On the Security of Frequency-Hiding Order-Preserving Encryption

Matteo Maffei[1], Manuel Reinert[2](✉), and Dominique Schröder[3]

[1] TU Wien, Wien, Austria
matteo.maffei@tuwien.ac.at
[2] CISPA, Saarland University, Saarbrücken, Germany
reinert@cs.uni-saarland.de
[3] Friedrich-Alexander Universität Erlangen-Nürnberg, Nürnberg, Germany
dominique.schroeder@fau.de

Abstract. Order-preserving encryption (OPE) is an encryption scheme with the property that the ordering of the plaintexts carry over to the ciphertexts. This primitive is particularly useful in the setting of encrypted databases because it enables efficient range queries over encrypted data. Given its practicality and usefulness in the design of databases on encrypted data, OPE's popularity is growing. Unfortunately, nearly all computationally efficient OPE constructions are vulnerable against ciphertext frequency-leakage, which allows for inferring the underlying plaintext frequency. To overcome this weakness, Kerschbaum recently proposed a security model, designed a frequency-hiding OPE scheme, and analyzed its security in the programmable random oracle model (CCS 2015).
In this work, we demonstrate that Kerschbaum's definition is imprecise and using its natural interpretation, we describe an attack against his scheme. We generalize our attack and show that his definition is, in fact, not satisfiable. The basic idea of our impossibility result is to show that any scheme satisfying his security notion is also IND-CPA-secure, which contradicts the very nature of OPE. As a consequence, no such scheme can exist. To complete the picture, we rule out the imprecision in the security definition and show that a slight adaption of Kerschbaum's tree-based scheme fulfills it.

1 Introduction

Outsourcing databases is common practice in today's businesses. The reasons for that are manifold, varying from the sharing of data among different offices of the same company to saving on know-how and costs that would be necessary to maintain such systems locally. Outsourcing information, however, raises privacy concerns with respect to the service provider hosting the data. A first step towards a privacy-preserving solution is to outsource encrypted data and to let the database application operate on ciphertexts. However, simply encrypting all entries does in general not work because several standard queries on the database do no longer work. To maintain as much functionality of the database as possible while adding confidentiality properties, researchers weakened the security properties of encryption schemes to find a useful middle ground. Examples include encryption schemes that support plaintext equality checks, or order-preserving encryption. In this work, we re-visit the recent work on frequency-hiding order preserving encryption by Kerschbaum [11] (CCS 2015).

Background and Related Work. Order-preserving encryption (OPE) [3, 20] is arguably the most popular building block for databases on encrypted data, since it allows for inferring the order of plaintexts by just looking at the respective ciphertexts. More precisely, for any two plaintexts p_1 and p_2, whenever $p_1 < p_2$, we have that $\mathsf{E}(p_1) < \mathsf{E}(p_2)$. Hence, OPE allows for efficient range queries and keyword search on the encrypted data. The popularity of this scheme is vouched for by plenty of industrial products (e.g., Ciphercloud[4], Perspecsys[5], and Skyhigh Networks[6]) and research that investigates OPE usage in different scenarios [1, 2, 10, 13, 17, 18]. Despite the growing popularity and usage in practice, OPE security is debatable. The ideal security notion for OPE is called *indistinguishability against ordered chosen plaintext attacks* (IND-OCPA), which intuitively says that two equally ordered plaintext sequences should be indistinguishable under encryption. Boldyreva *et al.* [3] show that stateless OPE cannot achieve IND-OCPA, unless the ciphertext size is exponential in the plaintext size. Consequently, either one has to relax the security notion or to keep a state.

The former approach has been explored in the context of classical OPE [3, 4, 21] as well as a slightly different notion called *order-revealing encryption* (ORE) [5, 6, 14, 19]. ORE is more general than OPE in the sense that comparison on the ciphertexts can happen by computing a comparison function different from "<". Either way, those schemes do not achieve IND-OCPA but target different, weaker security notions, which allow them to quantify the leakage incurred by a scheme or to restrict the attacker's capabilities. For instance, the scheme by Boldyreva *et al.* [3] is known to leak about the first half of the plaintexts and the scheme by Chenette *et al.* [6] leaks the first bit where two encrypted target plaintexts differ. To date, there exist several works that exploit this extra leakage in order to break OPE applied to different data sets such as medical data and census data [7–9, 15]. For instance, using a technique based on bipartite graphs, Grubbs *et al.* [9] have recently shown how to break the schemes of Boldyreva *et al.* [3, 4], thereby achieving recovery rates of up to 98%. As opposed to earlier work, this technique works even for large plaintext domains such as first names, last names, and even zip codes.

With regards to the latter approach based on stateful OPE schemes, Popa *et al.* [16] introduced a client-server architecture, where the client encrypts plaintexts using a deterministic encryption scheme and maintains a search tree on the server into which it inserts the ciphertexts. The server exploits the search tree when computing queries on encrypted data. This approach requires a significant amount of communication between the client and the server both for encryp-

[4] http://www.ciphercloud.com/
[5] http://perspecsys.com/
[6] https://www.skyhighnetworks.com/

tion and queries. Similarly, but rather reversed, Kerschbaum and Schroepfer [12] present an OPE scheme where the client stores a search tree that maps plaintexts to ciphertexts. The ciphertexts are chosen such that ordering is preserved and then inserted along with the plaintexts in the search tree. The server only learns the ciphertexts. This approach has less communication between client and server but requires the client to keep a state that is linear in the number of encrypted plaintexts. Both of these schemes are provably IND-OCPA-secure.

Even though these schemes achieve the ideal IND-OCPA security notion, Kerschbaum [11] raises general doubts about the security definition of OPE. Aside the leakage that is introduced by many schemes on top of the order information (e.g., leaking half of the plaintext [3] or the first bit where two plaintexts differ [6]), one central problem of OPE is the leakage of the plaintext frequency. It is easy to distinguish the encryption of data collections in which elements occur with different frequencies. For instance, the encryption of the sequences $1, 2, 3, 4$ and $1, 1, 2, 2$ are not necessarily indistinguishable according to the IND-OCPA security definition.

In order to solve the frequency-leakage problem, Kerschbaum has recently strengthened the IND-OCPA definition of OPE so as to further hide the frequency of plaintexts under the encryption, thus making the encryptions of the above two sequences indistinguishable [11] (CCS 2015). To this end, Kerschbaum introduces the notion of *randomized order*, which is a permutation of the sequence $1, \ldots, n$ where n is the length of the challenge plaintext sequence. Such a permutation is called randomized order if, when applied to a plaintext sequence, the resulting plaintext sequence is ordered with respect to "$\leq$". The original IND-OCPA security definition requires that the two challenge plaintext sequences agree on all such common randomized orders, which implies that every pair of corresponding plaintexts in the two sequences occurs with the same frequency. For instance, this does not hold for the above two sequences $1, 2, 3, 4$ and $1, 1, 2, 2$, since the former can only be ordered using the permutation $(1, 2, 3, 4)$ while the latter can be ordered by any of $(1, 2, 3, 4)$, $(1, 2, 4, 3)$, $(2, 1, 3, 4)$, or $(2, 1, 4, 3)$. Kerschbaum's insight to make the definition frequency-hiding is that the existence of one common randomized order should be sufficient in order not to be able to distinguish them. For instance, the above sequences both share the randomized order $(1, 2, 3, 4)$ and should thus be indistinguishable when encrypted. This intuition is captured by the security notion of *indistinguishability against frequency-analyzing ordered chosen plaintext attacks* (IND-FA-OCPA). Besides devising a novel definition, Kerschbaum also presents a cryptographic instantiation of an OPE scheme and analyzes its security with respect to the new definition in the programmable random oracle model.

Despite the seeming improvement added by Kerschbaum's scheme, Grubbs *et al.* [9] show that using auxiliary information, such as the plaintext distribution that is likely to underlie a certain ciphertext collection, this scheme can be broken with significant recovery rates. In contrast to the practical attacks in [9], our work targets the purely theoretic side of frequency-hiding OPE and we do not consider having auxiliary information at disposal.

Our Contributions. In this work, we present both negative and positive results for frequency-hiding order-preserving encryption. On the negative side, we observe that the original definition of IND-FA-OCPA is imprecise [11], which leaves room for interpretation. In particular, the security proof for the scheme presented in [11] seems to suggest that the game challenger chooses a randomized order according to which one of the challenge sequences is encrypted. This fact, however, is not reflected in the definition. Hence, according to a natural interpretation of the definition, we show that it is, in fact, not achievable. We develop this impossibility result for the natural interpretation of IND-FA-OCPA step by step. Investigating on Kerschbaum's frequency-hiding scheme [11], we show that it can actually be attacked–without using auxiliary information as done by Grubbs *et al.* [9]–allowing an adversary to win the IND-FA-OCPA game with very high probability. We further observe that this concrete attack can be generalized into a result that allows us to precisely quantify an attacker's advantage in winning the security game for two arbitrary plaintext sequences that adhere to the security game restrictions. Since Kerschbaum provides formal security claims for his construction [11], we identify where the security proof is incorrect. All these considerations on the concrete scheme finally lead to our main negative result: IND-FA-OCPA security is impossible to achieve or, more precisely, any IND-FA-OCPA secure OPE scheme is also secure with respect to IND-CPA, which clearly contradicts the very functionality of OPE. Hence, such an OPE scheme cannot exist.

As mentioned above, the impossibility of IND-FA-OCPA is bound to an imprecision in the definition in [11], which is only presented informally and lacks necessary information to make it achievable. We hence clarify those imprecisions. The underlying problem of the original definition lies in the capability of the game challenger, which, when reading the definition naturally, is very restricted. The challenger has, for instance, no means to ensure that the encryption algorithm chooses a common randomized order of the two challenge plaintext sequences. To remedy those shortcomings, we devise a more formal definition that removes the consisting imprecisions and makes it possible to devise a frequency-hiding OPE scheme. In particular, we first augment the OPE model, allowing for specifying a concrete ordering when encrypting plaintexts, e.g., to concretely say that the sequence $1, 1, 2, 2$ should be encrypted sticking to the randomized order $(1, 2, 4, 3)$. Secondly, we show that an extension of Kerschbaum's scheme [11], adapted to the new model, is provably secure with respect to the correct definition.

To summarize, our contributions are as follows.

– We show that the original definition of IND-FA-OCPA is imprecise. We then demonstrate that the frequency-hiding OPE scheme of [11] is insecure under a natural interpretation of IND-FA-OCPA. We further generalize the attack, which allows us to rigorously quantify the success probability of an attacker for two arbitrary plaintext sequences. To conclude on the concrete scheme, we identify and explain the problem in the security proof.

– Going one step beyond the concrete scheme, we prove a general impossibility result showing that IND-FA-OCPA cannot be achieved by any OPE scheme.
– We clarify the imprecise points in the original security definition and provide a corrected version called IND-FA-OCPA*.
– To define IND-FA-OCPA* in the first place, we have to augment the OPE model, adding a concrete random order as input to the encryption function.
– Finally, we prove that an extension of [11] fulfills our new definition.

Overall, we believe this work yields a solid foundation for order-preserving encryption, showing that a state-of-the-art security definition is impossible to realize along with an attack on a previously published scheme, and presenting an achievable definition and a concrete realization.

Outline. The rest of the paper is structured as follows. We recall the OPE model and its security definitions in Section 2. In Section 3 we describe the relevant parts of Kerschbaum's scheme [11]. We present our attack, its generalization, and the problem in the security proof in Section 4. Section 5 proves the impossibility result. In Section 6 we present the augmented OPE model and the definition of IND-FA-OCPA*. We show that an adaption of [11] to the new model achieves IND-FA-OCPA* in Section 7. Finally, we conclude this work in Section 8.

2 Order-Preserving Encryption

In this section, we briefly review the formal definitions of order-preserving encryption, originally proposed in [20], following the definition adopted in [11].

Definition 1 ((Order-Preserving) Encryption). *An* (order-preserving) encryption scheme $\mathcal{OPE} = (\mathsf{K}, \mathsf{E}, \mathsf{D})$ *is a tuple of* PPT *algorithms where* $S \leftarrow \mathsf{K}(\kappa)$ *.*

The key generation *algorithm takes as input a security parameter* κ *and outputs a secret key (or state)* S*.*

$(S', y) \leftarrow \mathsf{E}(S, x)$*. The* encryption *algorithm takes as input a secret key* S *and a message* x*. It outputs a new key* S' *and a ciphertext* y*;*

$x \leftarrow \mathsf{D}(S, y)$*. The* decryption *algorithm takes as input a secret key* S *and a ciphertext* y *and outputs a message* x*.*

An OPE scheme is *complete* if for all S, S', x, and y we have that if $(S', y) \leftarrow \mathsf{E}(S, x)$, then $x \leftarrow \mathsf{D}(S', y)$.

The next definition formalizes the property of order preservation for an encryption scheme. Roughly speaking, this property says that the ordering on the plaintext space carries over to the ciphertext space.

Definition 2 (Order-Preserving). *An encryption scheme* $\mathcal{OPE} = (\mathsf{K}, \mathsf{E}, \mathsf{D})$ *is* order-preserving *if for any two ciphertexts* y_1 *and* y_2 *with corresponding messages* x_1 *and* x_2 *we have that whenever* $y_1 < y_2$ *then also* $x_1 < x_2$*.*

This general definition allows for modeling both stateful as well as stateless versions of OPE. We focus on the stateful variant in this paper, hence, the *key* S defined above is actually the state. The definition, moreover, does not specify where the state has to reside, allowing us to model client-server architectures.

2.1 Security Definitions

Indistinguishability Against Ordered Chosen Plaintext Attacks. The standard security definition for order-preserving encryption is indistinguishability against ordered chosen plaintext attacks (IND-OCPA) [3]. Intuitively, an OPE scheme is secure with respect to this definition if for any two equally ordered plaintext sequences, no adversary can tell apart their corresponding ciphertext sequences. IND-OCPA is fulfilled by several schemes (e.g., [12, 16]). We recall the selective version of the definition in the following.

Definition 3 (IND-OCPA). *An order-preserving encryption scheme* $\mathcal{OPE} = (\mathsf{K}, \mathsf{E}, \mathsf{D})$ *has* indistinguishable ciphertexts under ordered chosen plaintext attacks *(IND-OCPA) if for any* PPT *adversary* $\mathcal{A}$*, the following probability is negligible in the security parameter* κ*:*

$$\left|\mathsf{Pr}[\mathsf{Exp}^{\mathcal{A}}_{\mathsf{OCPA}}(\kappa, 1) = 1] - \mathsf{Pr}[\mathsf{Exp}^{\mathcal{A}}_{\mathsf{OCPA}}(\kappa, 0) = 1]\right|$$

where $\mathsf{Exp}^{\mathcal{A}}_{\mathsf{OCPA}}(\kappa, b)$ *is the following experiment:*

```
Experiment $\mathsf{Exp}^{\mathcal{A}}_{\mathsf{OCPA}}(\kappa, b)$
    $(X_0, X_1) \leftarrow \mathcal{A}$ where $|X_0| = |X_1| = n$ and
        $\forall 1 \leq i, j \leq n.\ x_{0,i} < x_{0,j} \iff x_{1,i} < x_{1,j}$
    $S_0 \leftarrow \mathsf{K}(\kappa)$
    For all $1 \leq i \leq n$ run $(S_i, y_{b,i}) \leftarrow \mathsf{E}(S_{i-1}, x_{b,i})$
    $b' \leftarrow \mathcal{A}(y_{b,1}, \ldots, y_{b,n})$
    Output 1 if and only if $b = b'$.
```

Definition 3 requires that the challenge plaintext sequences are ordered exactly the same, which in particular implies that the plaintext frequency must be the same.

Indistinguishability Under Frequency-Analyzing Ordered Chosen Plaintext Attacks. A drawback of the previous definition is that it can be achieved by schemes that leak the plaintext frequency, although any two sequences in which plaintexts occur with different frequencies, e.g., $1, 2, 3, 4$ and $1, 1, 1, 1$, are trivially distinguishable by the attacker. In order to target even such sequences, Kerschbaum [11] proposes a different security definition: instead of requiring the sequences to have exactly the same order, it is sufficient for them to have a common *randomized* order. For a plaintext list X of length n, a randomized order is a permutation of the plaintext indices $1, \ldots, n$ which are ordered according to a sorted version of X. This is best explained by an example:

consider the plaintext sequence $X = 1, 5, 3, 8, 3, 8$. A randomized order thereof can be any of $\mathit{\Gamma}_1 = (1, 4, 2, 5, 3, 6)$, $\mathit{\Gamma}_2 = (1, 4, 3, 5, 2, 6)$, $\mathit{\Gamma}_3 = (1, 4, 2, 6, 3, 5)$, or $\mathit{\Gamma}_4 = (1, 4, 3, 6, 2, 5)$, because the order of 3 and 3 as well as the order of 8 and 8 does not matter in a sorted version of X. Formally, a randomized order is defined as follows.

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

$$\forall i, j.\ (x_i > x_j \implies \gamma_i > \gamma_j) \ \wedge\ (\gamma_i > \gamma_j \implies x_i \geq x_j)$$

Using this definition, Kerschbaum [11] defines security of OPE against *frequency-analyzing ordered chosen plaintext attacks*. Since the definition is informal in [11], we report the natural way to read the definition.

Definition 5 (IND-FA-OCPA). *An order-preserving encryption scheme $\mathcal{OPE} = (\mathsf{K}, \mathsf{E}, \mathsf{D})$ has* indistinguishable ciphertexts under frequency-analyzing ordered chosen plaintext attacks *(IND-FA-OCPA) if for any* PPT *adversary $\mathcal{A}$, the following probability is negligible in the security parameter κ:*

$$\left| \Pr[\mathsf{Exp}^{\mathcal{A}}_{\mathsf{FA-OCPA}}(\kappa, 1) = 1] - \Pr[\mathsf{Exp}^{\mathcal{A}}_{\mathsf{FA-OCPA}}(\kappa, 0) = 1] \right|$$

where $\mathsf{Exp}^{\mathcal{A}}_{\mathsf{FA-OCPA}}(\kappa, b)$ is the following experiment:

```
Experiment $\mathsf{Exp}^{\mathcal{A}}_{\mathsf{FA-OCPA}}(\kappa, b)$
    $(X_0, X_1) \leftarrow \mathcal{A}$ where $|X_0| = |X_1| = n$ and $X_0$ and $X_1$
        have at least one common randomized order $\mathit{\Gamma}$
    $S_0 \leftarrow \mathsf{K}(\kappa)$
    For all $1 \leq i \leq n$ run $(S_i, y_{b,i}) \leftarrow \mathsf{E}(S_{i-1}, x_{b,i})$
    $b' \leftarrow \mathcal{A}(y_{b,1}, \ldots, y_{b,n})$
    Output 1 if and only if $b = b'$
```

It is clear that while the standard IND-OCPA definition could be achieved, in principle, by a deterministic encryption scheme, the frequency-hiding variant can only be achieved by using randomized ciphertexts since otherwise frequencies are trivially leaked.

Discussion. Comparing the two definitions, we observe that IND-FA-OCPA is a generalization of IND-OCPA since the constraint on the sequences X_0 and X_1 allows for a greater class of instances. In order to see that, we have to consider the constraint, which is

$$\forall 1 \leq i, j \leq n.\ x_{0,i} < x_{0,j} \iff x_{1,i} < x_{1,j}.$$

This constraint is an alternative way of saying that X_0 and X_1 should agree on all randomized orders. Hence, duplicate plaintexts may occur in any of the sequences, but they should occur symmetrically in the other sequence as well.

3 Kerschbaum's Construction

We review the OPE scheme of [11]. At a high level, encryption works by inserting plaintexts into a binary search tree that stores duplicates as often as they occur. When an element arrives at its designated node, a ciphertext is selected according to this position.

More formally, let T be a binary tree. We denote by ρ the root of T. For a node $t \in T$ we write $t.m$ to denote the message stored at t and $t.c$ to denote the respective ciphertext. We further use $t.\mathit{left}$ and $t.\mathit{right}$ to denote the left and right child of t, respectively. There are several other parameters: N is the number of distinct plaintexts, n is the number of plaintexts in the sequence that is to be encrypted, $k = \log(N)$ is the required number of bits necessary to represent a plaintext in a node, $\ell = k\kappa$ is this number expanded by a factor of κ and refers to the size of the ciphertexts. Finally, the construction requires a source of randomness, which is called in terms of a function $\mathsf{RandomCoin}()$ (hereafter called $\mathsf{RC}()$ for brevity). According to Kerschbaum, this function can be implemented as a PRF that samples uniformly random bits.

We refer in the following to the client as the one storing the binary tree. This is well motivated in the cloud setting where the client outsources encrypted data to the cloud server who may not have access to the actual message-ciphertext mapping. One may wonder why a client that anyway has to store a mapping of plaintexts to ciphertexts cannot simply store the data itself: Kerschbaum also presents an efficient compression technique for the tree which in some cases can lead to compression ratios of 15.

Implementation of $S \leftarrow \mathsf{K}(\kappa)$. The client sets up an empty tree T. The state S consists of the tree T as well as all parameters k, ℓ, n, and N. Furthermore, S contains the minimum ciphertext $\mathit{min} = -1$ and the maximum ciphertext $\mathit{max} = 2^{\kappa \log(n)}$. These minimum and maximum numbers are only necessary to write the encryption procedure in a recursive way. Usually, n is not known upfront, so it has to be estimated. If the estimation is too far from reality, the tree has to be freshly setup with new parameters.

Algorithm 1 $\mathsf{E}(S, x)$ where $S = t, \mathit{min}, \mathit{max}$

```
1: if $t = \perp$ then
2:     $t.m = x$
3:     $t.c = \mathit{min} + \lfloor \frac{\mathit{max}-\mathit{min}}{2} \rfloor$
4:     if $t.c = 0$ then
5:         rebalance the tree
6:     end if
7:     return $t.c$
8: end if
9: $b \leftarrow -1$
10: if $x = t.m$ then
11:     $b \leftarrow \mathsf{RC}()$
12: end if
13: if $b = 1 \vee x > t.m$ then
14:     $\mathsf{E}(t.\mathit{right}, t.c + 1, \mathit{max}, x)$
15: else
16:     if $b = 0 \vee x < t.m$ then
17:         $\mathsf{E}(t.\mathit{left}, \mathit{min}, t.c - 1, x)$
18:     end if
19: end if
```

Implementation of $(S', y) \leftarrow \mathsf{E}(S, x)$. To encrypt a plaintext x, the client proceeds as follows. Whenever the current position in the tree is empty (especially in the beginning when the tree is empty), the client creates a new tree node and inserts x as the plaintext (lines 1.1–1.8). The ciphertext is computed as the mean value of the interval from *min* to *max* (line 1.3). In particular, the first ciphertext will be $2^{\kappa \log(n)-1}$. Whenever there is no ciphertext available (line 1.4), the estimation of n has shown to be wrong and the tree has to be rebalanced. We do not detail this step here since it is not important for our attack; instead we refer the interested reader to [11] for a detailed explanation. If instead, the current position in the tree is already occupied and the message is different from x, then we either recurse left (line 1.17) or right (line 1.14) depending on the relation between the occupying plaintext and the one to be inserted. The same happens in case x is equal to the stored message, but then we use the RC procedure to decide where to recurse (lines 1.9–1.12).

Implementation of $x \leftarrow \mathsf{D}(S, y)$. To decrypt a given ciphertext y, we treat the tree as a binary search tree where the key is $t.c$ and search for y. We return $t.m$ as soon as we reach a node t where $t.c = y$.

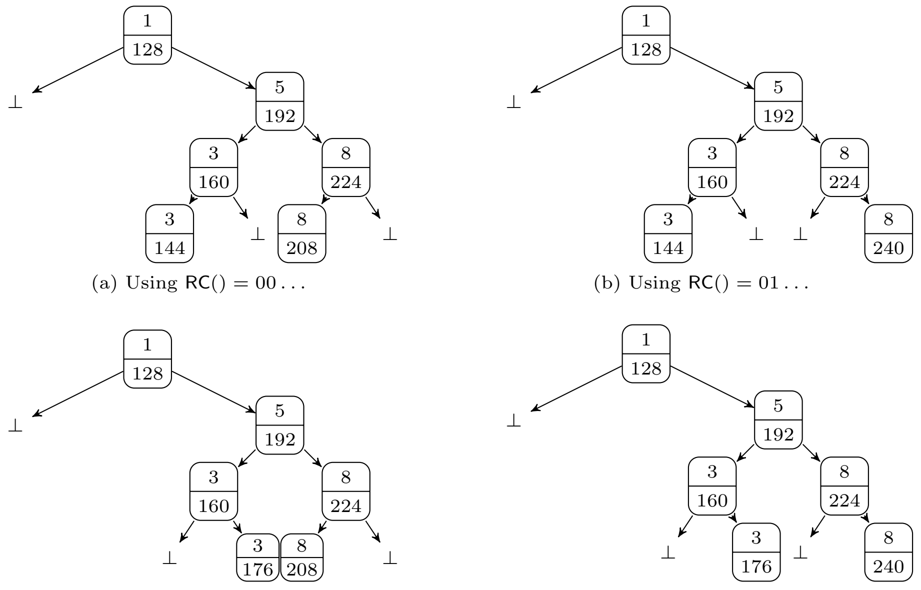

1
128
5
192
3
160
8
224
3
144
8
208
8
240
3
176
⊥
(a) Using $\mathsf{RC}() = 00\ldots$
(b) Using $\mathsf{RC}() = 01\ldots$
(c) Using $\mathsf{RC}() = 10\ldots$
(d) Using $\mathsf{RC}() = 11\ldots$

Fig. 1. An example for different binary search trees after inserting the sequence $X = 1, 5, 3, 8, 3, 8$, depending on the output of RC.

Example 1. To simplify the access to the construction, we review a detailed example. Figure 1 shows the four possible resulting binary search trees after inserting $X = 1, 5, 3, 8, 3, 8$, depending on the output of RC. We use a ciphertext space of $\{1, \ldots, 256\}$. Each different output of RC corresponds to one of the four possible randomized orders Γ_i for $1 \leq i \leq 4$.

Security. The scheme is proven secure against frequency-analyzing ordered chosen plaintext attack. To this end, [11] constructs a simulator which, given the two challenge plaintext sequences, produces identical views independent of which of two sequences is chosen. We investigate on the proof in the next section.

4 An Attack on Kerschbaum's FH-OPE Scheme

In this section, we investigate on the security achieved by Kerschbaum's construction [11]. In order to start, we observe that Kerschbaum proves his construction secure. However, as we show later in this section, the security proof makes an extra assumption on the game challenger's capabilities, namely that the challenger can dictate the randomized order used by the encryption algorithm to encrypt either challenge plaintext sequence. Using the natural interpretation of IND-FA-OCPA (see Definition 5), this additional assumption is not justified and, hence, Kerschbaum's scheme is no longer secure. We thus present a concrete attack, which is related to the distribution based on which randomized sequences are chosen for encryption (Section 4.1). We then explain more in detail why Kerschbaum's security result is incorrect with respect to Definition 5 (Section 4.2). Finally, we show that even if randomized orders are chosen uniformly at random, the scheme is still vulnerable (Section 4.3).

4.1 A Simple Attack

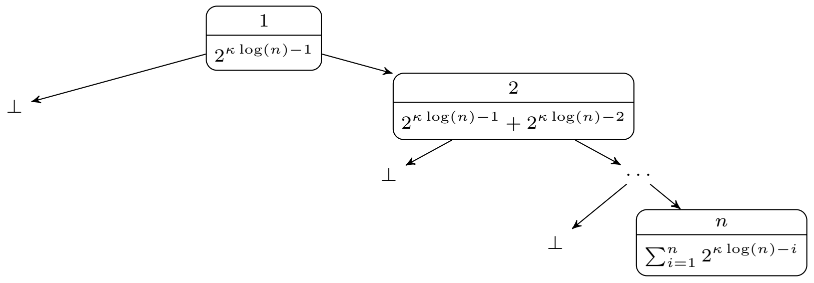

1
$2^{\kappa \log(n)-1}$
$\perp$
2
$2^{\kappa \log(n)-1} + 2^{\kappa \log(n)-2}$
$\perp$
$\cdots$
$\perp$
n
$\sum_{i=1}^{n} 2^{\kappa \log(n)-i}$

Fig. 2. The resulting binary search tree when encrypting sequence X_0.

Our attack consists of two plaintext sequences that are given to the challenger of the FA-OCPA game, who encrypts step-by-step randomly one of the two sequences. By observing the sequence of resulting ciphertexts, we will be able to determine which sequence the challenger chose with very high probability.

Consider the two plaintext sequences $X_0 = 1, 2, 3, \ldots, n$ and $X_1 = 1, \ldots, 1$ such that $|X_0| = |X_1| = n$. Clearly, both X_0 and X_1 have a common randomized order, namely $\Gamma = 1, 2, 3, \ldots, n$, i.e., the identity function applied to X_0. Moreover, consider the binary search tree produced by the scheme when encrypting X_0 in Figure 2. This tree is generated fully deterministically since the elements in X_0 are pairwise distinct, or equivalently, X_0 has only a single randomized order. Now let us investigate how X_1 would be encrypted by Algorithm 1. In every step, the RC procedure has to be called in order to decide where to insert the incoming element. If coins are drawn uniformly at random then only with a probability of $1/2^{n(n-1)/2}$ RC will produce the bit sequence $1 \ldots 1$ of length $n(n-1)/2$, which is required in order to produce the exact same tree as in Figure 2. Notice that the RC sequence must be of length $n(n-1)/2$ since for every plaintext that is inserted on the right, the function has to be called once more. Hence, the Gaussian sum $\sum_{i=1}^{n-1} i$ describes the number of required bits. Consequently, an adversary challenging the FH-OCPA challenger with X_0 and X_1 will win the game with probability $(1/2)(1 - 1/2^{n(n-1)/2})$ where the factor $1/2$ accounts for the probability that the challenger chooses X_1. Hence, if X_1 is chosen, our attacker wins with overwhelming probability, otherwise he has to guess. Notice that the combined probability is nevertheless non-negligible.

In conclusion, the central observation is that the number of calls to RC strongly depends on how many equal elements are met on the way to the final destination when encrypting an element. Therefore, not all ciphertext trees are equally likely.

4.2 Understanding the Problem

In this section, we analyze the core of the problem.

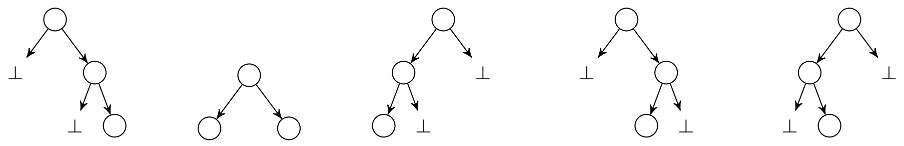
Artifacts of the Construction. The analysis in the previous section shows that randomized orders are not drawn uniformly random. Otherwise, the adversary's success probability would be $(1/2)(1 - 1/n!)$ since X_1 has $n!$ many randomized orders and the probability that a specific one is chosen uniformly is $1/n!$. Instead, we analyzed the probability that the *encryption algorithm* chooses that specific randomized order, which depends on the number of calls to RC and its results, which should all be 1.

In order to exemplify this artifact, we consider the sequence $1, 1, 1$. We depict the different trees when encrypting the sequence in Figure 3. As we can see, different trees require a different number of calls to RC, and have thus a different probability of being the encryption tree. On the one hand, the trees in Figure 3a and Figure 3c–3e all have a probability of $1/8$ to be the result of the encryption since each of them requires RC to be called three times. On the other hand, the two trees in Figure 3b have a probability of $1/4$ of being chosen each since RC has to be called only twice.

To formally capture the probability range of different randomized orders, we want to understand which randomized orders are most probable and which ones are least probable. Before we start the analysis, we observe that it does not matter whether we consider the probability or the number of calls to RC, since every call to RC adds a factor of $1/2$ to the probability that a certain randomized order is chosen. So as we have seen in the concrete counter-example in the previous section and the example above, a tree with linear depth represents the least likely randomized order since it requires the most calls to RC, which increases by one every time a new element is encrypted. Conversely, randomized orders represented by a perfectly balanced binary tree are more likely since they require the minimum number of calls to RC. Let H be the histogram of plaintext occurrences in a sequence. Then, as before, the number of calls to RC can be computed as the sum over every node's depth in the subtree in which all duplicate elements reside, which is at least

$$\begin{aligned}\sum_{p\in X}\frac{\sum_{i=1}^{H(p)}\log(i)}{H(p)} &= \sum_{p\in X}\frac{\log(H(p)!)}{H(p)} \geq \sum_{p\in X}\frac{\frac{H(p)}{2}\log\left(\frac{H(p)}{2}\right)}{H(p)}\\ &= -\frac{|X|}{2}+\frac{1}{2}\sum_{p\in X}\log(H(p))\end{aligned}$$

where we make use of the fact that $\left(\frac{n}{2}\right)^{\frac{n}{2}} \leq n! \leq n^n$ in the first inequality. All other randomized orders lie probability-wise somewhere in between.



(a) $\Gamma_1 = (1,2,3)$, $\mathsf{RC}() = 111$.

(b) $\Gamma_2 = (2,1,3)$, $\mathsf{RC}() = 01$, and $\Gamma_3 = (2,3,1)$, $\mathsf{RC}() = 10$.

(c) $\Gamma_4 = (3,2,1)$, $\mathsf{RC}() = 000$.

(d) $\Gamma_5 = (1,3,2)$, $\mathsf{RC}() = 110$.

(e) $\Gamma_6 = (3,1,2)$, $\mathsf{RC}() = 001$.

Fig. 3. The trees displaying different randomized orders for the sequence $1, 1, 1$.

Proof Technique. Despite our counter-example, the scheme is proven secure in the programmable random oracle model [11]; this obviously constitutes a contradiction. To understand the problem in depth, we have to have a closer look at the security proof. The idea behind the proof is as follows: the challenger selects uniformly at random a common randomized order with respect to the two challenge sequences. This common randomized order is then given as source of randomness to the random oracle which answers questions accordingly. More precisely, let $\Gamma = (\gamma_1, \ldots, \gamma_n)$ be the selected order. Whenever the algorithm cannot decide where to place a plaintext x_j in the search tree, i.e., $x_i = x_j$ for $i < j$, meaning that x_i is an entry that is already encrypted in the tree, then the challenger asks the random oracle for a decision on i and j. The oracle answers with 1 if $\gamma_i < \gamma_j$ and with 0 if $\gamma_i > \gamma_j$ (notice that $\gamma_i \neq \gamma_j$ by Definition 4). In this way, the challenger produces a search tree and corresponding ciphertexts that are valid for both challenge sequences and which are in fact independent of the bit. Hence, the adversary has no chance of determining which sequence has been chosen other than guessing.

The Simulation is Incorrect. The proof strategy clearly excludes the attack described in the previous section. The reason is that the RC function is supposed to output *uniformly random* coins. As we have seen, even if RC outputs truly random coins then not every possible randomized order of the chosen sequence is equally likely. Hence, the choice of the random sequence is in two aspects unfaithful: first, the challenger restricts the number of randomized orders to those that both sequences have in common while RC does not know the two sequences and can, hence, not choose according to this requirement. Second, the fact that the choice is uniform does not reflect the reality. As we have seen, one artifact of the construction is that not every randomized order is equally likely. Consequently, forcing RC to generate output based on a common randomized order changes the distribution from which coins are drawn and, hence, neither all randomized orders are possible nor are their probabilities of being chosen correctly distributed. In the extreme case described in Section 4.1, it even would disallow all but one randomized order to be the result of the encryption routine. As a consequence, the proof technique changes the behavior of RC to an extent that makes the simulation incorrect.

4.3 Generalizing the Attack in an Ideal Setting

Since the scheme is vulnerable to an attack and it chooses the randomized order under which a sequence is encrypted in a non-uniform way, we find it interesting to also investigate whether the scheme is still vulnerable in an ideal setting where the choice of the randomized order happens uniformly.

The answer to this question is unfortunately positive, as the following result shows. Concretely, only if two sequences agree on essentially all randomized orders, the adversary has a negligible advantage of distinguishing them.

Theorem 1. *Let X_0 and X_1 be two plaintext sequences of length n. Further assume that X_0 has m_0 and X_1 has m_1 randomized orders, respectively, and that they have m randomized orders in common. Then, for the idealized construction of [11] which encrypts plaintexts under a uniformly chosen randomized order, there exists an adversary whose success probability in winning the IND-FA-OCPA game is at least $1 - m\frac{m_0+m_1}{2m_0m_1}$.*

Proof. We construct an adversary, which submits both X_0 and X_1 to the FA-OCPA challenger. Since X_0 has m_0 randomized orders, the probability that one of those in common with X_1 is chosen by the encryption procedure is $\frac{m}{m_0}$ due to the uniformly random behavior. Likewise, for X_1, the probability that a common randomized order is chosen by the encryption procedure is $\frac{m}{m_1}$. Hence, depending on the challenger's bit b, the adversary sees a non-common randomized order with probability $1 - \frac{m}{m_b}$, which also reflects its success probability for winning the game when the challenger picks b. Consequently,

$$\begin{aligned}\mathsf{Pr}[\mathcal{A} \text{ wins}] &= \left|\mathsf{Pr}[\mathsf{Exp}^{\mathcal{A}}_{\mathsf{FA-OCPA}}(\kappa, 1) = 1] - \mathsf{Pr}[\mathsf{Exp}^{\mathcal{A}}_{\mathsf{FA-OCPA}}(\kappa, 0) = 1]\right| \\ &= \frac{1}{2}\left(1 - \frac{m}{m_0}\right) + \frac{1}{2}\left(1 - \frac{m}{m_1}\right) = 1 - m\frac{m_0 + m_1}{2m_0 m_1}\end{aligned}$$

In the example from the previous section, we have parameters $m_0 = 1$, $m_1 = n!$, and $m = 1$. Substituting those into Theorem 1, we get the aforementioned non-negligible success probability of

$$1 - m\frac{m_0 + m_1}{2m_0 m_1} = 1 - \frac{1 + n!}{2n!} = \frac{1}{2}\left(1 - \frac{1}{n!}\right).$$

5 Impossibility of IND-FA-OCPA

The previously presented results raise the question if IND-FA-OCPA, as presented in [11] can be achieved at all. It turns out that this is not the case: in this section, we prove an impossibility result for frequency-hiding order-preserving encryption as defined in Definition 5. Formally we prove the following theorem.

Theorem 2. *Let X_0 and X_1 be two arbitrary plaintext sequences of the same length that do not share any randomized order. Let furthermore $\mathcal{OPE}$ be an order-preserving encryption scheme secure against IND-FA-OCPA. Then, the probability of distinguishing whether X_0 is encrypted or whether X_1 is encrypted with $\mathcal{OPE}$ is negligibly close to 1/2.*

Before we prove the theorem using Definition 5, we argue why it implies the impossibility of frequency-hiding OPE. According to the theorem, no adversary can distinguish the encryptions of two arbitrary sequences of his choice that are ordered in a completely different manner. This constitutes a formulation of the IND-CPA property for multiple messages, restricted to sequences that do not share a randomized order. The restriction, however, is not necessary since sequences that share a randomized order are trivially indistinguishable by IND-FA-OCPA. To exemplify, let the two sequences be $X_0 = 1, 2, \ldots, n$ and $X_1 = n, n-1, \ldots, 1$, so X_1 is the reverse of X_0. According to Theorem 2, no adversary can distinguish which one of the two is encrypted. However, due to the correctness of OPE it must be the case that the encryption Y^* fulfills for all i and j and $b \in \{0, 1\}$

$$y^*_i \geq y^*_j \implies x^b_i \geq x^b_j.$$

Consequently, if X_0 is encrypted we have $y^*_i < y^*_j$ for $i < j$ and vice versa for X_1. Hence, an adversary could trivially distinguish which of the two sequences is encrypted. Hence, by contraposition of Theorem 2, an IND-FA-OCPA-secure OPE scheme cannot exist.

Proof (Theorem 2). In the security game $G(b)$, on input X_0 and X_1 by $\mathcal{A}$, the challenger encrypts sequence X_b, gives the ciphertexts Y^* to $\mathcal{A}$, who replies with a guess b'. $\mathcal{A}$ wins if $b = b'$.

We define three games. $G_1 = G(0)$. For G_2, we select a sequence X^* which has a randomized order in common with both X_0 and X_1. Notice that such a sequence always exists, e.g., take the series $a, a, \ldots, a$ (n times) for arbitrary a in an appropriate domain. Instead of encrypting X_0 as in G_1, we now encrypt X^*. Finally, $G_3 = G(1)$.

In order to show that $G_1 \approx G_2$, assume that there exists a distinguisher $\mathcal{A}$ that can distinguish between G_1 and G_2 with non-negligible probability. Then we construct a reduction $\mathcal{B}$ that breaks IND-FA-OCPA. On $\mathcal{A}$'s input, $\mathcal{B}$ forwards X_0 and a sequence X^* to the IND-FA-OCPA challenger. The challenger answers with Y^*, which $\mathcal{B}$ again forwards to $\mathcal{A}$. $\mathcal{A}$ outputs a bit b', which $\mathcal{B}$ again forwards to the challenger. The simulation is obviously efficient. If the internal bit of the IND-FA-OCPA challenger is 0, we perfectly simulate G_1, while we simulate G_2 when the bit is 1. Hence, the success probability of $\mathcal{A}$ carries over to $\mathcal{B}$ since $\mathcal{B}$ only forwards messages. Since we assumed that $\mathcal{A}$ can successfully distinguish G_1 and G_2 with non-negligible probability, it must be the case that $\mathcal{B}$ wins the IND-FA-OCPA game with non-negligible probability. This is a contradiction.

The proof of $G_2 \approx G_3$ is symmetric to the one above. In conclusion, we have that $G_1 \approx G_2 \approx G_3$, and hence, $G(0) \approx G(1)$, meaning that every adversary can distinguish between encryptions of X_0 and X_1 that do not share a randomized order only with negligible probability.

6 An Achievable Definition: IND-FA-OCPA*

Since the notion of indistinguishable ciphertexts under frequency-analyzing ordered chosen plaintext attacks is not achievable, it is desirable to understand the problem of the original definition and try to come up with a suitable one that still captures the idea of frequency-hiding but is achievable.

Interestingly enough, the solution to our problem can be found by investigating again Kerschbaum's security proof. The proof builds a random oracle that overcomes the issues of the definition. Even though this construction of the oracle is incorrect, as we have shown previously, it helps us identify the problem with the definition. In the definition, the challenger has no means to tell the encryption algorithm which randomized order to choose when encrypting the chosen challenge sequence. Hence, it could be the case that the algorithm chooses an order that is not common to both challenge sequences. Had the challenger a way to decide which order to encrypt with, the problem were gone.

Consequently, we tackle the problem from two angles: (1) we augment the OPE model by one more input to the encryption function, namely, the randomized order that is supposed to be used and (2) we strengthen the challenger's capabilities during the security game: it may now, additionally to selecting which sequence to encrypt, also choose a common randomized order as input to the encryption algorithm. This new definition still captures the notion of frequency-hiding in the same way, it just excludes the attacks presented in this work and makes the definition, thus, achievable.

6.1 Augmented OPE Model

We present the augmented model in the following definition. Notice that the only difference to Definition 1 is the additional input Γ to the encryption function. This additional input serves the purpose of deciding from outside of the function, which randomized order should be used to encrypt the plaintexts. In contrast, standard OPE decides about the ordering randomly inside of the function. We stress that augmented OPE is more general than OPE since the input Γ can be replaced by the result of a call to a random function.

Definition 6 (Augmented OPE). *An* augmented order-preserving encryption scheme $\mathcal{OPE}^* = (\mathsf{K}, \mathsf{E}, \mathsf{D})$ *is a tuple of* PPT *algorithms where*

$S \leftarrow \mathsf{K}(\kappa)$. *The* key generation *algorithm takes as input a security parameter κ and outputs a secret key (or state) S;*

$(S', y) \leftarrow \mathsf{E}(S, x, \Gamma)$. *The* encryption *algorithm takes as input a secret key S, a message x, and an order Γ and outputs a new key S' and a ciphertext y;*

$x \leftarrow \mathsf{D}(S, y)$. *The* decryption *algorithm $x \leftarrow \mathsf{D}(S, y)$ takes as input a secret key S and a ciphertext y and outputs a message x.*

6.2 The New Definition IND-FA-OCPA*

The new security game is close in spirit to Definition 5. The difference is that (1) it is defined over an augmented OPE scheme which makes the randomized order used for encryption explicit and (2) the challenger chooses that order uniformly at random from the orders that both challenge sequences have in common. Since we define the notion adaptively, we introduce some new notation with respect to randomized orders.

In the following definition, we let $\Gamma = \gamma_1, \ldots, \gamma_n$ and we use the notation $\Gamma{\downarrow_i}$ to denote the order of the sequence $\gamma_1, \ldots, \gamma_i$. Notice that this order is unique since Γ is already an order. For instance, take the randomized sequence $\Gamma = 1, 6, 4, 3, 2, 5$. Then, $\Gamma{\downarrow_3} = 1, 3, 2$, which is the order of $1, 6, 4$.

Definition 7 (IND-FA-OCPA*). *An augmented order-preserving encryption scheme $\mathcal{OPE}^* = (\mathsf{K}, \mathsf{E}, \mathsf{D})$ has* indistinguishable ciphertexts under frequency-analyzing ordered chosen plaintext attacks *if for any* PPT *adversary $\mathcal{A}$, the following probability is negligible in the security parameter κ:*

$$\left|\Pr[\mathsf{Exp}^{\mathcal{A}}_{\mathsf{FA-OCPA^*}}(\kappa, 1) = 1] - \Pr[\mathsf{Exp}^{\mathcal{A}}_{\mathsf{FA-OCPA^*}}(\kappa, 0) = 1]\right|$$

where $\mathsf{Exp}^{\mathcal{A}}_{\mathsf{FA-OCPA^*}}(\kappa, b)$ is the following experiment:

Experiment $\mathsf{Exp}^{\mathcal{A}}_{\mathsf{FA-OCPA}^*}(\kappa, b)$

$(X_0, X_1) \leftarrow \mathcal{A}$ where $|X_0| = |X_1| = n$ and X_0 and X_1
have at least one common randomized order
Select Γ uniformly at random from the common randomized orders of X_0, X_1
$S_0 \leftarrow \mathsf{K}(\kappa)$
For all $1 \le i \le n$ run $(S_i, y_{b,i}) \leftarrow \mathsf{E}(S_{i-1}, x_{b,i}, \Gamma{\downarrow_i})$
$b' \leftarrow \mathcal{A}(y_{b,1}, \dots, y_{b,n})$
Output 1 if and only if $b = b'$.

7 Constructing Augmented OPE

We show how to construct an augmented OPE scheme. Interestingly enough, the key observation to $\mathcal{OPE}^*$ is that the scheme of [11], which we presented in Section 3 can be modified so as to fit the new model.

As we introduce a third input to the encryption function, namely an order that is as long as the currently encrypted sequence plus one, we have to show how to cope with this new input in the construction. The key idea is quite simple: usually, the encryption scheme draws randomness from a PRF when the plaintext to be encrypted is already encrypted, in order to decide whether to move left or right further in the tree. The additional input solves this decision problem upfront, so there is no need for using randomness during the encryption.

While the setup and re-balancing algorithms are as described in [11], we describe the new encryption algorithm in Algorithm 2. Furthermore, we require that every node in the tree stores its index in the plaintext sequence, i.e., we add an attribute *index* to each node t. We further assume that the index of the message that is currently to be encrypted is the length of the order Γ. As we can see, the only difference between Algorithm 1 and Algorithm 2 is the behavior when the message to be inserted is equal to the message currently stored at t. Then, the order Γ is considered so as to decide whether to traverse the tree further to the left or right.

Algorithm 2 $\mathsf{E}(S, x, \Gamma)$ where $S = t, min, max$ and $\Gamma = \gamma_1, \dots, \gamma_k$

```
1:  if $t = \perp$ then
2:      $t.m = x$
3:      $t.index = k$
4:      $t.c = min + \lfloor \frac{max - min}{2} \rfloor$
5:      if $t.c = 0$ then
6:          rebalance the tree
7:      end if
8:      return $t.c$
9:  end if
10: $b \leftarrow -1$
11: if $x = t.m$ then
12:     $b \leftarrow \gamma_k > \gamma_{t.index}$
13: end if
14: if $b = 1 \vee x > t.m$ then
15:     $\mathsf{E}(t.right, t.c + 1, max, x, \Gamma)$
16: else
17:     if $b = 0 \vee x < t.m$ then
18:         $\mathsf{E}(t.left, min, t.c - 1, x, \Gamma)$
19:     end if
20: end if
```

The encryption algorithm also nicely demonstrates that it does not matter from which domain the ordering draws its elements. The only important property of such an ordering is the relation of the single elements to each other, i.e., that (1) all elements of the order are distinct and (2) whether $\gamma_i < \gamma_j$ or the other way around. We do, hence, not require the shrinking function $\Gamma{\downarrow_i}$ for this construction: when encrypting an overall sequence of plaintexts with a predetermined randomized order Γ, it is sufficient to just cut the Γ to size i when encrypting the i-th element. The reason is that the relative ordering of the first i elements is not changed after shrinking, which suffices to let the algorithm decide about where to branch.

7.1 Formal Guarantees

We argue in this section that the tree-based scheme presented in the previous section is IND-FA-OCPA* secure. The reason for that is as follows: first, the challenger can select a randomized order that is in common to both sequences and give that chosen order to the encryption algorithm. Second, no matter which of the two sequences is encrypted according to the order, the resulting ciphertexts are equivalent in both cases. Hence, the adversary cannot do better than guessing the bit, since the ciphertexts are independent of the underlying plaintexts.

Theorem 3. *The $\mathcal{OPE}^*$ scheme presented in Section 7 is* IND-FA-OCPA* *secure.*

Proof. Let $\mathcal{A}$ be an arbitrary adversary for the game in Definition 7. Let furthermore X_0 and X_1 be the two plaintext sequences chosen by $\mathcal{A}$. By definition, those sequences share at least one common randomized order. Let Γ be one of those common orders, selected uniformly at random from the universe of common randomized orders. When encrypting either X_0 or X_1, Algorithm 2 uses Γ to decide where to branch. Hence, Algorithm 2's decisions are independent of the input plaintext sequence, and thus independent of the chosen bit b. Consequently, all information that $\mathcal{A}$ receives from the challenger are independent of b and he can thus, only guess what b is. This concludes the proof.

8 Conclusion

Order-preserving encryption (OPE) is an enabling technology to implement database applications on encrypted data: the idea is that the ordering of ciphertexts matches the one of plaintexts so that inequalities on encrypted data are efficiently computable. Unfortunately, recent works showed that various attacks can be mounted by exploiting the inherent leakage of plaintext frequency. Frequency-hiding OPE [11] (CCS 2015) is a stronger primitive that aims at solving this problem by hiding the frequency of plaintexts.

We contribute to this line of work with the following results. First, we present an attack against the construction presented in [11], identifying the corresponding problem in the security proof. Second, we formulate a more general impossibility result, proving that the security definition introduced in [11] cannot be achieved by any OPE scheme. Third, to complete the picture and assess which theoretical security is achievable at all, we make the definition in [11] more precise by giving the challenger more capabilities and augmenting the OPE model so as to receive randomized orders as inputs which are used to break ties. We finally show that the more precise version of the definition can be achieved by a variant of the construction introduced in [11].

Despite this seemingly positive results, in the presence of the plethora of empirical attacks against (FH-)OPE and its variants (e.g., ORE), we suggest to not use any of those schemes for actual deployment since the security guarantees achieved do not reflect practical requirements. We recommend to move away from OPE in general, more towards other alternatives, even if there are none that solve the problem so conveniently; at the price of low to no security.

Acknowledgements. This research is based upon work supported by the German research foundation (DFG) through the collaborative research center 1223, by the German Federal Ministry of Education and Research (BMBF) through the Center for IT-Security, Privacy and Accountability (CISPA), and by the state of Bavaria at the Nuremberg Campus of Technology (NCT). NCT is a research cooperation between the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and the Technische Hochschule Nürnberg Georg Simon Ohm (THN). Dominique Schröder is supported by the German Federal Ministry of Education and Research (BMBF) through funding for the project PROMISE. Finally, we thank the reviewers for their helpful comments and our shepherd for the excellent and valuable feedback, which improved the paper.

References

1. Arasu, A., Blanas, S., Eguro, K., Kaushik, R., Kossmann, D., Ramamurthy, R., Venkatesan, R.: Orthogonal Security With Cipherbase. In: Proc. Biennial Conference on Innovative Data Systems Research (CIDR'13) (2013)
2. Bellare, M., Keelveedhi, S., Ristenpart, T.: DupLESS: Server-Aided Encryption for Deduplicated Storage. In: Proc. USENIX Security Symposium (USENIX'13). pp. 179–194. USENIX Association (2013)
3. Boldyreva, A., Chenette, N., Lee, Y., O'Neill, A.: Order-Preserving Symmetric Encryption. In: Proc. Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT'09). pp. 224–241. Lecture Notes in Computer Science, Springer Verlag (2009)
4. Boldyreva, A., Chenette, N., O'Neill, A.: Order-Preserving Encryption Revisited: Improved Security Analysis and Alternative Solutions. In: Proc. Advances in Cryptology (CRYPTO'11). pp. 578–595. Springer Verlag (2011)
5. Boneh, D., Lewi, K., Raykova, M., Sahai, A., Zhandry, M., Zimmerman, J.: Semantically secure order-revealing encryption: Multi-input functional encryption without obfuscation. In: Proc. Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT'15). pp. 563–594. Springer Verlag (2015)
6. Chenette, N., Lewi, K., Weis, S.A., Wu, D.J.: Practical Order-Revealing Encryption with Limited Leakage. In: Peyrin, T. (ed.) Proc. Fast Software Encryption (FSE'16). pp. 474–493. Springer Verlag (2016)

7. Durak, F.B., DuBuisson, T.M., Cash, D.: What Else is Revealed by Order-Revealing Encryption? In: Proc. Conference on Computer and Communications Security (CCS'16). pp. 1155–1166. ACM Press (2016)
8. Grubbs, P., McPherson, R., Naveed, M., Ristenpart, T., Shmatikov, V.: Breaking Web Applications Built On Top of Encrypted Data. In: Proc. Conference on Computer and Communications Security (CCS'16). pp. 1353–1364. ACM Press (2016)
9. Grubbs, P., Sekniqi, K., Bindschaedler, V., Naveed, M., Ristenpart, T.: Leakage-Abuse Attacks against Order-Revealing Encryption. In: Proc. IEEE Symposium on Security & Privacy (S&P'17). IEEE Computer Society Press (2017)
10. He, W., Akhawe, D., Jain, S., Shi, E., Song, D.: ShadowCrypt: Encrypted Web Applications for Everyone. In: Proc. Conference on Computer and Communications Security (CCS'14). pp. 1028–1039. ACM Press (2014)
11. Kerschbaum, F.: Frequency-Hiding Order-Preserving Encryption. In: Proc. Conference on Computer and Communications Security (CCS'15). pp. 656–667. ACM Press (2015)
12. Kerschbaum, F., Schroepfer, A.: Optimal Average-Complexity Ideal-Security Order-Preserving Encryption. In: Proc. Conference on Computer and Communications Security (CCS'14). pp. 275–286. ACM Press (2014)
13. Lau, B., Chung, S., Song, C., Jang, Y., Lee, W., Boldyreva, A.: Mimesis Aegis: A Mimicry Privacy Shield–A System's Approach to Data Privacy on Public Cloud. In: Proc. USENIX Security Symposium (USENIX'14). pp. 33–48. USENIX Association (2014)
14. Lewi, K., Wu, D.J.: Order-Revealing Encryption: New Constructions, Applications, and Lower Bounds. In: Proc. Conference on Computer and Communications Security (CCS'16). pp. 1167–1178. ACM Press (2016)
15. Naveed, M., Kamara, S., Wright, C.V.: Inference Attacks on Property-Preserving Encrypted Databases. In: Proc. Conference on Computer and Communications Security (CCS'15). pp. 644–655. ACM Press (2015)
16. Popa, R.A., Li, F.H., Zeldovich, N.: An Ideal-Security Protocol for Order-Preserving Encoding. In: Proc. IEEE Symposium on Security & Privacy (S&P'13). pp. 463–477. IEEE Computer Society Press (2013)
17. Popa, R.A., Redfield, C.M.S., Zeldovich, N., Balakrishnan, H.: CryptDB: Protecting Confidentiality with Encrypted Query Processing. In: Proc. ACM Symposium on Operating Systems Principles (SOSP'11). pp. 85–100. ACM Press (2011)
18. Popa, R.A., Stark, E., Valdez, S., Helfer, J., Zeldovich, N., Balakrishnan, H.: Building Web Applications on Top of Encrypted Data Using Mylar. In: Proc. USENIX Symposium on Networked Systems Design and Implementation (NSDI'14). pp. 157–172. USENIX Association (2014)
19. Roche, D.S., Apon, D., Choi, S.G., Yerukhimovich, A.: POPE: Partial Order Preserving Encoding. In: Proc. Conference on Computer and Communications Security (CCS'16). pp. 1131–1142. ACM Press (2016)
20. Song, D.X., Wagner, D., Perrig, A.: Practical Techniques for Searches on Encrypted Data. In: Proc. IEEE Symposium on Security and Privacy (S&P'00). pp. 44–55. IEEE Computer Society Press (2000)
21. Teranishi, I., Yung, M., Malkin, T.: Order-Preserving Encryption Secure Beyond One-Wayness, pp. 42–61. Springer Verlag (2014)