Hindawi
Security and Communication Networks
Volume 2021, Article ID 1160305, 8 pages
https://doi.org/10.1155/2021/1160305

Research Article

Practical Frequency-Hiding Order-Preserving Encryption with Improved Update

JiHye Yang and Kee Sung Kim

Dept. of Computer Software, Daegu Catholic University, Daegu, Republic of Korea

Correspondence should be addressed to Kee Sung Kim; kee21@cu.ac.kr

Received 17 August 2021; Revised 28 October 2021; Accepted 15 November 2021; Published 6 December 2021

Academic Editor: Spiridon Bakiras

Order-preserving encryption (OPE) that preserves the numerical ordering of plaintexts is one of the promising solutions of cloud security. In 2013, an ideally secure OPE, which reveals no additional information except for the order of underlying plaintexts, was proposed, along with the notion (mutable encryption) that ciphertexts can be changed. Unfortunately, even the ideally secure OPE can be vulnerable by inferring the underlying frequency of repeated plaintexts. To solve this problem, in 2015, Kerschbaum designed a frequency-hiding OPE (FH-OPE) scheme based on the notion of a randomized order under the strengthened security model. Later, Maffei et al. has shown that Kerschbaum's model is imprecise, which means no such OPE scheme can exist. Moreover, they provided a new FH-OPE scheme under the corrected security model. However, their scheme requires the order information of all the encrypted plaintexts as an input; therefore, it causes relatively high overhead during encryption. In this work, we propose a more efficient FH-OPE based on Maffei et al.' s security model and also present an improved update algorithm suitable for duplicate plaintexts.

1. Introduction

Cloud storage has become a common practice in recent years, but it still has privacy concerns with respect to the service provider hosting the data. In these data-outsourcing scenarios, encryption is one of the most reliable solutions. However, the existing normal encryption schemes have limitations; for instance, it is impossible to perform operations, e.g., range query, on encrypted data. To perform such operations, the client has to download all the encrypted data and decrypt them. To overcome these limitations, few solutions have been proposed by slightly weakening the security of the normal encryption schemes. Order-preserving encryption is one of the promising solutions and allows a client to perform efficient range queries on the encrypted data because it maintains the ordering of plaintexts in ciphertexts.

1.1. Related Works. The first concept of order-preserving encryption was introduced by Agrawal et al. [1]. In 2009, Boldyreva et al. [2] presented the first formal security notion of OPE, which is called indistinguishability against ordered chosen plaintext attacks (IND-OCPA). Moreover, they showed that any stateless OPE cannot guarantee the IND-OCPA security unless the ciphertext space is exponentially large in the plaintext space. They also presented the weaker security notion, which is known as pseudorandom order-preserving function advantage under chosen ciphertext attacks (POPF-CCA). However, this security model does not precisely quantify the leakage information of plaintexts. Later, Boldyreva et al. [3] and Xiao and Yen [4] showed that ciphertexts of [2] scheme leak approximately the first half bits of the underlying plaintexts. Yum et al. [5] improved Boldyreva's construction by extending their work to non-uniformly distributed plaintexts but still remained in the same security level of random order-preserving functions. Subsequently, few OPE schemes [6–15] that provide no formal security proof were proposed, but rather they provided an ad hoc security analysis.

Recently, some ideally secure (IND-OCPA secure) OPE schemes [16–20], which are stateful or interactive, have been proposed. Popa et al. [18] developed an interactive model for

clients and servers as a two-party protocol. The client encrypts plaintexts using a deterministic OPE algorithm and sends them to the server that maintains a search tree where ciphertexts are stored. When the client wants to perform range queries on the encrypted data, the server exploits the search tree. Moreover, they presented a notion of mutable encryption, which means that ciphertexts can be updated to achieve the IND-OCPA security. Their interacting scheme requires a large amount of communication. In 2014, Kerschbaum and Schröpfer [19] presented a revised ideally secure OPE scheme where the client stores the search tree. This approach makes it possible for their scheme to incur lower communication cost than that proposed by [18].

To solve the problem of deterministic OPE [1, 3, 18, 19, 21, 22] that are vulnerable to frequency analysis, sorting, and cumulative attack [23], Kerschbaum [16] presented a new frequency-hiding OPE to apply randomization to duplicate plaintexts. In addition, they introduced a stronger security notion than IND-OCPA, which is known as indistinguishability against frequency-analyzing ordered chosen plaintext attacks (IND-FA-OCPA). In 2017, Maffei et al. [17] has shown that Kerschbaum's security model is imprecise. Therefore, they designed a new construction based on the corrected security model. However, their scheme causes relatively high overhead during encryption due to requiring the order information of all encrypted plaintexts. Moreover, we figure out that the update algorithm used in [16, 17] cannot guarantee to produce a perfectly balanced search tree when duplicate plaintexts are encrypted.

Yang et al. [24] presented a semiorder-preserving encryption (SOPE) although with the sacrifice of the precision of order-preserving. In this scheme, two different plaintexts may be encrypted to the same ciphertext; thus, the ciphertext sequence cannot be mapped to a plaintext. Dyer et al. [25] presented OPE scheme based on the general approximate common divisor problem (GACDP). This approach is the first OPE scheme using a computational hardness, not on a security game. Like Liu and Wang [9], their scheme adds random noise to the initial plaintext so that if there are duplicate plaintexts, the ciphertexts seem like distinct. Kim [26] showed a new OPE scheme based on order-revealing encryption (ORE) and improved the round and client side storage complexities on the exiting ideally secure OPE [16, 20]. Tueno and Kerschbaum [27] introduced an oblivious OPE (OOPE) as an equivalent of a public-key OPE; they also showed a protocol for OOPE that combines existing ideally secure OPE [16, 19] with Paillier's homomorphic encryption and garbled circuits. In [28], Taigel et al. presented a real-life use case that combines OPE and decision tree classification to enable privacy-preserving forecasting of demand for spare parts based on distributed condition data. In [29], Meng and Feigenbaum described an application of OPE that combines OPE, pseudorandom functions (PRFs), and additively homomorphic encryption (AHE) to design a privacy-preserving XGBoost inference algorithm, that is, to create an encrypted regression tree.

1.2. Our Contributions. Table 1 shows the comparison of the existing FH-OPE schemes. As mentioned before, the original definition of IND-FA-OCPA of [16] is imprecise, and their FH-OPE scheme is insecure under the security model that they proposed. In fact, no FH-OPE scheme that can be proven under their security model can exist. The scheme of [17] guarantees the IND-FA-OCPA security that has been revised to be feasible, but the client has to maintain the order information of all the encrypted plaintexts to date; this maintenance causes inefficiency in the client's persistent storage and the encryption performance.

To summarize, our contributions are as follows:

(i) We propose a more practical FH-OPE scheme compared with the previous schemes. Our scheme does not require the order information of all the encrypted plaintexts; thus, the client does not need to maintain them. The security of the proposed scheme can be proven considering the IND-FA-OCPA security model of [17].

(ii) We figure out that the update algorithm in [16, 17] is not suitable for random duplicate distributions. Moreover, it cannot guarantee to produce a perfectly balanced search tree when duplicate plaintexts are encrypted. To overcome this problem, we propose an improved update algorithm. The proposed algorithm always produces a perfectly balanced search tree regardless of the distribution of plaintexts and positively affects the overall performance of FH-OPE.

(iii) We implement the schemes of [16, 17] and the proposed scheme and evaluate the schemes based on different plaintext distributions. Among others, the implementation results show the excellence of our scheme.

1.3. Outline. In Section 2, we recall the formal notion of (stateful) OPE and its security definitions. In Section 3, we analyze the scheme proposed by [17] and present that the scheme still needs to be improved in terms of storage and computational complexity. Section 4 proposes a new practical FH-OPE scheme and an improved update algorithm and shows that the proposed scheme achieves the IND-FA-OCPA security. We present the experimental results in Section 5. Finally, we conclude our work in Section 6.

2. Preliminaries

This section briefly recalls the formal notion of OPE and its security definitions.

2.1. Order-Preserving Encryption. The OPE scheme is defined in two ways: stateless and stateful. A stateless scheme is difficult to achieve the IND-OCPA security. Instead of the stateful OPE being a key-less scheme, a state operates as a secret key.

Table 1: Comparison among the existing FH-OPE schemes.

Open scheme	Encryption cost		Security	Additional persistent storage	Improved update	Note
	Best	Worst				
[16]	$O(\log n)$	$O(\log n)$	IND-OCPA	—	X	Imprecise
[17]	$O(\log n)$	$O(n)$	IND-FA-OCPA	$O(n \log n)$	X	Impractical
Ours	$O(\log n)$	$O(\log n)$	IND-FA-OCPA	—	O	—

* n is the number of plaintexts to be encrypted. Additional persistent storage means the information a client should maintain except for the search tree. Encryption cost denotes the computational complexity of encryption except for the rebalancing of the search tree.

Definition 1 (stateful OPE). A stateful OPE scheme consists of the following three algorithms (Setup, Encryption, and Decryption):

(i) $S \longleftarrow \text{Setup}(1^\lambda)$: the Setup algorithm takes as an input a security parameter λ and outputs a state S.

(ii) $(S', y) \longleftarrow \text{Encryption}(x, S)$: the Encryption algorithm takes as input a plaintext x and a state S. It outputs a ciphertext y and updates the state S to S'.

(iii) $x \longleftarrow \text{Decryption}(y, S)$: the Decryption algorithm takes as input a ciphertext y and a state S. It outputs a plaintext x.

Definition 2 (order-preserving). An OPE scheme is order-preserving if for any two ciphertexts y_1 and y_2 with corresponding messages x_1 and x_2, we have $y_1 \geq y_2 \Rightarrow x_1 \geq x_2$.

2.2. Security Definitions. The standard security notion of OPE is IND-OCPA [2]. It means that an adversary cannot know anything about plaintexts except for their order. Let n be the number of necessarily distinct plaintexts in sequence $X = \{x_1, x_2, \ldots, x_n\}$, where $x_i \in \mathbb{N}$ for all i. The security game $\text{Game}_{\text{IND-OCPA}}(\lambda)$ between the adversary $\mathscr{A}$ and challenger $\mathscr{C}$ for the security parameter λ proceeds as follows:

(1) The adversary $\mathscr{A}$ prepares two sequences X_0 and X_1 of necessarily distinct plaintexts with the same order. He sends them to the challenger $\mathscr{C}$.

(2) The challenger $\mathscr{C}$ randomly chooses $b \leftarrow \{0, 1\}$, executes the Setup (1^λ), and runs $(y_{i,b}, S_i) \leftarrow \text{Encryption}(x_{i,b}, S_{i-1})$. He sends $y_{1 \leq i \leq n,b}$ to the adversary $\mathscr{A}$.

(3) The adversary $\mathscr{A}$ guesses which sequence is encrypted and accordingly outputs guess b'.

We say that the adversary $\mathscr{A}$ wins $\text{Game}_{\text{IND-OCPA}}(\lambda)$ if $b = b'$. Let $Win_{ocpa}^{\mathscr{A},\lambda}$ be the winning probability of $\mathscr{A}$ in $\text{Game}_{\text{IND-OCPA}}(\lambda)$.

Definition 3 (IND-OCPA). A stateful OPE scheme is IND-OCPA secure if for any PPT adversary $\mathscr{A}$, $Win_{ocpa}^{\mathscr{A},\lambda}$ is negligible in the security parameter λ, i.e.,

$$\Pr\left[Win_{ocpa}^{\mathscr{A},\lambda}\right] < \frac{1}{2} + negl(\lambda). \tag{1}$$

Now, we review IND-FA-OCPA, which is originally presented in [16] and modified in [17]. To capture "frequency-hiding" security, it allows duplicate plaintexts, e.g., $X = \{1, 1, 1, 1\}, X' = \{1, 2, 3, 3\}$, and $X'' = \{2, 2, 4, 4\}$. However, two challenge sequences X_0 and X_1 have at least one common randomized order Γ. A randomized order Γ of X means any possible permutation of $\{1, 2, \ldots, |X|\}$ is placed in an order according to X, and the order of duplicate plaintexts is randomly determined. For example, with $X = \{1, 1, 3, 3\}$, the randomized order Γ for X can be any of $\Gamma_1 = \{1, 2, 3, 4\}$, $\Gamma_2 = \{1, 2, 4, 3\}$, $\Gamma_3 = \{2, 1, 3, 4\}$, or $\Gamma_4 = \{2, 1, 4, 3\}$. The randomized order Γ is precisely defined as follows.

Definition 4 (randomized order). Let n be the number of plaintexts in a sequence $X = \{x_1, x_2, \ldots, x_n\}$ that are not necessarily distinct, where $x_i \in \mathbb{N}$ for all i. A randomized order $\Gamma = \{\gamma_1, \gamma_2, \ldots, \gamma_n\}$, where $1 \leq \gamma_i \leq n$ and $i \neq j \Rightarrow \gamma_i \neq \gamma_j$ for all i and j of sequence X, it holds that

$$\forall i, j. \left(x_i > x_j \Longrightarrow \gamma_i > \gamma_j\right) \wedge \left(\gamma_i > \gamma_j \Longrightarrow x_i \geq x_j\right). \tag{2}$$

Two sequences $X_0 = \{1, 3, 3, 3\}$ and $X_1 = \{1, 2, 3, 3\}$ have only two common randomized order: $\Gamma_1 = \{1, 2, 3, 4\}$ and $\Gamma_2 = \{1, 2, 4, 3\}$. $\Gamma_{\downarrow_i}$ denotes the order of the sequences $\{\gamma_1, \ldots \gamma_i\}$. For instance, $\Gamma_{\downarrow_3}$ of $\Gamma = \{3, 5, 1, 2, 4\}$ means $\{2, 3, 1\}$.

The security game $\text{Game}_{\text{IND-FA-OCPA}}(\lambda)$ between an adversary $\mathscr{A}$ and a challenger $\mathscr{C}$ for a security parameter λ proceeds as follows:

(1) The adversary $\mathscr{A}$ prepares two sequences X_0 and X_1 that have at least one common randomized order. He sends them to the challenger $\mathscr{C}$.

(2) The challenger $\mathscr{C}$ randomly chooses $b \longleftarrow \{0, 1\}$ and selects Γ from the common randomized orders of X_0 and X_1. Then, the challenger $\mathscr{C}$ executes the Setup (1^λ) and runs $(y_{i,b}, (y_{i,b}, S_i) \longleftarrow \text{Encryption}(x_{i,b}, S_{i-1})$ based on $\Gamma_{\downarrow_i}$. It means that the relative order of duplicate plaintexts is determined by $\Gamma_{\downarrow_i}$. He sends $y_{1 \leq i \leq n,b}$ to the adversary $\mathscr{A}$.

(3) The adversary $\mathscr{A}$ guesses which sequence is encrypted and accordingly outputs the guess b'.

We say that the adversary $\mathscr{A}$ wins $\text{Game}_{IN\ D\text{–}FA\text{–}OCPA}(\lambda)$ if $b = b'$. Let $Win_{fa-ocpa}^{\mathscr{A},\lambda}$ be the winning probability of $\mathscr{A}$ in $\text{Game}_{IN\ D\text{–}FA\text{–}OCPA}(\lambda)$.

Definition 5 (IND-FA-OCPA). A FH-OPE scheme is IND-FA-OCPA secure if for any PPT adversary $\mathscr{A}$, $Win_{fa-ocpa}^{\mathscr{A},\lambda}$ is negligible in the security parameter λ, i.e.,

$$\Pr\left[Win_{fa-ocpa}^{\mathcal{A},\lambda}\right] < \frac{1}{2} + negl(\lambda). \quad (3)$$

As the randomized order of distinct plaintexts is equal to its order, the IND-FA-OCPA security is stronger than IND-OCPA.

3. Maffei Et Al.'s FH-OPE Scheme

We review the FH-OPE scheme of [17] in detail. The main idea is that the client maintains the randomized order Γ of all encrypted plaintexts and uses it as one of the inputs of the encryption algorithm. It externally determines their relative order for duplicate plaintexts in the encryption algorithm. A search tree T that maps plaintexts to ciphertexts is stored as a state S on the client side and used in the decryption algorithm. For a node of t of T, $t.m$ and $t.c$ represent a plaintext and a corresponding ciphertext. $t.left$ and $t.right$ denote the left and right child of t, respectively. Every node t in T stores its index based on the plaintext sequence. N is the number of distinct plaintexts, and n is the number of plaintexts in the sequence to be encrypted, which also means $|\Gamma|$. k is the number of plaintexts encrypted and stored on the server so far. M denotes the number of distinct ciphertexts, and its bit length is expanded by a factor of λ, i.e., $O(\lambda \log N)$. As described in Algorithm 1, the state S comprises min, max, and T. When the search tree T is empty, the state $(\min, \max)$ is initialized as $(0, M)$. The update (tree rebalancing) algorithm is as described in [16].

To review their scheme, we present a concrete example. Figure 1 shows the encryption results of $\{1, 1, 1, 1\}$, where $M = 128$. Here, we set the possible randomized order Γ of $\{1, 1, 1, 1\}$ to $\{3, 2, 4, 1\}$.

We can check that the algorithm produces ciphertexts $\{64, 32, 96, 16\}$ properly based on Γ. However, Γ should be updated continuously with each encryption. For the mutable OPE schemes whose state is stored on the client side, the computational cost of rebalancing T is similar; thus, it will be excluded from the following efficiency analysis.

Computational Cost. The encryption algorithm [17] has computational complexities $O(\log n)$ and $O(n)$, except the rebalancing T in the best and worst cases, respectively. This is because in the search tree, the cost of finding an empty node and placing a plaintext based on T required for each encryption is $O(\log n)$. In addition, the cost of updating Γ is required. In the case of increasing sequential plaintexts, e.g., $x_1 = 10$, then $\Gamma = \{1\}$; $x_2 = 20$, then $\Gamma = \{1, 2\}$; and $x_3 = 30$, then $\Gamma = \{1, 2, 3\}$; the cost of updating Γ is $O(1)$. On the other hand, for decreasing sequential plaintexts, e.g., $x_1 = 30$, then $\Gamma = \{1\}$; $x_2 = 20$, then $\Gamma = \{2, 1\}$; and $x_3 = 10$, then $\Gamma = \{3, 2, 1\}$; the cost of updating Γ is $O(n)$.

Storage Cost. There are n elements in Γ, and each can be represented by $\log n$ bits. Thus, the client requires $O(n \log n)$ bits for additional persistent storage, except for T.

Rebalancing Tree. In [16, 17], if there is no available ciphertext in T, it has to be rebalanced by calling the update algorithm. However, the algorithm presented in [19] was designed assuming that there were no duplicate plaintexts. Therefore, the algorithm cannot guarantee to produce a perfectly balanced tree when duplicate plaintexts are encrypted. The result quality of the update algorithm significantly impacts the overall performance; thus, a new improved algorithm is needed.

Input: x, S, and $\Gamma = \{\gamma_1, \ldots, \gamma_k\}$
Output: y
State: $S = (t, \min, \max)$
if t is empty **then**
  $t.m = x$
  $t.in\ de\ x = k$
  $t.c = \min + \lceil \max - \min/2 \rceil$
  if $t.c = 0$ **then**
    rebalance the tree
  return $t.c$
$b \longleftarrow -1$
if $x = t.m$ **then**
  $b \longleftarrow \gamma_k > \gamma_{t.in\ de\ x}$
if $b = 1 \vee x > t.m$ **then**
  Encryption $(x, t.right, t.c, \max, \Gamma)$
else
  if $b = 0 \vee x < t.m$ **then**
    Encryption $(x, t.\text{left}, t.c, \max, \Gamma)$

Algorithm 1: Encryption [17].

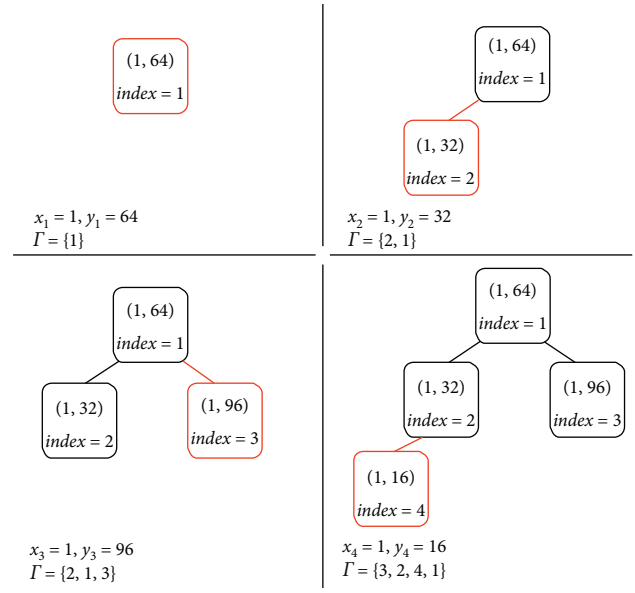

Figure 1: Each resulting search tree of $\{1, 1, 1, 1\}$ in [17].

4. Proposed Scheme

We propose a practical FH-OPE scheme described in Algorithm 2 that achieves the IND-FA-OCPA security with an improved update algorithm. Our search tree T does not need to store the index of the encrypted plaintexts. Let H: $\{0, 1\}^* \longrightarrow \{0, 1\}$ be a hash function with 1-bit output modeled as a random oracle. Our main idea is to replace the

inefficient input Γ with the combination of a single random value $r \longleftarrow \{0,1\}^{ploy(\lambda)}$ and H. In our scheme, the selection of empty nodes for duplicate plaintexts is determined by $H(r\|\text{count})$. It means that the order of duplicate plaintexts is not determined internally but intended externally. The other notations and the initialization T are defined as described in the previous sections.

Figure 2 shows the encryption of duplicate plaintexts {1, 1, 1, 1} based on our scheme. We can check that the algorithm produces distinct ciphertexts {64, 96, 80, 112} based on the chosen random values, e.g., r'' has the same role as $\Gamma = \{1, 3, 2, 4\}$ in [17].

The existing update algorithm for FH-OPE [16, 17, 19] sorts the plaintext sequence $X = \{t.m_1, t.m_2, \ldots, t.m_n\}$ in ascending order and simply re-encrypts the sequence. The algorithms cannot guarantee to produce a perfectly balanced tree because the node positions are randomly selected for the duplicate plaintexts. The idea of our improved update Algorithm 3 is simple. We build a new search tree T^* on $\{1, 2, \ldots, n\}$ where n is the number of nodes in T and replace $t^*.m_i$ with $t.m_i$, where $1 \le i \le n$. We check that the resulting T^* is a perfectly balanced tree because it has been built based on the distinct plaintexts.

In stateful OPE, the decryption algorithm can be omitted by the state that is stored on client side. However, this omission is without loss of correctness of OPE scheme. To decrypt a given ciphertext y, he uses the binary search tree T and finds the node t that includes $t.m$ and $t.c$ where $t.c = y$. Thus, he can simply decrypt the ciphertext y and return a plaintext x by performing the binary search.

Next, we will prove the security of our proposed scheme with regard to the IND-FA-OCPA security model and analyze our construction in terms of efficiency.

Theorem 1. *Let Γ denote any possible randomized order of the plaintext sequence X. Y_Γ denotes the ciphertext sequence when Γ is used as plaintexts. Then, the challenger $\mathscr{C}$ in IND-FA-OCPA can always simulate the ciphertexts of X, which is identical to Y_Γ.*

Proof. In the encryption algorithm of $x_{1\le i\le n}$, let $b_{i,j}$ be the outputs of $H(r_i\|j)$, where $count = j$. Then, we can compute $b_{i,j}$, which is identical to the search tree as if Γ is encrypted. As shown in Figure 2, we can obtain $b_{2,1} = 1$, $b_{3,1} = 1$, $b_{3,2} = 0$, $b_{4,1} = 1$, and $b_{4,2} = 1$ for $\Gamma = \{1, 3, 2, 4\}$. Finally, the challenger $\mathscr{C}$ chooses n random values $\{r_1, r_2, \ldots, r_n\}$ and simulates the random oracle H as $b_{i,j} \longleftarrow H(r_i\|j)$; otherwise, it returns a bit chosen randomly.

Based on Theorem 1, if the challenger $\mathscr{C}$ outputs Y_Γ in step 3 of IND-FA-OCPA, where Γ is the chosen common randomized order in the step 2, there is no advantage for an attacker $\mathscr{A}$ to distinguish X_0 and X_1.

Efficiency. The security in Table 1 demonstrates that the scheme of [16] achieves only the IND-OCPA security and shows that Kerschbaum's model is imprecise, which means no such FH-OPE can exist. Furthermore, both [16, 17] do not provide an improved update algorithm. We can know that the proposed update algorithm positively affects the computational performance through some experiments in Section 5. Compared with [17], our scheme does not require the order information of all the encrypted plaintext while Maffei's scheme requires $O(n \log n)$ bits for additional

Input: x, S, and $(r\|\text{count})$
Output: y
State: $S = (t, \min, \max)$
if *count* = *null* **then**
  count = 1
if max − min = 1 **then**
  Update()
else
  if t is Empty **then**
    $t.m = x$
    $t.c$ = min + ⌈max − min/2⌉
    return $t.c$
  else
    if $x < t.m$ **then**
      Encryption $(x, t.\text{left}, \min, t.c)$ $(r\|\text{count})$
    else if $x = t.x$ **then**
      $b \longleftarrow H(r\|\text{count})$
      count ++
      if $b = 0$ **then**
        Encryption $(x, t.\text{left}, \min, t.c)$, $(r\|\text{count})$
      else
        Encryption $(x, t.\text{left}, \min, t.c)$, $(r\|\text{count})$
    else
      Encryption $(x, t.\text{left}, t.c, \max)$, $(r\|\text{count})$

Algorithm 2: Our encryption.

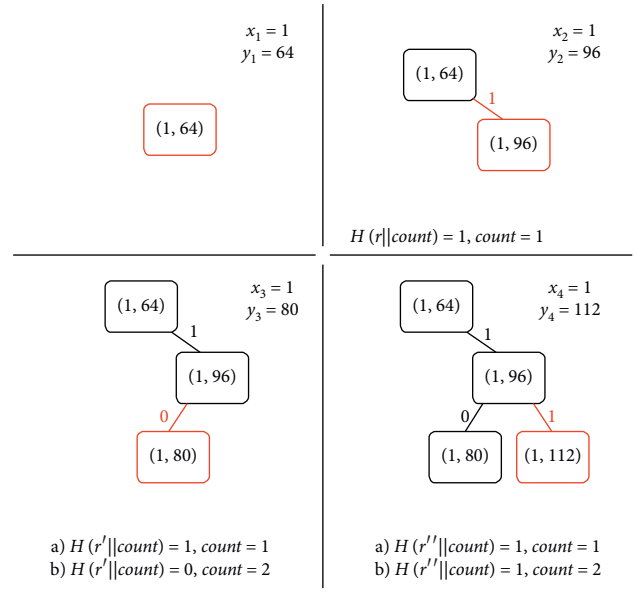

Figure 2: Each resulting search tree of {1, 1, 1, 1} based on our scheme.

Input: a set of $t.m$ in tree T
Output: a balanced search tree T^*
Initialization: a new empty search tree T^*
n = the number of nodes in T
Encryption $(\lceil n/2 \rceil, t^*.\text{root}, \text{null})$
if $n = 2$ **then**
  Encryption $(\lceil n/2 \rceil, t^*.\text{root}, \text{null})$
if $n = 3$ **then**
  Encryption $(\lceil n/2 \rceil, t^*.\text{root}, \text{null})$
  Encryption $(\lceil n/2 \rceil, t^*.\text{root}, \text{null})$
if $n > 3$ **then**
  Update $(\{1, 2, \ldots, \lceil n/2 \rceil - 1\})$
  Update $(\{\lceil n/2 \rceil + 1, \ldots, n-1, n\})$
if End recursively iterate **then**
  Call $X = \{t.m_1, \ldots, t.m_n\}$ in ascending order
  Call $X = \{t^*.m_1, \ldots, t^*.m_n\}$ in ascending order
  for $j \longleftarrow 1$ *to* n **do**
    $t^*.m_j \longleftarrow t.m_j$
  return tree T^*

Algorithm 3: Our update.

persistent storage, except for a state. Moreover, a series of sorting every element in the randomized order Γ causes very low computational performance of [17] because these operations occur whenever a plaintext is inserted into the encryption algorithm of [17].

5. Experiments

We analyze the performance of [16, 17] and the proposed scheme using a system that includes an AMD Ryzen 5 3600 6-core processor 3.59 GHz, 16 GB RAM in Python 3.9.5. We use different plaintext sizes N and the number of plaintexts to be encrypted n, but the ciphertext size M is fixed at 2048.

5.1. Random Duplicate Plaintexts. Figure 3 shows the comparison of the encryption of n plaintexts that are randomly selected in $\{1, 2, \ldots, N\}$, allowing duplicates, where $n \in \{128, 256, 384, 512, 640, 768, 896, 1024\}$ and the corresponding $N \in \{64, 128, 192, 256, 320, 384, 448, 512\}$. As the main operation of [17] is to maintain the order of all encrypted plaintexts, their scheme requires additional Γ updates except for the ciphertext updates. Figure 3 shows that scheme of [17] exhibits lower performance than the other.

5.2. Random Distinct Plaintexts. Here, we encrypt n plaintexts that are selected randomly in $\{1, 2, \ldots, N\}$, not allowing duplicates where $n = N$. Figure 4 shows that the overall speed improved in all the schemes owing to the blockage of the duplicate plaintexts, but the encryption time of [17] is more than the other.

5.3. Sequential Plaintexts. We encrypt plaintexts $\{N, N-1, \ldots, 1\}$, where $N \in \{128, 256, 384, 512, 640, 768, 896, 1024\}$. As these plaintexts cause the rebalancing tree most frequently, Figure 5 shows that all the schemes take more time to encrypt data owing to the frequent updates of search tree. However, as the number of plaintexts to be encrypted increases, the encryption time of the scheme [17] sharply increases.

5.4. Ciphertext Update Cost. In this section, we prove that our update algorithm is better than other algorithms. We encrypt n plaintexts that are selected randomly in

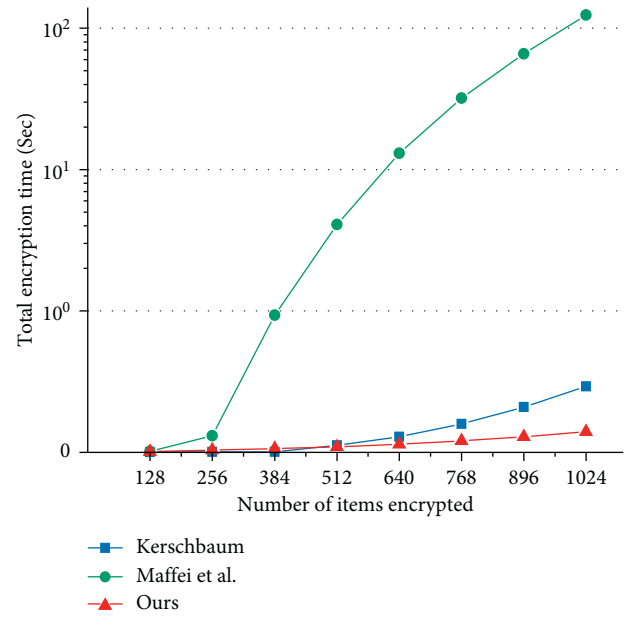

Figure 3: Encryption time of [16, 17] and ours based on random duplicate plaintexts.

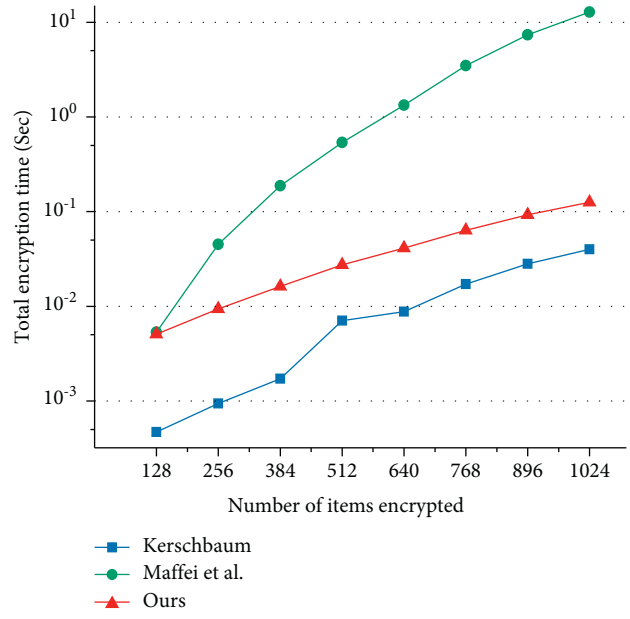

Figure 4: Encryption time of [16, 17] and ours based on random distinct plaintexts.

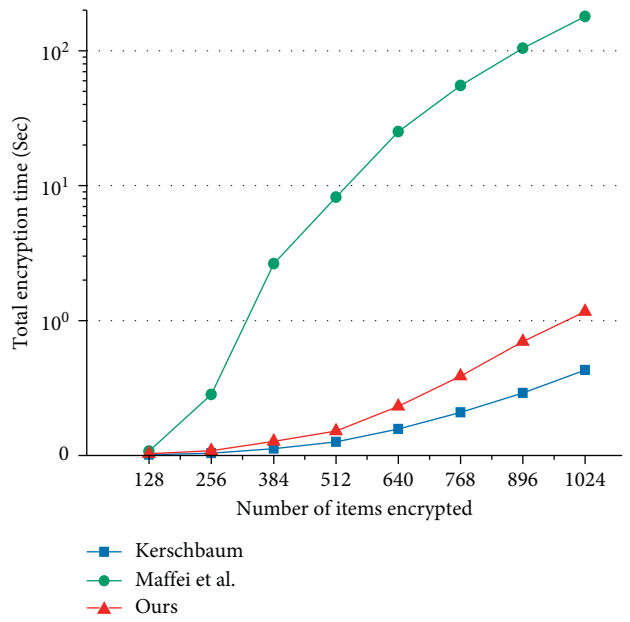

Figure 5: Encryption time of [16, 17] and ours based on sequential plaintexts.

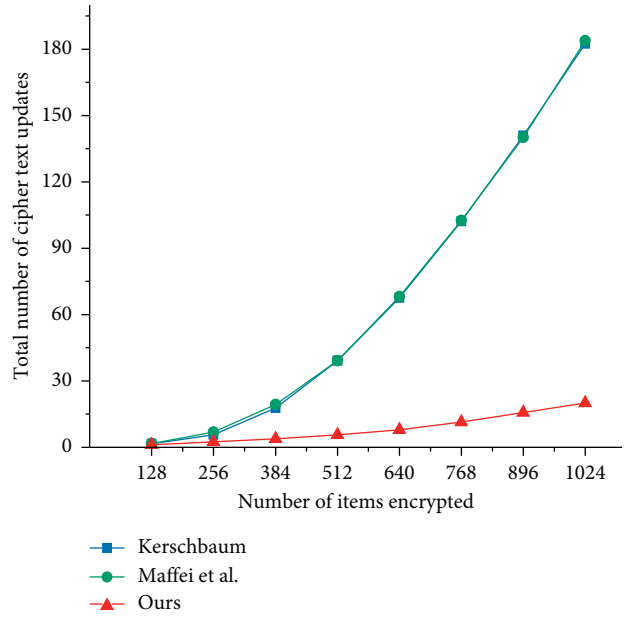

Figure 6: The number of updates of [16, 17] and ours based on random duplicate plaintexts.

$\{1, 2, \ldots, N\}$, allowing duplicates where $n \in \{128, 256, 384, 512, 640, 768, 896, 1024\}$ and the corresponding $N \in \{64, 128, 192, 256, 320, 384, 448, 512\}$. In our update algorithm, there is no case of producing worst case that the update algorithm outputs a skewed search tree. The update algorithm in [16, 17] may produce the unbalanced search tree after executing the ciphertext updates. Therefore, Figure 6 shows that the number of updates in our case is relatively small.

6. Conclusion

We review the construction presented by Maffei et al. and conclude that the scheme still needs to be improved in terms of storage and computational complexity. Then, we propose a more practical FH-OPE scheme with the formal IND-FA-OCPA security proof. Moreover, we figure out that the previous update algorithms are not suitable for the duplicate plaintexts and propose an improved update algorithm that helps produce a perfectly balanced search tree regardless of the distribution of the plaintexts. Finally, we present some experimental results to demonstrate the excellence of the proposed scheme.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. NRF-2019R1G1A1097540).

References

[1] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, "Order preserving encryption for numeric data," in *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, pp. 563–574, ACM Digital Library, New York, June 2004.

[2] A. Boldyreva, N. Chenette, Y. Lee, and A. O'neill, "Order-preserving symmetric encryption," in *Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 224–241, Springer, Berlin, Heidelberg, October 2009.

[3] A. Boldyreva, N. Chenette, and A. O'Neill, "Order-preserving encryption revisited: improved security analysis and alternative solutions," in *Proceedings of the Annual Cryptology Conference*, pp. 578–595, Springer, Berlin, Heidelberg, April 2001.

[4] L. Xiao and I.-L. Yen, "Security analysis for order preserving encryption schemes," in *Proceedings of the 2012 46th annual conference on information sciences and systems (CISS)*, pp. 1–6, IEEE, Princeton, NJ, USA, March 2012.

[5] D. H. Yum, D. S. Kim, J. S. Kim, P. J. Lee, and S. J. Hong, "Order-preserving encryption for non-uniformly distributed plaintexts," in *Proceedings of the International Workshop on Information Security Applications*, pp. 84–97, Springer, Berlin, Heidelberg, November 2011.

[6] S. Lee, T.-J. Park, D. Lee, T. Nam, and S. Kim, "Chaotic order preserving encryption for efficient and secure queries on databases," *IEICE - Transactions on Info and Systems*, vol. E92-D, no. 11, pp. 2207–2217, 2009.

[7] 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 on Info and Systems, vol. E93-D, no. 9, pp. 2520–2533, 2010.

[8] S. Hildenbrand, D. Kossmann, T. Sanamrad, C. Binnig, F. Faerber, and J. Woehler, "Query processing on encrypted data in the cloud by," *Technical report*, vol. 735, 2011.

[9] D. Liu and S. Wang, "Programmable order-preserving secure index for encrypted database query," in *Proceedings of the 2012 IEEE Fifth International Conference on Cloud Computing*, pp. 502–509, IEEE, Honolulu, HI, USA, June 2012.

[10] L. Xiao, I.-L. Yen, and D. T. Huynh, "Extending order preserving encryption for multi-user systems," *IACR Cryptol. ePrint Arch*, vol. 2012, p. 192, 2012.

[11] D. Liu and S. Wang, "Nonlinear order preserving index for encrypted database query in service cloud environments," *Concurrency and Computation: Practice and Experience*, vol. 25, no. 13, pp. 1967–1984, 2013.

[12] T. Boelter, R. Poddar, and R. A. Popa, "A secure one-roundtrip index for range queries," *IACR Cryptol. ePrint Arch.*vol. 2016, p. 568, 2016.

[13] H. Kadhem, T. Amagasa, and H. Kitagawa, "A secure and efficient order preserving encryption scheme for relational databases," *KMIS*, pp. 25–35, 2010.

[14] L. Xiao, I.-L. Yen, and D. Huynh, "A note for the ideal order-preserving encryption object and generalized order-preserving encryption," *IACR Cryptol. ePrint Arch.*vol. 2012, p. 350, 2012.

[15] G. W. Ang, J. H. Woelfel, and T. P. Woloszyn, "System and method of sort-order preserving tokenization," *uS Patent*, vol. 8, no. 739, p. 265, 2014.

[16] F. Kerschbaum, "Frequency-hiding order-preserving encryption," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pp. 656–667, ACM Digital Library, New York, October 2015.

[17] M. Maffei, M. Reinert, and D. Schröder, "On the security of frequency-hiding order-preserving encryption," in *Proceedings of the International Conference on Cryptology and Network Security*, pp. 51–70, Springer, Switzerland AG, November 2018.

[18] R. A. Popa, F. H. Li, and N. Zeldovich, "An ideal-security protocol for order-preserving encoding," in *Proceedings of the 2013 IEEE Symposium on Security and Privacy*, pp. 463–477, IEEE, Berkeley, CA, USA, May 2013.

[19] F. Kerschbaum and A. Schröpfer, "Optimal average-complexity ideal-security order-preserving encryption," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pp. 275–286, ACM Digital Library, New York, November 2014.

[20] D. S. Roche, D. Apon, S. G. Choi, and A. Yerukhimovich, "Pope: partial order preserving encoding," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1131–1142, ACM Digital Library, New York, October 2016.

[21] I. Teranishi, M. Yung, and T. Malkin, "Order-preserving encryption secure beyond one-wayness," in *Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security*, pp. 42–61, Springer, Berlin, Heidelberg, August 2014.

[22] S. F. Krendelev, M. Yakovlev, and M. Usoltseva, "Order-preserving encryption schemes based on arithmetic coding and matrices," *Proceedings of the 2014 Federated Conference on Computer Science and Information Systems*, vol. 2, pp. 891–899, 2014.

[23] M. Naveed, S. Kamara, and C. V. Wright, "Inference attacks on property-preserving encrypted databases," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pp. 644–655, ACM Digital Library, New York, October 2015.

[24] C. Yang, W. Zhang, and N. Yu, "Semi-order preserving encryption," *Information Sciences*, vol. 387, pp. 266–279, 2017.

[25] J. Dyer, M. Dyer, and K. Djemame, "Order-preserving encryption using approximate common divisors," *Journal of Information Security and Applications*, vol. 49, p. 102391, 2019.

[26] K. S. Kim, "New construction of order-preserving encryption based on order-revealing encryption," *Journal of Information Processing Systems*, vol. 15, no. 5, pp. 1211–1217, 2019.

[27] A. Tueno and F. Kerschbaum, "Efficient secure computation of order-preserving encryption," in *Proceedings of the 15th ACM Asia Conference on Computer and Communications Security, ACM Digital Library*, pp. 193–207, New York, October 2020.

[28] F. Taigel, A. K. Tueno, and R. Pibernik, "Privacy-preserving condition-based forecasting using machine learning," *Journal of Business Economics*, vol. 88, no. 5, pp. 563–592, 2018.

[29] X. Meng and J. Feigenbaum, "Privacy-preserving xgboost inference," *Cryptography and Security*, vol. 1, 2020.