DEIM Forum 2011 C5-6

Optimization Techniques for Range Queries in the Multivalued-Partial Order Preserving Encryption Scheme

Hasan KADHEM[†], Toshiyuki AMAGASA[†,††], and Hiroyuki KITAGAWA[†,††]

† Graduate School of Systems and Information Engineering
†† Center for Computational Sciences
University of Tsukuba
1–1–1 Tennodai, Tsukuba, Ibaraki 305-8573
E-mail: †hsalleh@kde.cst.tsukuba.ac.jp, ††{amagasa,kitagawa}@cs.tsukuba.acjp

Abstract Encryption is a well-studied technique for protecting the privacy of sensitive data. However, encrypting relational databases affects the performance during query processing. Multivalued-Partial Order Preserving Encryption Scheme (MV-POPES) allows privacy-preserving queries over encrypted databases with reasonable overhead and an improved security level. It divides the plaintext domain into many partitions and randomizes them in the encrypted domain. Then, one integer value is encrypted to different multiple values to prevent statistical attacks. At the same time, MV-POPES preserves the order of the integer values within the partitions to allow comparison operations to be directly applied on encrypted data. However, MV-POPES supports range queries at a high overhead. In this paper, we present some optimization techniques to reduce the overhead for range queries in MV-POPES by simplifying the translated condition and controlling the randomness of the encrypted partitions. The basic idea of our approaches is to classify the partitions into many supersets of partitions, then restrict the randomization within each superset. The supersets of partitions are created either based on predefined queries or using binary recursive partition. Experiments show high improvement percentage in performance using the proposed optimization approaches. Also, we study the affect of those optimization techniques on the privacy level of the encrypted data.

Key words Encryption, order-preserving, range queries, binary recursive partition

1. Introduction

Preserving the order of the encrypted values is a useful technique to perform quires over the encrypted database with a reasonable overhead. For instance, given three integers $\{a,b,c\}$, such that $(a<b<c)$, then the encrypted values are $(E^K(a)<E^K(b)<E^K(c))$. Here, $E^K(v)$ denotes ciphertext value of v with encryption key K. The order preserving encryption scheme OPES proposed firstly by [1]. The strength and novelty of OPES is that comparison operations, equality and range queries as well as aggregation queries involving MIN, MAX and COUNT can be evaluated directly on encrypted data, without decryption. Another encryption scheme proposed by [2], [3], where a sequence of polynomial functions is used to encrypt integer values while preserving the order. The decryption is made by solving the inverses of each polynomial function in the sequence in reverse order.

Unfortunately, the previous order preserving encryption (OPE) schemes are not secure against known plaintext attacks and statistical attacks. In those attacks, it is assumed that the attacker has a prior knowledge about plaintext values or statistical information on plaintext domain. Here, the attacker who has access to the encrypted values and has knowledge about the plaintext can map both the plaintext and the encrypted values and make use of them to obtain the key. This is because the OPE schemes preserve the order of all integers in the domain, so the order of encrypted values is exactly the order of plaintext values (we called those schemes as full OPE).

In [4], we proposed a database encryption scheme called MV-POPES (Multivalued-Partial Order Preserving Encryption Scheme), which divides the plaintext domain into many partitions and randomizes them in the encrypted domain. It allows one integer to be encrypted to many values using the same encryption key while preserving the order of the integer values within the partitions. Here, we can still get benefit from the partial order preserving in the encrypted data to perform queries directly at the server without decrypting data. At the same time, we can prevent attackers from inferring individual information from the encrypted database even if they have statistical and special knowledge about the plaintext database. The reason is that the encrypted values are totally in different order compared with the plaintext values. The results from an implementation of MV-POPES show that security for sensitive data can be achieved with reasonable overhead in performing different types of queries. However, MV-POPES supports range queries over encrypted database at a high overhead compared with the full OPE schemes. The overhead of range queries in MV-POPES is mainly dependent on the number of partitions. Large number of partitions will lead to high overhead on performing range queries over encrypted database. That is because the condition in the WHERE clause becomes complex with many sub conditions connected with boolean connectors (AND and OR).

In this paper, we present some optimization techniques to reduce the overhead for range queries in MV-POPES by simplifying the translated condition and controlling the randomness of the encrypted partitions. The basic idea of our approaches is to classify the partitions into many supersets of partitions, then we restrict the randomization within each superset. The supersets of partitions are created either based on predefined queries or using Binary Recursive Partition (BRP). Experiments show high improvement percentage in performance using the proposed optimization approaches. Also, we study the affect of those optimization techniques on privacy level of the encrypted data.

1.1 Organization of the Paper

The rest of paper is organized as follows. We first discuss related work in Section 2. Section 3 gives a brief overview of the MV-POPES. Section 4 describes merging conditions technique. Section 4 discusses the multilevel partitioning based on predefined queries and binary recursive partition. Section 6 reports the experimental results. Section 7 analyzes the security effects of the optimization techniques. Section 8 introduces the privacy-performance trade-off. We conclude with a summary and directions for future work in Section 9.

2. Related Work

Many full OPE schemes [1]∼[3], [5], [6] have been proposed, but no work discussed the security of the encryption schemes against known plaintext attack and statistical attack. The authors in [7] proposed a new encryption scheme (Chaotic Order Preserving Encryption (COPE)). COPE [7] hides the order of the encrypted values by changing the order of buckets in the plaintext domain. It is secure against known plaintext attack. However, COPE can be used just on trusted server where the encryption keys are used to perform many queries such as join and range queries. The overhead of range queries over encrypted database is much higher than the overhead of range queries over plaintext database. In addition, it uses many keys to change the order of buckets and in some cases that may lead to have duplicated values. Another drawback in COPE is the encryption and decryption cost. That is because of the computation complexity to randomize the buckets and assign the correct order within each bucket.

The bucketing approach [8]∼[12] is closely related to our scheme in sense of dividing the plaintext domain into many partitions (buckets). The encrypted database in the bucketing approach is augmented with additional information (the index of attributes), thereby allowing query processing to some extent at the server without endangering data privacy. The encrypted database in the bucketing approach contains etuples (the encrypted tuples) and corresponding bucket-ids (where many plaintext values are indexed to same bucket-id). In this scheme, executing a query over the encrypted database is based on the index of attributes. The result of this query is a superset of records containing false positive tuples. These false hits must be removed in a post filtering process after etuples returned by the query are decrypted. Because only the bucket id is used in a join operation, filtering can be complex, especially when random mapping is used to assign bucket ids rather than order preserving mapping. The number of false positive records depends on the number of buckets involved. Using a small number of buckets will hide the real values within the bucket index, but the filtering overhead can become excessive. On the other hand, a large number of buckets will reduce the filtering overhead, but the scheme will be vulnerable to estimation exposure. In bucketing, the projection operation is not implemented over the encrypted database, because a row level encryption is used. In addition, updating attributes in the bucketing approach requires that two attributes be updated, the bucket-id and the etuple. This means that all attributes in the row must be re-encrypted, thereby increasing overhead for the update query.

Many works [13]∼[16] studied the challenges in balancing query efficiency and data secrecy using bucketing based encryption. The basic idea of these researches is to design a bucketization algorithm based on predefined queries that minimizes the false positive tuples while ensuring the privacy of data. It assumed that the data and queries are not frequently updated. While in real world most databases systems are frequently uploaded and updated.

PID	EPID	PREV	F	L	NEXT
1	3	80	1	20	81
2	1	-	21	40	61
3	5	100	41	60	101
4	2	40	61	80	1
5	4	20	81	100	41

Fig. 1 partitioning metadata.

3. Preliminaries

3.1 An Overview of MV-POPES

We begin with the brief overview of the MV-POPES. When encrypting plaintext values in a column having values in the range $[D_{min}, D_{max}]$, first, we divide the domain into n partitions and assign for each partition a random number from 1 to n. This number will be the order of partitions in the encrypted domain. We change the order of partitions to hide the original order of plaintext values. Then, boundaries are generated for all integers in all partitions using an order preserving function. The order is preserved within the partition to be able to evaluate queries efficiently on encrypted database. The generated boundaries identify the intervals. For instance, interval I_i is identified by $[B_i, B_{i+1})$. We then generate the encrypted values for integer i as random values from the interval I_i, so one plaintext value is encrypted to many different values. This will change the frequencies of the plaintext values to prevent the encrypted database against statistical attack.

3.2 Partitioning and Metadata

Here, we explain the partitioning function for each attribute's domain and what is stored in the metadata for each domain. We first divide the plaintext domain of values $[D_{min}, D_{max}]$ into partitions $\{p_1, \ldots, p_n\}$, such that these partitions cover the whole domain and there is no overlap between them. Then, we assign for each partition a unique random number in the range of $[1,n]$. This number is the new order of partitions in the encrypted domain.

As an example, Figure 1 shows the partitions metadata for the domain $[1,100]$. The domain is divided into 5 partitions: $[1,20],[21,40],[41,60],[61,80],[81,100]$. (F), (L) are the first and the last number in the partition. The partition identifier (PID) represents the original order of partitions in plaintext domain. The encrypted partition identifier $(EPID)$ represents the order of partition in the encrypted domain. $(PREV)$ and $(NEXT)$ are the previous and the next number in the encrypted domain for a partition. For instance, the partition $[61,80]$ where $PID{=}4$, and $EPID{=}2$, the $(PREV)$ will be the last number for the partition with $EPID{=}1$ (which is 40), and the $(NEXT)$ will be the first number for the partition with $EPID{=}3$ (which is 1).

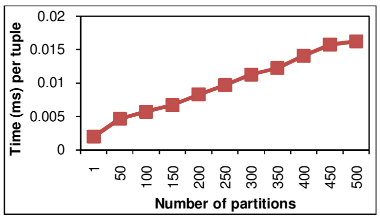


Fig. 2 Range query overhead in MV-POPES.

3.3 Partition Identification Functions

The partition identification functions will be used to translate conditions that contain comparison operations. Let A be an attribute, v be a value in the domain and i be a partition identifier. Table 1 shows the partition identification functions. Using the running example, $PID_A^{<}(50){=}\{1,2\}$ and $PID_A^{>}(50){=}\{4,5\}$.

3.4 Translation of Range Query

In this section, we explain the translation of range query using the condition **Attribute** $<$ **Value** as an example. Since the MV-POPES preserves the order of the encrypted values within the partition ($v_i{<}v_j \rightarrow E^K(v_i){<}E^K(v_j)$, v_i and v_j belong to the same PID), the translation of $(A{<}v)$ is as follows:

$$(A^E{<}B_v \ \wedge \ A^E{\geq}B_{F(PID_A(v))}) \ \vee$$

$$\bigvee_{i \in PID_A^{<}(v)} (A^E{\geq}B_{F(i)} \ \wedge \ A^E{>}B_{NEXT(i)})$$

where $F(i)$ is the first integer in the partition i, and $NEXT(i)$ is the next integer of the partition i. Simply, the result contains all encrypted values that are less than the left boundary (B_v) of the interval (I_v) within the partition that contains v. In addition, all partitions whose PID are less than the partition of v are included in the result. For example, the translation for the condition $(A{<}55)$ is:

$$(A^E{<}B_{55} \wedge A^E{\geq}B_{50}) \vee (A^E{\geq}B_1 \wedge A^E{<}B_{81}) \vee (A^E{\geq}B_{21} \wedge A^E{<}B_{61})$$

Figure 2 shows the range query (selectivity=50%) execution times on MV-POPES over 10^5 encrypted tuples that picked randomly from a uniform distribution between 1 and 10^5. We can clearly see that the overhead is sharply increasing with increasing the number of partitions, because the condition becomes more complicated. Thus, some optimization techniques are needed to reduce the overhead for range

Table 1 Partition Identification Functions.

PID_A	set of PID for attribute A
$PID_A(v)$	PID to which value v belongs in the domain of A
$PID_A^{<}(v)$	set of PID for attribute A that are less than the partition that contains v
$PID_A^{>}(v)$	set of PID for attribute A that are greater than the partition that contains v
$PID_A^{<i}$	set of PID for attribute A that are less than i
$PID_A^{>i}$	set of PID for attribute A that are greater than i

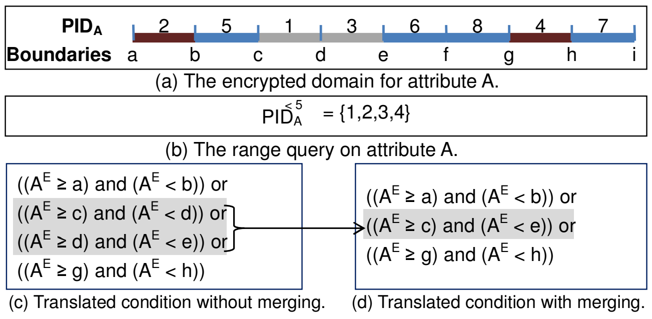

PID_A 2 5 1 3 6 8 4 7
Boundaries a b c d e f g h i
(a) The encrypted domain for attribute A.
PID_A^{<5} = {1,2,3,4}
(b) The range query on attribute A.
((A^E ≥ a) and (A^E < b)) or
((A^E ≥ c) and (A^E < d)) or
((A^E ≥ d) and (A^E < e)) or
((A^E ≥ g) and (A^E < h))
((A^E ≥ a) and (A^E < b)) or
((A^E ≥ c) and (A^E < e)) or
((A^E ≥ g) and (A^E < h))
(c) Translated condition without merging.
(d) Translated condition with merging.

Fig. 3 Example of merging conditions in MV-POPES.

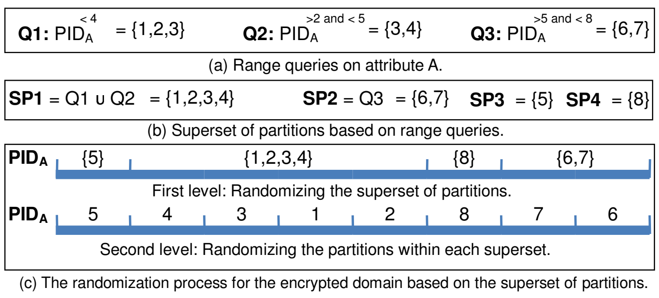

Q1: PID_A^{<4} = {1,2,3}
Q2: PID_A^{>2 and <5} = {3,4}
Q3: PID_A^{>5 and <8} = {6,7}
(a) Range queries on attribute A.
SP1 = Q1 ∪ Q2 = {1,2,3,4}
SP2 = Q3 = {6,7}
SP3 = {5}
SP4 = {8}
(b) Superset of partitions based on range queries.
PID_A {5} {1,2,3,4} {8} {6,7}
First level: Randomizing the superset of partitions.
PID_A 5 4 3 1 2 8 7 6
Second level: Randomizing the partitions within each superset.
(c) The randomization process for the encrypted domain based on the superset of partitions.

Fig. 4 Example of randomizing the partitions based on predefined queries.

queries in MV-POPES by simplifying the translated condition.

4. Merging Conditions

The first solution to reduce the overhead of range queries in MV-POPES is to merge two or more conditions into one condition. Thus, the query condition in the WHERE clause will be less complex. This can be done in the query translation by searching for neighbor encrypted partitions that are included in the range query. For instance, the original condition shown in Figure 3 (c) over encrypted domain (Figure 3 (a)) consists of 8 sub conditions that specify the first and the last values for each encrypted partition. Knowing that partitions with ids 1 and 3 are adjusted to each other in the encrypted domain, the four sub conditions related to those partitions are simplified to two sub conditions (Figure 3 (d)). Using partitioning metadata, the neighbors' partitions in the encrypted domain can be discovered easily and then merged in the translated condition.

The improvement in performance by merging conditions is based on the number of neighbors' partitions in the range query. The experiments show that the maximum improvement percentage by merging conditions is just 5%. That is because the partitions are randomized over the whole encrypted domain. Thus, the probability to have neighbors' partitions in the encrypted domain close to the order of partitions in the plaintext domain is really small.

5. Multilevel Partitioning

Merging conditions is not enough to improve the performance of range queries in MV-POPES because it is based on the randomness of the encrypted partitions. Thus, in order to reduce the overhead, we need to control the randomness of the encrypted partitions.

5.1 Predefined Queries

The first approach to control the randomness of the encrypted partitions is based on a set of predefined range queries. Given a set of range queries $Q=\{q_1,\ldots,q_k\}$ on an attribute A, such that $q_i=[l,h]$ where $l<h$ and $l,h\in PID_A$. The supersets of the partitions are created by merging all queries that have a partition in common. So, for queries q_i,q_j, such that $q_i\cap q_j\neq\phi$, a superset of partitions $SP=q_i\cup q_j$ is created. A separate superset is created for each partition that is not included in any queries. Then, the supersets of partitions are randomized at a first level of randomization. The second level of randomization is done within each superset by randomizing the set of partition for each query. At the final level, the partitions in each set are randomized restrictly within each set of partitions. Figure 4 shows an example of controlling the randomness of 8 encrypted partitions based on 3 range queries. Figure 4 (a) shows the predefined range queries on attribute A, Figure 4 (b) shows the supersets of partitions based on the predefined range queries and Figure 4 (c) shows the process of randomizing the partitions in the encrypted domain.

Using such a technique, we can clearly see that the translated condition will be approximately as simple as the condition on the plaintext domain. That is because most of the partitions that are included in a range query q_i will be neighbors in the encrypted domain even if they are in a different order. So, merging conditions technique now is effective. The overhead of performing range queries over encrypted domain will be much closer to the overhead on the plaintext domain. However, this technique is complicated to implement. In addition, when the set of range queries changes, all the data in the attribute need to be re-encrypted based on the new supersets of partitions. Thus, this technique is not good when the data and queries are changing frequently.

5.2 Binary Recursive Partitioning (BRP)

The second approach to control the randomness of the encrypted partitions is based on the binary recursive partitioning BRP . BRP is an iterative process of splitting the data into partitions, and then splitting it up further on each of the branches. Here, BRP is used to control the randomness of the encrypted partitions by dividing the partitions into homogeneous subgroups or subsets of partitions. Using BRP, the partitions are divided recursively into two subsets till they reach the predefined level or predefined rules. There are many types of rules that decide the splitting position. For

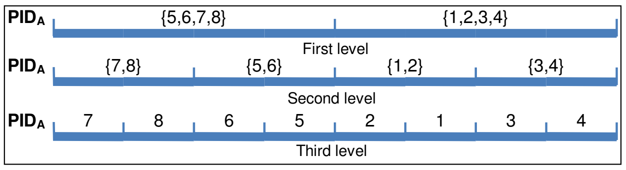


Fig. 5 Example of randomizing the partitions based on BRP.

simplicity of analysis, we will restrict our attention to splitting the subsets of partitions based on equi-width method. Given a set of partition $PID_A=\{1,...,n\}$ for attribute A and a predefined level pl to stop the partitioning process. At each level in the partitioning process, and for each subset of partitions that consists of $[l,h]$ where $l<h$ and $(l,h \in PID_A)$, the splitting point is $\frac{l+h}{2}$. For each level, the new subsets are randomized within the superset position. At the last level, the partitions are randomized within each subset's position they belong. For instance, Figure 5 shows an example for 3 levels of binary recursive partition applied on 8 partitions.

6. Experiments

We have conducted experiments to examine the validity and the effectiveness of the BRP for performing range queries in MV-POPES. We generated $100{,}000$ records picked randomly from a uniform distribution using $[1,100000]$ as an input domain. The evaluations were performed on a various number of partitions ($\{50,100,...,500\}$) using different BRP levels ($\{1,2,3,4\}$). The queries used in those experiments have different selectivity $\{10\%,20\%,30\%,40\%,50\%\}$. Figure 6 (a) shows the execution time for the set of range queries performed on non-optimized partitions. The overhead is sharply increasing with increasing the number of partitions. As we discussed before, that is because the condition in the WHERE clause becomes more complicated using large number of partitions.

Figure 6 (b, c, d, e) shows the execution time of the same set of range queries performed on an optimized encrypted domain (using different BRP levels). The figures show that the overhead in the optimized encrypted attribute is less than the non-optimized domain. Also, we notice from those figures that the overhead is decreased by increasing the level in BRP. That is because the translated condition can be simplified more in advance BRP level by merging the sub conditions for neighbor partitions. Figures 6 (b, c, d) show that the high selectivity queries (50%) take less time than the low selectivity queries (10%), especially with the large number of partitions ($[400-500]$). The reason behind that is that high selectivity queries consist of a large number of partitions, thus the probability for merging partitions into one condition is high. On the other hand, in the case of low selectivity queries such as 10%, the queries consist of small number of partitions; so the probability for merging neighbor partitions is small.

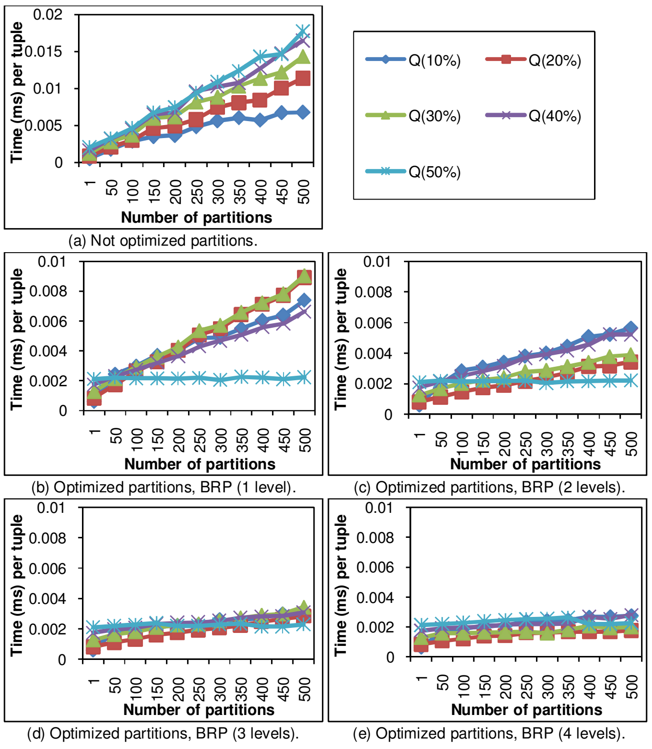


Fig. 6 Range queries on non-optimized and optimized encrypted partitions (using BRP).

Thus, the overhead becomes higher when large number of partitions is used to encrypt the plaintext domain. Overall, in advanced BRP level, the overhead using large number of partitions (MV-POPES) is slightly more than the overhead using one partition (full OPE scheme).

Figure 7 shows the improvement percentage in the execution time for range queries using optimized encrypted partitions (different BRP levels) compared with non-optimized domain. The figure shows that the improvement is increasing with increasing the number of partitions. Also, we can notice that the improvement percentage is increasing with increasing in level for BRP. The improvement reaches approximately 92% in case of 50% selectivity with 500 partitions in the case of 4 BRP levels. The overhead in optimized partitions is slightly increasing using large number of partitions, while in non-optimized partitions the overhead is sharply increasing by using large number of partitions. Thus, the improvement using large number of partitions such as 500 is higher than the improvement using smaller number of partitions such as 50.

7. Security Analysis

The techniques proposed above to control the randomness of the encrypted partitions affect the privacy level of the encrypted domain. That is because those techniques restrict the position probability for a partition in the encrypted domain. Without optimization techniques the probability for

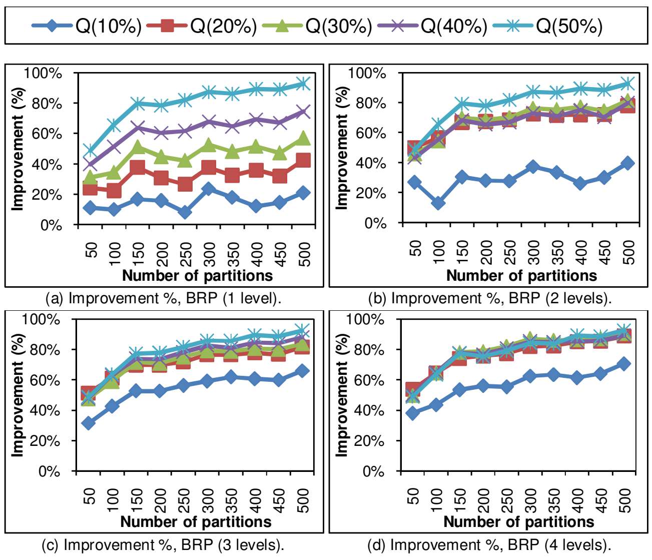


Fig. 7 Improvement percentage in execution time for range queries using different BRP levels.

a partition to be in a position in the encrypted domain is $\frac{1}{n}$, where n is the number of partitions. On the other hand, using BRP, the probability for a partition to be in a position in the encrypted domain just after the first level is $\frac{1}{(n/2)}$, this number becomes smaller in advanced level of BRP. In this section, we analyze the privacy of the encrypted partitions in both cases; with optimization technique (BRP) and without optimization. We study the **Entropy (H)** and **Variance (V)** of the distribution of encrypted partitions as measures of privacy. We base our choice of entropy and variance on the security definitions presented in [4]. The basic idea to ensure privacy of the encrypted domain is to have as much as possible the encrypted domain in a different order compared with the plaintext domain.

7.1 Entropy

Entropy is a measure of disorder, or more precisely unpredictability. It is well-known that entropy of a random variable X is a measure of its uncertainty [17] . Using the entropy of the encrypted domain we can see how encrypted partitions spread over the encrypted domain. Generally, entropy of a random variable X taking values $x_i{=}1,\ldots,n$ with corresponding probabilities p_i ,$i{=}1,\ldots,n$ is given by:

$$Entropy(X){=}H(X){=}-\sum_{i=1}^{n} p_i \times log_2(p_i)$$

Given the partitions in the plaintext domain $PID{=}\{1,\ldots,n\}$, the corresponding partitions in the encrypted domain $EPID{=}\{\forall i{\in}PID,\exists! j{\in}EPID,(1{\le}j{\le}n)\}$ and the corresponding probabilities p_i , the entropy of the encrypted partitions $H(EPID)$ without controlling the randomness can be written as:

$$\begin{aligned}
H(EPID) &= -\sum_{i=1}^{n} p_i \times log_2(p_i) \\
&\quad Since\ p_i\ for\ all\ partitions\ (p_i{=}\tfrac{1}{n}) \\
&= -\sum_{i=1}^{n} \tfrac{1}{n} \times log_2(\tfrac{1}{n}) \\
&= -\ n \times \tfrac{1}{n} \times log_2(\tfrac{1}{n}) \\
&= log_2(n)
\end{aligned}$$

The entropy in the encrypted domain without optimization depends on the number of partitions. A large number of partitions has more entropy than a small number of partitions which means it is more secure. Using multilevel partitioning to control the randomness of the encrypted partitions, the entropy depends on the number of super partitions and the number of partitions on each super partition. That is because the entropy for a set of partitions $(SP_i,\ i{=}\{1,\ldots,m\})$ is the summation of entropy in each super partition $H(SP_i)$ multiplied by the probability p_{SP_i} for each super partition (p_{SP_i} is the number of elements in each partition divided by the total number of elements). Given the partitions in the plaintext domain $PID{=}\{1,\ldots,n\}$ and super partitions $SP_i,\ i{=}1,\ldots,m$, the entropy of the encrypted partitions $H(EPID)$ can be written as:

$$H(EPID)\ =\ \sum_{i=1}^{m} \frac{|SP_i|}{n} \times H(SP_i)$$

in which $|SP_i|$ is the number of partitions in SP_i. The entropy for the encrypted domain $H(EPID)$ in this case will be less than the entropy without optimization. As an example, we analyze the entropy of the optimized encrypted domain using binary recursive partition. Given the partitions in the plaintext domain $PID{=}\{1,\ldots,n\}$, the level pl of the BRP and super partitions $SP_i,\ i{=}1,\ldots,2^{pl}$, the entropy of the encrypted partitions $H(EPID)$ using BRP can be written as:

$$\begin{aligned}
H(EPID) &= \sum_{i=1}^{2^{pl}} p_{SP_i} \times H(SP_i) \\
&\quad Since\ SP_i\ has\ same\ size\ (p_{SP_i}{=}\tfrac{1}{2^{pl}}) \\
&= \sum_{i=1}^{2^{pl}} \tfrac{1}{2^{pl}} \times H(SP_i) \\
&= 2^{pl}\ \times \tfrac{1}{2^{pl}} \times H(SP_i) \\
&= H(SP_i) \\
&= log_2(\tfrac{n}{2^{pl}})
\end{aligned}$$

Here, the entropy is the summation of entropy for each super partition $H(SP_i)$ multiplied by the probability for each super partition p_{SP_i}. Since all super partitions have same size and same probability, the entropy of the encrypted domain with BRP after pl levels is $log_2(\frac{n}{2^{pl}})$. Thus, the entropy will be smaller by increasing the level in BRP. The privacy also decreases in an advanced level of BRP. That is because the randomness in an advanced level of BRP will be less effective to hide the order of the encrypted partitions. Figure 8 shows the entropy of the encrypted domain in both cases; (a) without optimization and (b) optimized domain (BRP) using different levels.

7.2 Variance

The variance and the closely-related standard deviation are measures of how spread out a distribution is. In other words, they are measures of variability. In this section, we study two types of variance; **Horizontal Variance (HV)**

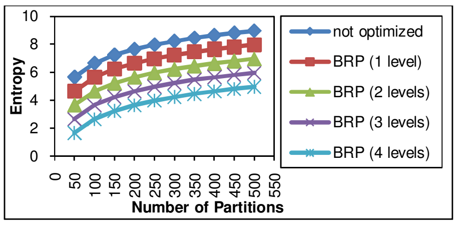


Fig. 8 Entropy of the encrypted domain.

and **Vertical Variance (VV)**. The horizontal variance used to measure the difference of orders between plaintext and encrypted domain. While vertical variance used to measure the partial order in the encrypted domain. Given the partitions in the plaintext domain $PID=\{1,\dots,n\}$ and the corresponding partitions in the encrypted domain $EPID=\{\forall i\in PID,\exists! j\in EPID,(1\leq j\leq n)\}$, the horizontal variance HV can be written as:

$$HV=\frac{1}{n}\sum_{i=1}^{n}\quad PID(i)-EPID(i)\quad -HM\quad{}^{2}$$

where HM is the horizontal mean and can be calculated as follows:

$$HM=\frac{1}{n}\sum_{i=1}^{n}\quad PID(i)-EPID(i)$$

Note that the horizontal mean HM will be 0 for all permutation. Thus, the horizontal variance can be rewritten as:

$$HV=\frac{1}{n}\sum_{i=1}^{n}\quad PID(i)-EPID(i)\quad{}^{2}$$

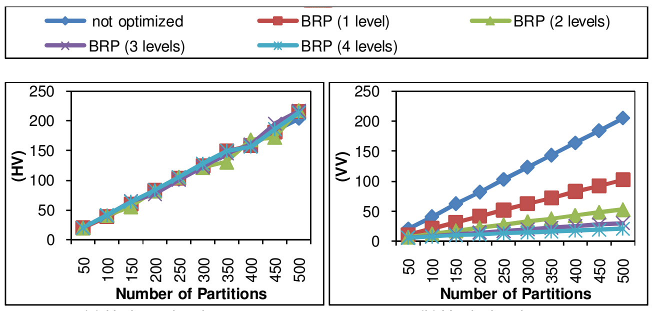
The horizontal variance describes how far the order of the encrypted partitions lies from the order of partitions in the plaintext domain. The minimum value for HV is 0 when both PID and $EPID$ have same order. High value for HV means high difference of order between partitions in plaintext domain and partitions in encrypted domain, which leads to high privacy level. However, horizontal variance dose not measure the partial order in the encrypted domain. We can have high HV when encrypted partitions are in descending order which is not secure at all. Thus, we need another measure to describe the order in the encrypted domain. Here, we calculate the variance within the encrypted domain (vertical variance) by taking the difference between each two sequenced partitions. The vertical variance VV for the encrypted partitions $EPID$ is given as follows:

$$VV=\frac{1}{n-1}\sum_{i=1}^{n-1}\quad EPID(i)-EPID(i+1)\quad -VM\quad{}^{2}$$

where VM is the vertical mean and can be calculated as follows:

$$VM=\frac{1}{n-1}\sum_{i=1}^{n-1}\quad EPID(i)-EPID(i+1)$$

The vertical variance describes the degree of partial order in the encrypted domain. The minimum value for VV is 0 when the encrypted partitions are totally ordered either in ascending or descending order. High value for VV means the degree of partial ordering in the encrypted domain is low, which leads to high privacy level.



(a) Horizontal variance. (b) Vertical variance.

Fig. 9 Horizontal and vertical variance in non-optimized and optimized (BRP) encrypted domain.

Figure 9 shows the effect of BRP (levels=$\{1,2,3,4\}$) on horizontal and vertical variance of encrypted partitions in MV-POPES. The horizontal variance (Figure 9 (a)) using different BRP levels is approximately same as the HV in non-optimized encrypted domain. That is because the encrypted partitions can be in different order compared with the original order in the plaintext domain, even when we restrict the randomness options by using different BRP levels. Figure 9 (b) shows that the vertical variance is reduced because of BRP. That is because the partial order in the encrypted domain using BRP is much more than the non-optimized domain. Also, we can clearly see from the figure that the vertical variance is decreasing by increasing the level of BRP. The results are expectable since the process of controlling the randomness for encrypted partitions in BRP keeps the encrypted partitions somehow ordered within each superset of partitions. Thus, increasing the level of BRP will increase the partial order level in the encrypted domain, and that helps to perform range queries over the encrypted domain efficiently. However, increasing the level of BRP will decrease the privacy level.

8. The Privacy-Performance Trade-off

The optimization techniques proposed in this paper lead to better performance for range queries in MV-POPES, for a given number of partitions. However, those techniques also lead to reducing the level of privacy (that is the encrypted partitions might not have a large enough entropy and variance). The research issue here is how to re-randomize the encrypted partitions, starting with the optimized performance for range queries and allowing a bounded amount of performance degradation, in order to maximize the two measures related to the privacy (entropy and variance) simultaneously (Figure 10). We formalize the problem being addressed below:

Trade-off Problem: Given a domain $D=[D_{min},D_{max}]$ for attribute A, and an initial set of encrypted parti-

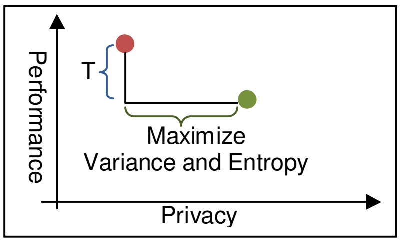


Fig. 10 Privacy-performance trade-off.

tions $\{EPID_1,...,EPID_n\}$ in an optimized order, re-randomize and/or divide the encrypted partitions into new order $\{C\ EPID_1,\ ...,\ C\ EPID_w\}$ such that no more than a factor T of performance degradation is introduced and the entropy and vertical variance for the encrypted partitions are simultaneously maximized.

This problem required advanced analysis and algorithms to maximize the entropy and variance within T factor of performance degradation by swapping and subdividing the partitions. We will study this problem in future work.

9. Conclusion and Future Work

MV-POPES supports range queries at a high overhead compared with the full OPE schemes. In this paper, we presented some optimization techniques to reduce the overhead for range queries in the MV-POPES by simplifying the translated condition and controlling the randomness of the encrypted partitions. The basic idea of our approaches is to classify the partitions into many supersets of partitions, then we restrict the randomization of the partitions within each superset. The supersets of partitions are created either based on predefined queries or using binary recursive partition BRP. Experiments show high improvement percentage in performance using the proposed optimization approaches. However, those techniques also lead to reducing the level of privacy. In the future, we will study the privacy-performance trade-off in the MV-POPES.

Acknowledgments

This work was supported by MEXT Grant-in-Aid for Scientific Research on Innovative Areas (#21013004) and JSPS Grant-in-Aid for Young Scientists (B) (#21700093).

References

[1] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, “Order preserving encryption for numeric data,” SIGMOD ’04: Proceedings of the 2004 ACM SIGMOD international conference on Management of data, New York, NY, USA, pp.563–574, ACM, 2004.

[2] S.S. Chung and G. Ozsoyoglu, “Anti-Tamper databases: Processing aggregate queries over encrypted databases,” ICDEW ’06: Proceedings of International Conference on Data Engineering Workshops, Washington, DC, USA, p.98, IEEE Computer Society, 2006.

[3] G. Ozsoyoglu, D.A. Singer, and S.S. Chung, “Anti-tamper databases: Querying encrypted databases,” 17th Annual IFIP Working Conference on Database and Applications Security, Estes Park, pp.4–6, 2003.

[4] H. Kadhem, T. Amagasa, and H. Kitagawa, “A secure and efficient order preserving encryption scheme for relational databases,” KMIS 2010: Proc. of International Conference on Knowledge Management and Information Sharing, pp.25–35, 2010.

[5] H. Wang and L.V.S. Lakshmanan, “Efficient secure query evaluation over encrypted XML databases,” VLDB ’06: Proceedings of the 32nd international conference on Very large databases, pp.127–138, VLDB Endowment, 2006.

[6] H. Kadhem, T. Amagasa, and H. Kitagawa, “Mv-opes: Multivalued-order preserving encryption scheme: A novel scheme for encrypting integer value to many different values,” IEICE Transactions, vol.93-D, no.9, pp.2520–2533, 2010.

[7] S. Lee, T. Park, D. Lee, T. Nam, and S. Kim, “Chaotic order preserving encryption for efficient and secure queries on databases,” IEICE Transactions on Information and Systems, vol.92, pp.2207–2217, 2009.

[8] H. Hacigümüş, B. Iyer, C. Li, and S. Mehrotra, “Executing SQL over encrypted data in the database-service-provider model,” SIGMOD ’02: Proceedings of the 2002 ACM SIGMOD international conference on Management of data, New York, NY, USA, pp.216–227, ACM, 2002.

[9] E. Damiani, S.D.C. di Vimercati, M. Finetti, S. Paraboschi, P. Samarati, and S. Jajodia, “Implementation of a storage mechanism for untrusted dbmss,” Security in Storage Workshop, International IEEE, vol.0, p.38, 2003.

[10] H. Hacigümüs, B.R. Iyer, and S. Mehrotra, “Ensuring the integrity of encrypted databases in the database-as-a-service model,” DBSec 2003: Seventeenth Annual Working Conference on Data and Application Security, pp.61–74, Kluwer, 2003.

[11] E. Damiani, S.D.C. di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati, “Metadata management in outsourced encrypted databases,” SDM 2005: Proceedings of the Second VLDB Workshop on Secure Data Management, Lecture Notes in Computer Science, pp.16–32, Springer, 2005.

[12] A. Ceselli, E. Damiani, S.D.C.D. Vimercati, S. Jajodia, S. Paraboschi, and P. Samarati, “Modeling and assessing inference exposure in encrypted databases,” ACM Trans. Inf. Syst. Secur., vol.8, no.1, pp.119–152, 2005.

[13] Y. Tang, J. Yun, and Q. Zhou, “A multi-agent based method for reconstructing buckets in encrypted databases,” Proceedings of the IEEE/WIC/ACM international conference on Intelligent Agent Technology, IAT ’06, Washington, DC, USA, pp.564–570, IEEE Computer Society, 2006.

[14] J. Li and E. Omiecinski, “Efficiency and security trade-off in supporting range queries on encrypted databases,” DBSec 2005: Proceedings of 19th Annual IFIP WG 11.3 Working Conference on Data and Applications Security, Lecture Notes in Computer Science, vol.3654, pp.69–83, Springer, 2005.

[15] B. Hore, S. Mehrotra, and G. Tsudik, “A privacy-preserving index for range queries,” Proceedings of the Thirtieth international conference on Very large data bases - Volume 30, VLDB ’04, pp.720–731, VLDB Endowment, 2004.

[16] E. Damiani, S.D.C. Vimercati, S. Jajodia, S. Paraboschi, and P. Samarati, “Balancing confidentiality and efficiency in untrusted relational dbmss,” Proceedings of the 10th ACM conference on Computer and communications security, CCS ’03, New York, NY, USA, pp.93–102, ACM, 2003.

[17] T.M. Cover and J.A. Thomas, Elements of information theory, Wiley-Interscience, New York, NY, USA, 1991.