is calculated by the following steps: (1) calculate the vertices of the original query range with the dimensional bounds (the (k+1)-th dimension is 1 and the (k+2)-th dimension is bounded by $(0, \alpha]$); (2) transform the vertices with the composite encryption; (3) Find the bounding box of the transformed vertices as the MBR.

Algorithm 2 Query transformation and encryption.

```
**QueryEnc**$(Cond, A)$
Input: Cond: $2k$ simple conditions, 2 for each dimensions. $A$:the key;

**for** each condition $C_i$ in Cond **do**
    $\mathbf{u}_i \leftarrow zeros(k+2, 1)$;
    **if** $C_i$ is like $X_j \le a_j$ **then**
        $u_{ij} \leftarrow 1, u_{i,k+1} \leftarrow -a_j$;
    **end if**
    **if** $C_i$ is like $X_j \ge a_j$ **then**
        $u_{ij} \leftarrow -1, u_{i,k+1} \leftarrow a_j$;
    **end if**
    $\mathbf{w}_i \leftarrow zeros(k+2, 1)$;
    $w_{i,k+2} \leftarrow 1$;
    $\Theta_i \leftarrow (A^{-1})^T \mathbf{u}\mathbf{w}^T A^{-1}$;
**end for**
Use the vertex transformation method to find the MBR of the transformed queries;
submit MBR and the filtering conditions $\{\Theta_i\}$ to the server;
```

In Algorithm 3, the two-stage query processing uses the MBR to find the initial query result and then filters the result with the transformed query conditions $\mathbf{y}^T \Theta_i \mathbf{y} \le 0$, where the matrices $\{\Theta_i\}$ are passed by the client and $\mathbf{y}$ is a record in the first-stage query result.

Algorithm 3 Two-Stage Query Processing.

```
**ProcessQuery**$(MBR, \{\Theta_i\})$
Input: MBR: MBR for the transformed query; $\{\Theta_i\}$:filtering conditions;

$Y \leftarrow$ use the indexing tree to find answers for MBR;
$Y' \leftarrow \emptyset$;
**for** each record $\mathbf{y}_i$ in Y **do**
    success $\leftarrow$ 1
    **for** each condition $\Theta_i$ **do**
        **if** $\mathbf{y}^T \Theta_i \mathbf{y} > 0$ **then**
            success $\leftarrow$ 0;
            break;
        **end if**
    **end for**
    **if** success = 1 **then**
        add $\mathbf{y}_i$ into $Y'$;
    **end if**
**end for**
return $Y'$ to the client;
```