Reducing the Leakage in Practical Order-Revealing Encryption

David Cash
Rutgers University

Feng-Hao Liu
Florida Atlantic University

Adam O'Neill*
Georgetown University

Cong Zhang
Rutgers University

ABSTRACT

We study practical order-revealing encryption (ORE) with a well-defined leakage profile (the information revealed about the plaintexts from their ciphertexts), a direction recently initiated by Chenette, Lewi, Weis, and Wu (CLWW). ORE, which allows public comparison of plaintext order via their ciphertexts, is a useful tool in the design of secure outsourced database systems. We first show a general construction of ORE with reduced leakage as compared to CLWW, by combining ideas from their scheme with a new type of "property-preserving" hash function. We then show how to construct such a hash function efficiently based on bilinear maps. Our resulting ORE scheme is fairly practical: for n-bit plaintexts, ciphertexts consists of about $4n$ group elements, and order comparison requires about n^2 pairings. The leakage is, roughly speaking, the "equality pattern" of the most-significant differing bits, whereas CLWW's is the location and values of the most-significant differing bits. We also provide a generalization of our scheme that improves the leakage and/or efficiency.

To analyze the quality of our leakage profile, we show several additional results. In particular, we show that order-*preserving* (OPE) encryption, an important special case of ORE scheme in which ciphertexts are ordered, cannot be secure wrt. our leakage profile. This implies that our ORE scheme is the first one without multilinear maps that is proven secure wrt. a leakage profile unachievable by OPE. We also also show that our generalized scheme meets a "semantically meaningful" one-wayness notion that schemes with the leakage of CLWW do not.

Keywords

Encryption, Order-revealing encryption

1. INTRODUCTION

An emerging area of cryptography concerns the design and analysis of "leaky" protocols (see *e.g.* [28, 31, 11] and additional references below), which are protocols that deliberately give up some security in order to achieve practical efficiency. A domain of particular practical importance in which this is being pursued is that of searching on encrypted data (aka. outsourced database protocols); see [24] for an excellent overview. In this setting, a client has a database whose storage it wishes to outsource to an untrusted server. The question how to construct protocols that allow efficient query processing by the server but at the same time maintain reasonable confidentiality of the client's data.

In this work, we focus on an important tool in this domain, *order-revealing encryption* [7, 8].[1] Order-revealing encryption (ORE) is symmetric encryption with an additional operation Comp, in which plaintexts are numbers from say 1 to M, and for any key K and any plaintexts $m_0 < m_1$, $\mathsf{Comp}(\mathcal{E}(m_0), \mathcal{E}(m_1))$ outputs 1. That is, order-revealing encryption allows public comparison of plaintexts order via their ciphertexts (note Comp does not take input K). To see how this could be useful in outsourced database protocols, consider a client with a simple relational database consisting of one table supporting range queries. Suppose the server encrypts the search key value in each row under ORE and the remainder of the row under a standard (semantically secure) encryption scheme. When the server receives the this encrypted database it can index it using *e.g.* a B-tree and process a client's encrypted range queries efficiently, as this only requires comparing search key values.

Constructing ORE is challenging. Initial results (which actually predated the notion) focused on the special case of order-*preserving* encryption (OPE) [1, 6], in which Comp is simple numerical comparison. Unfortunately, OPE leaks much more information that just plaintext order (which is to some extent inherent, see below). Later, Boneh, Lewi, Raykova, Sahai, Zhandry, and Zimmerman [8] gave a construction of ORE leaking only plaintext order (which is the "ideal"' leakage here), based on multilinear maps. While this is an encouraging feasibility result, it is a far cry from practical. On the other hand, Chenette, Lewi, Weis and Wu (CLWW) [15] recently gave a ORE practical construction leaking the most significant differing-bits of the plaintexts. We view construction as a valuable data point in the design space, but it still leaks a lot of information. To what extent can this leakage be reduced while retaining practicality?

1.1 Our Results

*Work done in part while visiting Florida Atlantic University.

[1] In [7], it was called efficiently-orderable encryption.

ORE from Property-Preserving Hashing. Our first main result is a new way to build on the scheme of CLWW to reduce the leakage. To explain the idea, we first briefly recall the scheme of CLWW. In their (basic) scheme, the encryption key is a key K for a pseudorandom function PRF. To encrypt a plaintext x of length n, for each prefix p_i of x one computes $y_i = \mathsf{PRF}_K(p_i) + x_{i+1}$ where x_{i+1} is the $(i+1)$-st bit of x; the resulting ciphertext is $(y_1 \ldots, y_n)$. Now to compare two ciphertexts $(y_1 \ldots, y_n)$ and $(y'_1 \ldots, y'_n)$, one finds the smallest index i such that $y_i \neq y'_i$, and outputs 1 if $y'_i - y_i = 1$. To understand our approach for reducing the leakage, first consider what happens if we modify the scheme to not output $(y_1 \ldots, y_n)$ as the ciphertext but a *random permutation* of these elements, chosen fresh for each encryption. We can still compare ciphertexts by appropriately modifying the comparison algorithm: now given $c = (y_1 \ldots, y_n)$ and $c' = (y'_1 \ldots, y'_n)$ (permuted as above), it will look for indices i, j such that either $y'_i - y_j = 1$, in which case it outputs 1, or $y_j - y'_i = 1$, in which case it outputs 0. However, the leakage is not reduced: an adversary can still determine the most significant differing bit by counting how many elements c and c' have in common.

To solve this issue, we employ a new notion of *property-preserving hashing* (PPH) we introduce. PPH can be seen as the hashing (meaning, no decryption) analogue of the notion of property-preserving encryption, a generalization of order-revealing encryption to arbitrary properties due to Pandey and Rouselakis [30]. (This can also be seen as a symmetric-key version of the notion of "relational hash" due to Mandal and Roy [27].) Specifically, we employ a PPH for the property $P_1(x, x') = 1; x = x' + 1$. (Here x, x' are not plaintexts of the ORE scheme, think of them as other inputs determined below.) Security requires that this is *all* that is leaked; in particular, input equality is *not* leaked by the hash values (which requires a randomized hashing algorithm). Now, the idea is to modify the scheme to include a key K_H for such a PPH $\mathcal{H}$, and the encryption algorithm to not only randomly permute the y_i's but hash them as well, i.e., output $(h_1, \ldots, h_n)$ where $h_i = \mathcal{H}_{K_H}(y_i)$ for the permuted y_i's.[2] The comparison algorithm can again be modified appropriately, namely to not to check if $y'_i - y_j = 1$ but rather if their h'_i and h'_j hash values satisfy P_1 via the PPH (and similarly for the check $y_j - y'_i = 1$).

Intuitively, the resulting ORE scheme leaks only the *equality pattern* of most significant differing bits in addition to plaintext order, not the values or locations of these bits. Said another way, for any triple of plaintexts m_1, m_2, m_3, it leaks whether m_2 differs from m_1 (in terms of the most significant differing bit) *before* m_3 does, meaning the index of the most significant differing bit is smaller. However, proving this ORE scheme secure wrt. this leakage based on an achievable notion of security for the PPH turns out to be technically challenging. Nevertheless, manage to prove it "non-adaptively secure," meaning the adversary is required to non-adaptively choose the dataset, which is realistic for a passive adversary in the outsourced database setting.

Property-Preserving Hash from Bilinear Maps. Next we turn to constructing a practical property-preserving hash (PPH) for the property $P_1(x, x') = x = x' + 1$. For this, we adapt techniques from perfectly one-way hash functions [9, 27] to the symmetric-key setting and use asymmetric bilinear groups. Roughly, in our construction the key for the hash function is a key K for a pseudorandom function PRF and, letting $e\colon G_1 \times G_2 \to G_T$ be an asymmetric bilinear map on prime order cyclic groups G_1, G_2 with generators g_1, g_2, the hash of x is

$$\mathcal{H}_K(x) = (g_1^{r_1}, g_1^{r_1 \mathsf{PRF}_K(x)}, g_2^{r_2}, g_2^{r_2 \mathsf{PRF}_K(x+1)})$$

for fresh random $r_1, r_2 \in \mathbb{Z}_p$. (Thus, the PRF is also pushed to our PPH construction and can be dropped from from the higher-level ORE scheme when our hash function is plugged-in.) The bilinear map allows testing whether $P_1(x, x')$ from $\mathcal{H}_K(x), \mathcal{H}_K(x')$, and intuitively our use of asymmetric bilinear groups prevents testing other relations such as equality (formally we use the XSDH assumption). We prove the construction secure under an indistinguishability-based notion in which the adversary has to distinguish between the hash of a random challenge x^* and a random hash value, and can query for hash values of inputs x of its choice as long as $P_1(x, x^*)$ and $P_1(x^*, x)$ are both 0. Despite being restricted,[3], this notion suffices in our ORE scheme above.

When our PPH is plugged-in to our ORE scheme, the result is fairly practical: Ciphertexts consist of $4n$ group elements, and order comparison requires $n(n-1)$ pairing computations on average. We leave a detailed implementation and efficiency analysis to future work, but note that this level of efficiency is in-line with many schemes based on bilinear maps (e.g., attribute-based encryption schemes [19]) that are considered fairly practical. We also note that CLWW gave an improved version of their scheme is which ciphertexts are size $O(n + \lambda)$ rather than $O(n\lambda)$ for security parameter λ, however, we have reason to believe this may be difficult for schemes with our improved leakage profile, see below.

Generalizing our ORE Scheme. In order to further improve the efficiency and/or leakage of our ORE scheme, we introduce the following generalization. We view plaintexts as consisting of d-blocks of n/d bits each, and apply our basic construction "block-wise." This requires a PPH for the more general predicate $P_d(x, x') = x' \in \{x + 1, \ldots, x + 2^d - 1\}$ and leaks only the equality pattern of most-significant differing *blocks* (our basic scheme corresponds to $d = 1$). We also show how to extend our PPH construction to support $P_d(x, x')$, where the hash values consist of 2^{d+1} group elements, so the construction only supports small d. Interestingly, for $d = 2$ there is an efficiency improvement in the resulting ORE scheme as well: Ciphertexts are still $4n$ elements but comparison requires $(3/4)n(n-1)$ pairings (rather than $n(n-1)$) on average, and for $d = 3$ ciphertexts are slightly larger at $(16/3)n$ elements but comparison only requires $(7/9)n(n-1)$ pairings on average. For larger d the leakage continues to reduce but efficiency compared to the basic scheme (in terms of both ciphertext size and pairings required for comparison) decreases. Again, we leave a detailed implementation and efficiency analysis to future work.

[2] A minor issue here is that we now lose decryptability for the resulting ORE scheme; however, this can easily be added back in a generic way by also encrypting the plaintext separately under a semantically secure scheme.

[3] More generally, following [30] one could allow the adversary to choose two challenge inputs and make queries that do not allow it to trivially distinguish them, but we are unable to prove our construction secure under this stronger notion.

Analyzing the Leakage. In order to assess the quality of our leakage, we prove several auxiliary results. First, we show that an order-*preserving* encryption (OPE) scheme cannot be secure wrt. our leakage profile. Previously, [6] showed that an OPE scheme cannot be secure wrt. the ("ideal") leakage profile that only reveals the order of plaintexts. Interestingly, our proof uses entirely different techniques and is based on the observation that for two plaintexts our leakage is ideal. We also note that CLWW [15] *do* show an OPE scheme with their leakage profile. This in particular means our results give the first construction of ORE without multilinear maps secure wrt. a leakage profile not achievable by OPE. Furthermore, the fact that there is an OPE scheme with CLWW's leakage profile is a basis for their above-mentioned reduction in ciphertext length — in that CLWW reduce ciphertext length in their basic scheme, or rather a slight variation thereof, by making it "not OPE" — suggesting such a reduction in ciphertext length may difficult for our scheme (since it is already not OPE).

Next, we show that there are distributions on the database according to which schemes meeting the leakage profile of our generalized scheme is one-way, but schemes meeting the leakage of CLWW are not [15]. One-wayness is a "semantically meaningful" notion (see below) previously considered for order-preserving encryption in [7]. Finally, we initiate a combinatorial study of the structure of our leakage, showing that there at most $2^{m^2 \log n}$ possible leakage outputs for datasets consisting of m plaintexts of length n.

We view these results as only a first step in understanding our leakage profile, setting the stage for future work. Indeed, much work remains to be done on how to assess leakage profiles in the area of leaky cryptography more generally.

1.2 Discussion and Perspective

We follow CLWW in that we study practical ORE wrt. to a "well-defined" (*i.e.*, concisely reprsented) leakage profile as a contrast with the of OPE scheme of [6] that does not have such a leakage profile but rather leaks "whatever a random order-preserving function does." But we stress that whether a scheme has a "well-defined" leakage profile or not, *additional analysis must be done to analyze the leakage.* In particular, one would like to prove it achieves "semantically meaningful" notions of security that talk about what information about the data the adversary is able to deduce from the leakage. We have taken some initial steps in that direction, as explained above. Such analysis is actually possible in principle regardless, e.g. [7] provide an initial such analysis for the OPE scheme of [6], but is more tractable when the leakage profile is well-defined.

Recently, there have been various attacks [21, 22, 3, 17, 26, 23, 10, 29] on specific higher-level outsourced database protocols (*e.g.*, [31]) based on searchable symmetric encryption or property-preserving encryption schemes like ORE, even assuming ideal leakage, using auxiliary information about the data. We do not claim that our ORE scheme is safe to use in any specific higher-level protocol, and our analysis of our leakage profile does not show this.

Our perspective here is that a foundational study of practical ORE schemes is still warranted. Indeed, it may well be the case that that such ORE schemes are useful in the design of efficient and "reasonably secure" outsourced database protocols, but current proposals of such protocols are using them in a wrong (less secure) way. Furthermore, ORE may turn out to be useful in other emerging practical applications in databases and networking, and has recently found applications to "private learning" [**?**]. Overall, we believe it is a natural and compelling primitive that serves as an important case study in the area of leaky cryptography.

1.3 Related Work

Work done on "leaky cryptography" includes work on multiparty computation [28], searchable symmetric and structured encryption [32, 18, 12, 16, 13, 11, 25], and property-preserving encryption [5, 6, 30]. In the database community, the problem of querying an encrypted database was introduced by Hacigümüş, Iyer, Li and Mehrotra [20], leading to a variety of proposals there but mostly lacking formal security analysis. Proposals of specific outsourced database systems based on property-preserving encryption like ORE include CryptDB [31], Cipherbase [2], and TrustedDB [4].

2. ORE DEFINITIONS

NOTATION. All algorithms are assume to be polynomial-time in the security parameter (though we will sometimes refer to efficient algorithms explicitly). We will denote the security parameter by λ. If $\mathcal{A}$ is a randomized algorithm, we write $y \xleftarrow{\$} \mathcal{A}(x)$ to denote running $\mathcal{A}$ on input x with a fresh random tape and letting y be the random variable induced by its output.

We let $[M] = \{1, \ldots, M\}$. If P is a predicate, we write $\mathbf{1}(P)$ for the function that takes the inputs to P and returns 1 if P holds and 0 otherwise.

PRFs. We use the standard notion of a PRF. A function $F : \{0,1\}^\lambda \times D \to \{0,1\}^\lambda$ is said to be a *PRF with domain* D if for all efficient $\mathcal{A}$ we have that

$$|\Pr[\mathcal{A}^{F(K,\cdot)}(1^\lambda) = 1] \Pr[\mathcal{A}^{g(\cdot)}(1^\lambda) = 1]|$$

is a negligible function of λ, where K is uniform over $\{0,1\}^\lambda$ and g is uniform over all functions from D to $\{0,1\}^\lambda$.

ORE. The following definition of syntax for an order-revealing encryption makes explicit that comparison may use helper information (e.g. a description of a particular group) by incorporating a *comparison key*, denote ck.

DEFINITION 2.1 (ORE). *A* ORE scheme *is a tuple of algorithms* $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{C})$ *with the following syntax.*

- *The key generation algorithm* $\mathcal{K}$ *is randomized, takes inputs* $(1^\lambda, M)$*, and always emits two outputs* $(\mathsf{sk}, \mathsf{ck})$*. We refer to the first output* sk *as the* secret key *and the second output* ck *as the* comparison key*.*
- *The encryption algorithm* $\mathcal{E}$ *is randomized, takes inputs* (sk, m) *where* $m \in [M]$*, and always emits a single output* c*, that we refer to as a* ciphertext*.*
- *The comparison algorithm* $\mathcal{C}$ *is deterministic, takes inputs* (ck, c_1, c_2)*, and always emits a bit.*

If the comparison algorithm $\mathcal{C}$ *is simple integer comparison (i.e., if* $\mathcal{C}(\mathsf{ck}, c_1, c_2)$ *is a canonical algorithm that treats its the ciphertexts and binary representations of integers and tests which is greater) then the scheme is said to be an* order-preserving encryption (OPE) *scheme.*

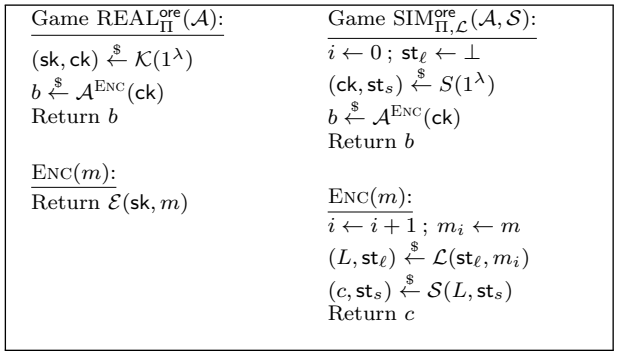

Figure 1: Games $\text{REAL}^{\mathsf{ore}}\Pi(\mathcal{A})$ (left) and $\text{SIM}^{\mathsf{ore}}_{\Pi,\mathcal{L}}(A,S)$ (right), where $\Pi = (\mathcal{E},\mathcal{C})$ is an ORE scheme, $\mathcal{L}$ is a leakage profile, A is an adversary, and S is a simulator.

CORRECTNESS OF ORE SCHEMES. Intuitively, an ORE scheme is correct if the comparison algorithm returns 1 whenever c_2 is generated with a message that is greater than the message used to generate c_1. Thus the comparison algorithm can be used to tell which message is greater.

Our constructions will only be *computationally* correct, i.e. correct with overwhelming probability when the input messages are provided by an efficient process, under hardness assumptions. Formally, we define correctness using the game $\text{COR}^{\mathsf{ore}}_{\Pi}(\mathcal{A})$, which is defined as follows: The game starts by running $(\mathsf{sk},\mathsf{ck}) \xleftarrow{\$} \mathcal{K}(1^\lambda, M)$, and it gives $(\mathsf{sk},\mathsf{ck})$ to $\mathcal{A}$. The adversary $\mathcal{A}$ then outputs two messages $x, y \in [M]$. The game computes $c_1 \xleftarrow{\$} \mathcal{E}(\mathsf{sk}, x)$ and $c_2 \xleftarrow{\$} \mathcal{E}(\mathsf{sk}, y)$, outputs 1 if $x < y$ but $\mathcal{C}(\mathsf{ck}, c_1, c_2) = 0$.

We say that an ORE scheme Π is *computationally correct* if for all efficient adversaries $\mathcal{A}$, all $M = \text{poly}(\lambda)$, we have that $\Pr[\text{COR}^{\mathsf{ore}}_{Pi}(\mathcal{A}) = 1]$ is a negligible function in the security parameter.

SECURITY OF ORE SCHEMES. The following simulation-based security definition is due to Chenette et al. [14]. Here a *leakage profile* is any randomized algorithm. The definition refers to games given in Figure 1, which we review now. In the real game, key generation is run and the adversary is given the comparison key and oracle access to the encryption algorithm with the corresponding secret key. The adversary eventually outputs a bit that the game uses as its own output. In the ideal simulation game, the adversary is interacting with the same oracle, but the comparison key is generated by a stateful simulator, and the oracle responses are generated by the simulator which receives leakage from the stateful leakage algorithm $\mathcal{L}$.

DEFINITION 2.2 ($\mathcal{L}$-SIMULATION-SECURITY FOR ORE). *For an ORE scheme Π, an adversary $\mathcal{A}$, a simulator $\mathcal{S}$, and leakage profile $\mathcal{L}$, we define the games* $\text{REAL}^{\mathsf{ore}}_{\Pi}(\mathcal{A})$ *and* $\text{SIM}^{\mathsf{ore}}_{\Pi,\mathcal{L}}(\mathcal{A})$ *in Figure 1. The advantage of $\mathcal{A}$ with respect to $\mathcal{S}$ is defined as* $\mathsf{Adv}^{\mathsf{ore}}_{\Pi,\mathcal{L},\mathcal{A},\mathcal{S}}(\lambda) =$

$$\left|\Pr[\text{REAL}^{\mathsf{ore}}_{\Pi}(\mathcal{A}) = 1] - \Pr[\text{SIM}^{\mathsf{ore}}_{\Pi,\mathcal{L}}(\mathcal{A},\mathcal{S}) = 1]\right|.$$

We say that Π is $\mathcal{L}$-simulation-secure if for every efficient adversary $\mathcal{A}$ there exists an efficient simulator $\mathcal{S}$ such that $\mathsf{Adv}^{\mathsf{ore}}_{\Pi,\mathcal{L},\mathcal{A},\mathcal{S}}(\lambda)$ *is a negligible function.*

We also define non-adaptive *variants of the games where $\mathcal{A}$ gets a single query to an oracle that accepts a vector of messages of unbounded size. In the real game* $\text{REAL}^{\mathsf{ore-na}}_{\Pi}(\mathcal{A})$*, the oracle returns the encryptions applied to sequentially to each message. In the ideal game* $\text{SIM}^{\mathsf{ore-na}}_{\Pi}(\mathcal{A})$*, the leakage function gets the entire vector of messages as input and produces and output L that is then given to $\mathcal{S}$ which produces a vector of ciphertexts, which are returned by the oracle.*

We define the non-adaptive advantage *of $\mathcal{A}$ with respect to $\mathcal{S}$ analogously, and denote it* $\mathsf{Adv}^{\mathsf{ore-na}}_{\Pi,\mathcal{L},\mathcal{A},\mathcal{S}}(\lambda)$. Non-adaptive $\mathcal{L}$-simulation security *is defined analogously.*

3. ORE CONSTRUCTION

We start by defining the security we target via a leakage function. Then we recall a primitive for our construction called a *property-preserving hash (PPH) function*, and state and analyze our ORE construction using a PPH. In a later section we instantiate the PPH to complete the construction.

3.1 Leakage Function

We first define some notation. For $m_1, m_2 \in \{0,1\}^n$, the *most significant different bit of m_1 and m_2*, denoted $\mathsf{msdb}(m_1, m_2)$, is defined to be the index of first bit where m_1 and m_2 differ, or $n+1$ if $m_1 = m_2$. More precisely, $\mathsf{msdb}(m_1, m_2) = \min\{i : m_1[i] \neq m_2[i]\} \cup \{n+1\}$.

Then We define the non-adaptive version of the our leakage definition of an ORE scheme. The leakage profile takes in input a vector of messages $\vec{m} = (m_1, \ldots, m_n)$ and produces the following:

$$\mathcal{L}_f(m_1, \ldots, m_t) := \left\{ \begin{array}{l} \mathbf{1}(m_i < m_j), \\ \mathbf{1}(\mathsf{msdb}(m_i, m_j) = \mathsf{msdb}(m_i, m_k)) \\ \text{for } 1 \leq i, j, k \leq q \end{array} \right\}.$$

The profile can be represented by two matrices $\mathbf{D} \in \{0,1\}^{n \times n}$, $\mathbf{M} \in \{0,1\}^{n \times n \times n}$, where $\mathbf{D}_{ij} = \mathbf{1}(m_i < m_j)$ for $i, j \in [n]$, and $\mathbf{M}_{ijk} = \mathbf{1}(\mathsf{msdb}(m_i, m_j) = \mathsf{msdb}(m_i, m_k))$, for $i, j, k \in [q]$.

EXAMPLE. Let $n = 3$, and consider

$$m_1 = 000,\ m_2 = 100,\ m_3 = 101$$

and

$$m'_1 = 000,\ m'_2 = 100,\ m'_3 = 110.$$

The leakage on (m_1, m_2, m_3) will be the same as the leakage on (m'_1, m'_2, m'_3), and thus these tuples will be indistinguishable when our construction is used. Under the leakage profile of [15], however, these tuples are distinguishable because $\mathsf{msdb}(m_2, m_3) \neq \mathsf{msdb}(m'_2, m'_3)$.

3.2 Property Preserving Hash

Our construction will depend on a tool we term a *property preserving hash (PPH)*, which is essentially a property-preserving encryption scheme [30] without a decryption algorithm. In this section we recall the syntax and security of a PPH.

DEFINITION 3.1. *A* property-preserving hash (PPH) scheme *is a tuple of algorithms $\Gamma = (\mathcal{K}_h, \mathcal{H}, \mathcal{T})$ with the following syntax:*

- *The key generation algorithm $\mathcal{K}_h$ is randomized, takes as input 1^λ and emits two outputs $(\mathsf{hk}, \mathsf{tk})$ that we refer to as the* hash key hk *and* test key tk. *These implicitly define a domain D and range R for the hash.*

Game $\text{IND}^{\mathsf{pph}}_{\Gamma,P}(\mathcal{A})$:

$(\mathsf{hk}, \mathsf{tk}) \xleftarrow{\$} \mathcal{K}_h(1^\lambda)$
$x^* \xleftarrow{\$} D$; $h_0 \xleftarrow{\$} \mathcal{H}(\mathsf{hk}, x^*)$
$h_1 \xleftarrow{\$} R$; $b \xleftarrow{\$} \{0,1\}$
$b' \xleftarrow{\$} \mathcal{A}^{\text{HASH}}(\mathsf{tk}, x^*, h_b)$
Return $(b \stackrel{?}{=} b')$

HASH(x):
If $P(x^*, x) = 1$ or $P(x, x^*) = 1$
Then $h \leftarrow \perp$
Else $h \xleftarrow{\$} \mathcal{H}(\mathsf{hk}, x)$
Return h

Figure 2: Game $\text{IND}^{\mathsf{pph}}_{\Gamma,P}(\mathcal{A})$.

- *The evaluation algorithm $\mathcal{H}$ is randomized, takes as input the hash key hk, an input $x \in D$, and emits a single output $h \in R$ that we refer to as the hash of x.*

- *The test algorithm $\mathcal{T}$ is deterministic, takes as input the test key tk, and two hashes h_1, h_2, and emits a bit.*

CORRECTNESS OF PPH SCHEMES. Let P be a predicate on pairs of inputs. We define correctness of a PPH Γ via game $\text{COR}^{\mathsf{pph}}_{\Gamma,P}(\mathcal{A})$, which is as follows: It starts by running $(\mathsf{hk}, \mathsf{tk}) \xleftarrow{\$} \mathcal{K}_h(1^\lambda)$ and gives $\mathcal{A}$ tk. Then $\mathcal{A}$ outputs x, y. The game computes $h \xleftarrow{\$} \mathcal{H}(\mathsf{hk}, x), h' \xleftarrow{\$} \mathcal{H}(\mathsf{hk}, y)$ and outputs 1 if $\mathcal{T}(\mathsf{tk}, h, h') \neq P(x, y)$. We say that Γ is *computationally correct* if for all efficient $\mathcal{A}$,

$$\Pr[\text{COR}^{\mathsf{pph}}_{\Gamma,P}(\mathcal{A}) = 1]$$

is a negligible function of λ.

SECURITY OF PPH SCHEMES. We recall a simplified version of the security definition for PPH that is a weaker version of PPE security defined by Pandey and Rouselakis [30].

DEFINITION 3.2. *Let P be some predicate and $\Gamma = (\mathcal{K}_h, \mathcal{H}, \mathcal{T})$ be a PPH scheme with respect to P. For an adversary $\mathcal{A}$ we define the game $\text{IND}^{\mathsf{pph}}_{\Gamma,P}(\mathcal{A})$ in Figure 2. The* restricted-chosen-input advantage of $\mathcal{A}$ *is defined to be*

$$\mathsf{Adv}^{\mathsf{pph}}_{\Gamma,P,\mathcal{A}}(\lambda) = 2\Pr[\text{IND}^{\mathsf{pph}}_{\Gamma,P}(\mathcal{A}) = 1] - 1.$$

We say that Γ is restricted-chosen-input secure *if for all efficient adversaries $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{pph}}_{\Gamma,P,\mathcal{A}}(\lambda)$ is negligible.*

3.3 ORE from PPH

CONSTRUCTION. Let $F : K \times ([n] \times \{0,1\}^n) \to \{0,1\}^\lambda$ be a secure PRF. Let $P(x, y) = \mathbf{1}(x = y + 1)$ be the relation predicate that outputs 1 if and only if $x = y + 1$, and let $\Gamma = (\mathcal{K}_h, \mathcal{H}, \mathcal{T})$ be a PPH scheme with respect to P. In our construction, we interpret the output of F as a λ-bit integer, which is also the input domain of the PPH Γ. We define our ORE scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{C})$ as follows:

- $\mathcal{K}(1^\lambda)$: On input the security parameter, the algorithm chooses a key k uniformly at random for F, and runs the Setup algorithm of the relational hash function $\Gamma.\mathcal{K}_h$ to obtain the hash and test keys $(\mathsf{hk}, \mathsf{tk})$. It sets $\mathsf{ck} \leftarrow \mathsf{tk}$, $\mathsf{sk} \leftarrow (k, \mathsf{hk})$ and outputs $(\mathsf{ck}, \mathsf{sk})$.

- $\mathcal{E}(\mathsf{sk}, m)$: On input the secret key SK and a message m, the algorithm computes the binary representation of $m = (b_1, \ldots, b_n)$, and then calculates:

  $$u_i = F(k, (i, b_1 b_2 \cdots b_i || 0^{n-i})) + b_i, t_i = \Gamma.\mathcal{H}(\mathsf{hk}, u_i).$$

  Then the algorithm chooses a random permutation π, applies it on $(t_1, \ldots, t_n)$, and gets $(v_1, \ldots, v_n)$, where $v_i = t_{\pi(i)}$. The algorithm outputs $\mathsf{CT} = (v_1, \ldots, v_n)$.

- $\mathcal{C}(\mathsf{ck}, CT_1, CT_2)$: on input the public parameter, two ciphertexts $\mathsf{CT}_1, \mathsf{CT}_2$ where

  $$\mathsf{CT}_1 = (v_1, \ldots, v_n); \mathsf{CT}_2 = (v'_1, \ldots, v'_n),$$

  the algorithm runs $\Gamma.\mathcal{T}(\mathsf{tk}, v_i, v'_j)$ and $\Gamma.\mathcal{T}(\mathsf{tk}, v'_i, v_j)$ for every $i, j \in [n]$. If there exists a pair (i^*, j^*) such that $\Gamma.\mathcal{T}(\mathsf{tk}, v_{i^*}, v'_{j^*}) = 1$, then the algorithm outputs 1, meaning $m_1 > m_2$; else if there exists a pair (i^*, j^*) such that $\Gamma.\mathcal{T}(\mathsf{tk}, v'_{i^*}, v_{j^*}) = 1$, then the algorithm outputs 0, meaning $m_1 < m_2$; otherwise it outputs it outputs $\perp$, meaning $m_1 = m_2$.

CORRECTNESS OF ORE. For two messages m_1, m_2, let $(b_1, \ldots b_n)$ and $(b'_1, \ldots, b'_n)$ be their binary representations. We know that if $m_1 > m_2$, then there must exists a unique index $i^* \in [n]$ such that the prefixes of their binary representations up to i^*, say $u = (b_1, \ldots, b_{i^*})$, $u' = (b'_1, \ldots, b'_{i^*})$, satisfy the following relation: $u = u' + 1$. By the correctness of the PPH, we know that $\Gamma.\mathcal{T}(\Gamma.\mathcal{H}(\mathsf{hk}, u), \Gamma.\mathcal{H}(\mathsf{hk}, u')) = 1$, with overwhelming probability. We can use the same argument for the case $m_1 < m_2$.

For the case $m_1 = m_2$, we know that all prefixes of the two messages are identical. For this case, the test algorithm of Γ outputs $\perp$ (for all possible pairs) with overwhelming probability. This proves the correctness of our ORE scheme.

SECURITY. Next analyze the security of our construction.

THEOREM 3.3. *The ORE scheme Π is $\mathcal{L}_\mathsf{f}$-non-adaptively-simulation secure, assuming F is a secure PRF and Γ is restricted-chosen-input secure.*

PROOF. Fix a security parameter λ, and let $\mathcal{A}$ be the adversary for the ORE game, to prove the security, we give an efficient simulator $\mathcal{S}$ for which the outputs of the distributions $\text{REAL}^{\mathsf{ore}}_\Pi(\mathcal{A})$ and $\text{SIM}^{\mathsf{ore}}_{\Pi,\mathcal{L}_\mathsf{f}}(\mathcal{A}, \mathcal{S})$ are computationally indistinguishable.

We use a sequence of hybrid argument to prove the indistinguishability, and present that for each hybrid we have $H_i \overset{\mathsf{comp}}{\approx} H_{i+1}$. Here are the definitions of hybrid experiments:

- **Hybrid** H_0: This is the real experiment $\text{REAL}^{\mathsf{ore}}_\Pi(\mathcal{A})$;

- **Hybrid** H^*_0: Same as H_0, except during Setup, a random function $F_1 \xleftarrow{\$} \mathsf{Funs}([n] \times \{0,1\}^n, \{0,1\}^\lambda)$ is chosen. In all invocations of $\mathcal{E}$, the function $F_k(\cdot)$ is replaced by F_1;

- **Hybrid** $\mathsf{H}_{\mathsf{i}\times\mathsf{q}+\mathsf{j}}$. We define these hybrids inductively. H_1 is exactly the same as H^*_0. Then for any $k \in [2, qn]$, let $k = i \times q + j$. If **Event** $\mathsf{Switch}_{(\mathsf{i},\mathsf{j})} = \mathsf{0}$, then H_k is set exactly the same as $\mathsf{H}_{\mathsf{k}-1}$. Otherwise, the hash of the prefix u_i of message m_j (i.e. t_i) is replaced with a random string $r_i \in \{0,1\}^\lambda$. We define these events below.

It's easy to see that $\mathsf{H}_0 \overset{\mathsf{comp}}{\approx} \mathsf{H}_0^*$ by the security of PRF F and $\mathsf{H}_0^* = \mathsf{H}_1$. We will show that for any $i \in [1, qn-1]$, $\mathsf{H}_i \overset{\mathsf{comp}}{\approx} \mathsf{H}_{i+1}$. Then it suffices to state that there exists a simulator $\mathcal{S}$, satisfying the distribution of outputs in H_{qn} is computationally indistinguishable from $\mathrm{SIM}^{\mathsf{ore}}_{\Pi,\mathcal{L}_\mathsf{f}}(\mathcal{A},\mathcal{S})$. Next we proceed to define the events we mentioned above.

DEFINITION 3.4. *For fixed messages $m_1, \ldots, m_q$, we define binary events* ***Event*** $\mathsf{Switch}_{(\mathsf{i},\mathsf{j})}$ *for $i \in [n], j \in [q]$ as follows:* ***Event*** $\mathsf{Switch}_{(\mathsf{i},1)}$ *is set to 0 for every $i \in [n]$. For $j > 1$,* ***Event*** $\mathsf{Switch}_{(\mathsf{i},\mathsf{j})} = 1$ *if and only if for every $k \in [n]$, $k \neq j$, we have $\mathsf{ind}_{\mathsf{diff}}(m_j, m_k) \neq i$. Otherwise, the event is set to 0.*

Intuitively, **Event** $\mathsf{Switch}_{(\mathsf{i},\mathsf{j})} = 1$ means that the i-th bit of message m_j is not leaked by the leakage profile. Next we prove the adjacent hybrids are computationally indistinguishable.

LEMMA 3.5. *For any $k \in [0, qn-1]$, where $k+1 = i \times q + j (i \in \{0, \ldots n-1\}, j \in [q])$, $H_k \overset{\mathsf{comp}}{\approx} H_{k+1}$, if the Γ is a secure PPH with respect to the predicate P.*

PROOF. If Event $\mathsf{Swith}_{\mathsf{i},\mathsf{j}} = 0$ happens, then H_{k+1} is exactly the same as H_k. Therefore, it suffices to show that the two hybrids are computational indistinguishable in the case $\mathsf{Swith}_{\mathsf{i},\mathsf{j}} = 1$. Suppose $\mathcal{A}$ can distinguish H_k from H_{k+1} with some non-negligible advantage ϵ, then we are going to build a simulator $\mathcal{B}$ that breaks the security of the PPH scheme Γ with advantage ϵ. This suffices to prove the lemma. We define $\mathcal{B}$ as follows.

Firstly $\mathcal{B}$ picks a random function F_1 (using the lazy sampling technique), and plays the ORE game with $\mathcal{A}$. In the initial phase, $\mathcal{A}$ sends a sequence of messages $(m_1, m_2, \ldots, m_q)$ to $\mathcal{B}$, and $\mathcal{B}$ calculates $X^* = F_1(i, b_1 b_2 \cdots b_i || 0^{n-i}) + b_i$, and uses X^* as the challenge value in the PPH security game. Here we use a natural order for tuples (i, j) where $(i, j) > (i', j')$ if $iq + j > i'q + j'$. Then $\mathcal{B}$ does the following:

- Init $\mathcal{B}$ submits X^* to the PPH challenger $\mathcal{C}$, where X^* is the input value $\mathcal{B}$ intends to attack.
- For any $(i', j') \neq (i, j)$, if $\mathsf{Switch}_{\mathsf{i}',\mathsf{j}'} = 0$, $\mathcal{B}$ calculates $u_{i',j'} = F_1(i', b_{j',1} b_{j',2}, \cdots b_{j',i'} || 0^{n-i'}) + b_{j',i'}$, where $b_{j',1}, \ldots b_{j',n}$ is the binary representation of $m_{j'}$, and query the hash oracle $\Gamma.\mathcal{H}(\mathsf{hk}, u_{i'j'})$ and sets $t_{i',j'} = \Gamma.\mathcal{H}(u_{i'j'})$. Since F_1 is a random function, the probability that $u_{i',j'} = X^* \pm 1$ is negligible, $\mathcal{B}$ is permitted to query the hash value in overwhelming probability. For $(i', j') \neq (i, j)$ and $\mathsf{Switch}_{\mathsf{i}',\mathsf{j}'} = 1$, $\mathcal{B}$ picks a random element R in the range of the hash function and sets $t_{i',j'} = R$.
- The relational hash challenger $\mathcal{C}$ gives $\mathcal{B}$ a challenge term T which is either $\Gamma.\mathcal{H}(\mathsf{hk}, X^*)$ or a random element from the range.
- Guess After receiving T, $\mathcal{B}$ sets $t_{i,j} = T$. Then for any $j \in [q]$, choose a random permutation π_j and sets the ciphertext $CT_j = (v_{1,j}, \ldots, v_{n,j})$ where $v_{i,j} = t_{\pi_j(i),j}$. $\mathcal{B}$ sets $(CT_1, CT_2, \ldots, CT_q)$ to $\mathcal{A}$, and outputs whatever $\mathcal{A}$ outputs.

We recall that we analyze the case where $\mathsf{Switch}_{\mathsf{i},\mathsf{i}} = 1$. Therefore, when $T = \Gamma.\mathcal{H}(\mathsf{hk}, X^*)$, then $\mathcal{B}$ properly simulates $H_k (k+1 = iq + j)$, and when T is a random element in the range of the hash function, then $\mathcal{B}$ properly simulates H_{k+1}. Hence, if $\mathcal{A}$ has advantage ϵ to distinguish H_k and H_{k+1}, $\mathcal{B}$ can break the restricted-chosen-input secure of Γ with advantage ϵ. □

Next, we will present that there exists a simulator $\mathcal{S}$ such that the distribution of outputs in H_{qn} is computationally indistinguishable from $\mathrm{SIM}^{\mathsf{ore}}_{\Pi,\mathcal{L}_\mathsf{f}}(\mathcal{A},\mathcal{S})$

Description of the simulator. For fixed a message set $\mathcal{M} = \{m_1, \ldots, m_q\}$, the simulator $\mathcal{S}$ is given the leakage information $\mathcal{L}_f(m_1, \ldots, m_q)$. The task is to construct simulated ciphertexts that are indistinguishable from the ones in the hybrid H_{qn}. Without loss of generality, we assume that $m_1 > m_2, \cdots > m_q$, as the order is revealed from the leakage profile.

To simulate the ciphertexts, the simulator first keeps a matrix $\mathcal{B}$ of dimension $q \times n$. Then it runs $\mathsf{FillMatrix}(1, 1, q)$ to fill in the entries, where $\mathsf{FillMatrix}(i, j, k)$ can be described as the following recursive algorithm:

- If $j = k$, then for every $i' \in [i, n]$ sample a (new) random string r and set $\mathcal{B}[i'][j] = (1||r)$.
- Else, the algorithm proceeds as follows:
  - Find the smallest $j' \in [j, k]$ such that $P(m_j, m_{j'}) = P(m_j, m_k)$.
  - Sample a new random string r', and set $\mathcal{B}[i][j'] = (0||r')$ for $j' \in [j, j'-1]$ and set $\mathcal{B}[i][''j] = (0||r' - 1)$ for $j'' \in [j', j]$.
  - Recursively call $\mathsf{FillMatrix}(i+1, j, j'-1)$ and $\mathsf{FillMatrix}(i+1, j', k)$.

We observe that these recursive subroutines will never write on the same entry twice. Since there are only qn entries of the matrix, the recursive algorithm will terminates in at most qn steps.

Finally, $\mathcal{S}$ runs $\Gamma.\mathcal{K}_h(1^\lambda)$ and gets the keys tk, hk, then $\mathcal{S}$, for every $i \in [n], j \in [q]$, parses $\mathcal{B}[i][j]) = (b_{i,j}||r_{i,j})$ (a bit along with a string). If $b_{i,j} = 0$, set $t_{i,j} = r_{i,j}$, and otherwise $t_{i,j} = \Gamma.\mathcal{H}(\mathsf{hk}, r_{i,j})$. Finally $\mathcal{S}$ permutes all the $t_{i,j}$'s randomly as the encryption algorithm, and outputs the permuted ciphertexts $\mathsf{CT}_1, \ldots, \mathsf{CT}_q$.

We note that in the hybrid H_{qn}, $u_{i,j}$ and $u_{i,j'}$ have relation $(u_{i,j} = u_{i,j'} + 1 \cup u_{i,j} = u_{i,j'} \cup u_{i,j} = u_{i,j'} - 1)$ if and only if the i-th bit of m_j and $m_{j'}$ are both leaked bits. Otherwise hash values of $u_{i,j}$'s are replaced completely by random numbers. And for the simulator, $\mathcal{S}$ only preserves the relationship between $u_{i,j}$ and $u_{i,j'}$ when the i-th bit are both leaked bits, and for other $u_{i,j}$'s hash value, $\mathcal{S}$ also just sets random bits. Regarding the randomness of permutation, the distribution of ciphers in H_q and $\mathcal{S}$ are statistically indistinguishable. This completes the proof of the theorem.

4. PPH FROM BILINEAR MAPS

In this section, we present a PPH scheme for the predicate P, required in our ORE construction. That is, $P(x, y) = 1$ if and only if $x = y + 1$. Our construction uses an asymmetric bilinear map and a PRF F. Below, we let F be a PRF and $e : \mathbb{G} \times \hat{\mathbb{G}} \to \mathbb{G}_T$ be a bilinear pairing over groups $(\mathbb{G}, \hat{\mathbb{G}})$ of prime order p, with generators $g \in \mathbb{G}$ and $\hat{g} \in \hat{\mathbb{G}}$ respectively. Let $F : \{0,1\}^\lambda \times (\{0,1\}^\lambda) \to \{0,1\}^\lambda$ and in

the following we will sometimes view the output of F as the binary representation of a λ-bit integer.

CONSTRUCTION. We now define our PPH $\Gamma = (\mathcal{K}_h, \mathcal{H}, \mathcal{T})$.

- $\mathcal{K}_h(1^\lambda)$ This algorithm takes the security parameter as input, generates a bilinear groups map $e : \mathbb{G} \times \hat{\mathbb{G}} \to \mathbb{G}_T$ with the generators $g, \hat{g}$, and chooses $k \xleftarrow{\$} \{0,1\}^\lambda$. Then it sets the hash key $\mathsf{hk} \leftarrow (k, g, \hat{g})$, the test key $\mathsf{tk} \leftarrow (\mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T, e)$, a description of the bilinear map and groups, and outputs $(\mathsf{hk}, \mathsf{tk})$.

- $\mathcal{H}(\mathsf{hk}, x)$ This algorithm takes as input the hash key hk, an input x, picks two random non-zero $r_1, r_2 \in \mathbb{Z}_p$ and outputs

$$\mathcal{H}(\mathsf{hk}, x) = (g^{r_1}, g^{r_1 \cdot F(k,x)}, \hat{g}^{r_2}, \hat{g}^{r_2 \cdot F(k,x+1)}).$$

- $\mathcal{T}(\mathsf{tk}, h_1, h_2)$ To test two hash values (A_1, A_2, B_1, B_2) and (C_1, C_2, D_1, D_2), $\mathcal{T}$ outputs 1 if

$$e(A_1, D_2) = e(A_2, D_1),$$

and otherwise it outputs 0.

Hence the domain D is $\{0,1\}^\lambda$ and the range R is $(\mathbb{G}^2, \hat{\mathbb{G}}^2)$

CORRECTNESS. The condition tested is equivalent to $F(k, y+1) = F(k, x)$ If $x = y + 1$ then this is obviously true. If not, then it is easily shown that finding x, y with this property with non-negligible probability leads to an adversary that contradicts the assumption that F is a PRF.

Security Proof. We prove that PPH is restricted-chosen-input secure, assuming that F is a PRF and that a pairing-based assumption holds. We recall the assumption first.

DEFINITION 4.1. *Let $\mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T$ be prime-order p groups, g be generator of $\mathbb{G}$ and $\hat{g}$ be a generator of $\hat{\mathbb{G}}$, tand $e : \mathbb{G} \times \hat{\mathbb{G}} \to \mathbb{G}_T$ be a bilinear pairing. We say the* symmetric external Diffie-Hellman assumption holds *with respect to these groups and pairing if for all efficient $\mathcal{A}$,*

$$|\Pr[\mathcal{A}(g, g^a, g^b, g^{ab}) = 1] \Pr[\mathcal{A}(g, g^a, g^b, T) = 1]|$$

and

$$|\Pr[\mathcal{A}(\hat{g}, \hat{g}^a, \hat{g}^b, \hat{g}^{ab}) = 1] \Pr[\mathcal{A}(\hat{g}, \hat{g}^a, \hat{g}^b, T) = 1]|$$

are negligible functions of λ, where a, b, c are uniform over $\mathbb{Z}_p$ and T is uniform over G_T.

We can now state and prove our security theorem.

THEOREM 4.2. *Our PPH Γ is restricted-chosen-input secure, assuming F is a PRF and the SXDH assumption hold with respect to the appropriate groups and pairing.*

PROOF. The proof proceeds by a hybrid argument across a number of games. Let $(A_1, A_2, B_1, B_2 \in \mathbb{G}^2 \times \hat{\mathbb{G}}^2$ denotes the challenge hash value given to the adversary during the real game $\mathrm{IND}^{\mathsf{pph}}_{\Gamma,P}(\mathcal{A})$, which we rename **Game 0** here . Additionally, let R be a random element of $\mathbb{G}$, $\hat{R}$ be a random element of $\hat{\mathbb{G}}$, both independent of the rest of the random variables under consideration. Then we define the following hybrid experiments:

- **Game 0** The challenge hash value is (A_1, A_2, B_1, B_2), which is the real game in sCHA security;
- **Game 1** During the setup procedure, a uniformly random function $F^* \xleftarrow{R} \mathsf{Funs}[\{0,1\}^\lambda, \{01,\}^\lambda]$ is sampled instead of the PRF key K, the rest remain unchanged.
- **Game 2** The challenge hash value is (A_1, R, B_1, B_2);
- **Game 3** The challenge hash value is $(A_1, R, B_1, \hat{R})$

We note that **Game 0** is exactly the real game, and in **Game 3**, the adversary is given a random element from the range $\mathcal{R}$. Therefore,

$$\mathsf{Adv}^{\mathsf{pph}}_{\Gamma,P,\mathcal{A}}(\lambda) = |Pr[\mathcal{A}^{\mathbf{Game\ 0}} = 1] - Pr[\mathcal{A}^{\mathbf{Game\ 3}} = 1]|$$

To prove **Game 0** is indistinguishable from **Game 3**, we show that each step of the hybrid is indistinguishable from the next. Firstly, it's apparent that Game 0 and Game 1 are computational indistinguishable by the PRF security, then:

LEMMA 4.3. ***Game 1*** $\approx$ ***Game 2*** *under the SXDH assumption.*

Let $\mathcal{A}$ be an adversary playing the PPH security game, and let $\mathcal{A}$

$$\epsilon = |Pr[\mathcal{A}^{\mathbf{Game\ 1}} = 1] - Pr[\mathcal{A}^{\mathbf{Game\ 2}} = 1]|.$$

Then we can build a simulator $\mathcal{B}$ to SXDH assumption with advantage ϵ. $\mathcal{B}$ is given as input $(g, \hat{g}, g^b, g^c)$ and the challenge term T either g^{bc} or R, then $\mathcal{B}$ works by interacting with $\mathcal{A}$ in the PPH security game as follows:

- Initial $\mathcal{A}$ randomly chooses $x^* \in \{0,1\}^\lambda$ and sends it to $\mathcal{B}$.
- Setup $\mathcal{B}$ runs the Setup algorithm, sends the test key e to $\mathcal{A}$ and sets $g, \hat{g}$ as the secret key.
- Phase1 In this phase, $\mathcal{B}$ will answers the hash query of input $x \pm 1 \neq x^*$, firstly $\mathcal{B}$ forms a table, empty as initial, to record the value of $F^*(x)$, where $\mathcal{B}$ computes $F^*(x)(x \neq x^*)$ as follows:
  - If there is a entry (x, k) in the table, returns k;
  - Otherwise, chooses a random $k \xleftarrow{\$} \{0,1\}^\lambda$, return it, and store (x, k) in the table

  and implicitly sets $F^*(x^*) = b$. When $\mathcal{A}$ calls a hash query for $x \neq x^*$, $\mathcal{B}$ calculates $F^*(x)$ and $F^*(x+1)$ (it's doable due to $x, x+1 \neq x^*$), picks r_1, r_2 randomly and computes:

$$\mathcal{H}(x) = g^{r_1}, g^{r_1 \cdot F^*(x)}, \hat{g}^{r_2}, \hat{g}^{r_2 \cdot F^*(x+1)};$$

  If $\mathcal{A}$ runs a call for x^*, $\mathcal{B}$ calculate $F^*(x^* + 1)$, picks r_1', r_2' randomly and computes:

$$\mathcal{H}(x^*) = g^{r_1'}, g^{r_1' \cdot b}, \hat{g}^{r_2'}, \hat{g}^{r_2' \cdot F^*(x^*+1)};$$

  And it's obvious that those hashes are properly distributed.
- Challenge In the phase, $\mathcal{B}$ calculate $F^*(x^* + 1)$, picks r^* randomly, computes

$$A_1 = g^c, A_2 = T, B_1 = \hat{g}^{r^*}, B_2 = \hat{g}^{r^* F^*(x^*+1)}$$

  and sends it to $\mathcal{A}$.

- Phase2 Same as Phase1.
- Guess $\mathcal{B}$ outputs whatever $\mathcal{A}$ outputs.

We note that in $\mathcal{A}$'s view, without querying $\mathcal{A}(x^*-1)$, $\mathcal{B}$ simulates the RCI game properly. If $T = g^{bc}$, then $\mathcal{B}$ simulates **Game 1**, and if $T = R$ then **Game 2**. Hence if $\mathcal{A}$ has an advantage ϵ to distinguish **Game 1** and **Game 2**, $\mathcal{B}$ has the same advantage to break SXDH assumption.

By symmetry, we also have the following lemma:

LEMMA 4.4. ***Game 2*** $\approx$ ***Game 3*** *under the SXDH assumption.*

The proof is symmetrically same as the one in lemma 4.3, except $\mathcal{B}$ sets A_1, A_2 as two random elements in $\mathbb{G}$ in Challenge, so we skip it here due to the space limit.

Combining them together, we have **Game 0** $\overset{\text{comp}}{\approx}$ **Game 3**, which completes the proof of Theorem 4.2. □

5. IMPROVING LEAKAGE AND EFFICIENCY

5.1 Improved Leakage Profile

In this section, we show how to construct an ORE scheme with a better leakage profile and better efficiency. Intuitively, we encrypt the plaintext d-bit by d-bit, rather than bit by bit as our basic ORE. For the set of n-bit positive integers with the following leakage function:

$$\mathsf{L}_R(m_1, \ldots, m_q) := \{1(m_i < m_j), 1(P(m_i, m_j), P(m_i, m_k))\}$$

where $P(m_i, m_j)$ is the most significant differ-block of m_i, m_j. For example, $n = 10, d = 2$, we set $m_1 = (0010010101)_2, m_1 = (0010111100)_2$, then $P(m_1, m_2) = 3$ since every block contains 2 bits, and m_1, m_2 differs in the third block.

5.2 Generalized ORE

Comparing to the basic ORE, our generalized ORE encrypts the plaintext d-bit by d-bit. Here is the description of the scheme. Fix a security parameter $\lambda \in \mathbb{N}$, let $F : \mathcal{K} \times ([n] \times \{0,1\}^n) \to \{0,1\}^\lambda$ be a secure PRF. Let $P_d(x_1, x_2) = x_1 \in \{x_2+1, \ldots, x_2+2^d-1\}$ let $\Gamma = (\mathcal{K}_h, \mathcal{H}, \mathcal{T})$ be a generalized PPH scheme with respect to predicate P_d. In our construction, we interpret the output of F as a λ-bit integer, which is also the input domain of Γ. We define our ORE scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{C})$ as follows:

- $\mathcal{K}(1^\lambda)$: on input the security parameter λ, the algorithm picks a uniform key $\mathsf{k} \in \mathcal{K}$ for the PRF F and runs the Setup algorithm of the generalized PPH $\Gamma.\mathcal{K}_h$ to obtain the hash and test keys $(\mathsf{hk}, \mathsf{tk})$. It sets the comparison key $\mathsf{ck} = \mathsf{tk}$ and secret key $\mathsf{sk} = k, \mathsf{hk}$.
- $\mathcal{E}(\mathsf{sk}, m)$: on input a secret key sk and a message $m = (b_1 || \ldots || b_{n/d})$(later we denote $\ell = n/d$ for short), where b_i is d-bit block, this algorithm calculates:

  $$u_i = F(k, (i, b_1 b_2 \cdots b_{(i-1)d} || 0^{n-(i-1)d})) + b_i$$
  $$t_i = \Gamma.\mathcal{H}(\mathsf{hk}, u_i)$$

  (Here we abuse the notation b_i as an integer value according to its binary representation.) Then it chooses a random permutation π, applies it on $(t_1, \ldots, t_\ell)$ and gets $(v_1, \ldots, v_\ell)$, where $v_i = t_{\pi(i)}$. The algorithm outputs $\mathsf{CT} = (v_1, \ldots, v_\ell)$.
- $\mathcal{C}(\mathsf{ck}, CT_1, CT_2)$: on input the public parameter, two ciphertexts $\mathsf{CT}_1, \mathsf{CT}_2$ where

  $$\mathsf{CT}_1 = (v_1, \ldots, v_\ell); \mathsf{CT}_2 = (v'_1, \ldots, v'_\ell),$$

  the algorithm runs the test algorithm $\Gamma.\mathcal{T}(\mathsf{tk}, v_i, v'_j)$ and $\Gamma.\mathcal{T}(\mathsf{tk}, v'_i, v_j)$ for every $i, j \in [\ell]$. If there exists a pair (i^*, j^*) such that $\Gamma.\mathcal{T}(\mathsf{tk}, v_{i^*}, v'_{j^*}) = 1$, then the algorithm outputs 1, meaning $m_1 > m_2$; else if there exists a pair (i^*, j^*) such that $\Gamma.\mathcal{T}(\mathsf{tk}, v'_{i^*}, v_{j^*}) = 1$, then the algorithm outputs 0, meaning $m_1 < m_2$; otherwise it outputs it outputs $\perp$, meaning $m_1 = m_2$.

CORRECTNESS OF THE GENERALIZED ORE. For two messages m_1, m_2, let $(b_1, \ldots b_\ell)$ and $(b'_1, \ldots, b'_\ell)$ be their d-bit block representations. We know that if $m_1 > m_2$, then there must exists a unique index $i^* \in [\ell]$ such that the prefixes of their d-bit block representations up to i^*, say $u = (b_1, \ldots, b_{i^*})$, $u' = (b'_1, \ldots, b'_{i^*})$, satisfy the following relation: $u = u' + i, i = 1, \ldots, 2^d$. By the correctness of the generalized PPH, we know that, with overwhelming probability:

$$\Gamma.\mathcal{T}(\Gamma.\mathcal{H}(\mathsf{hk}, u), \Gamma.\mathcal{H}(\mathsf{hk}, u')) = 1$$

We can use the same argument for the case $m_1 < m_2$.

For the case $m_1 = m_2$, we know that all prefixes of the two messages are identical. For this case, the Test of Γ outputs $\perp$ (for all possible pairs) with overwhelming probability. This proves the correctness of our ORE scheme.

THEOREM 5.1. *The generalized ORE scheme* Π *is* $\mathcal{L}_\mathsf{f}$*-non-adaptively-simulation secure, assuming* F *is a secure PRF and* Γ *is augmented-restricted-chosen-input secure.*

The proof this theorem is very similar to that of Theorem 3.3, using a sequence of hybrid argument, and it is omitted from this version of the paper to save space.

5.3 Generalized PPH

In this section, we present a PPH for a family of predicts P_d, $d \geq 1$ that generalizes the predicate P above as follows. We let $P_d(x, y) = 1$ if $x \in \{y+1, \ldots, y+2^d-1\}$(later we denote $T = 2^d - 1$ for short), and 0 otherwise. For technical reasons we will need to achieve a slightly stronger definition that is given after the construction.

CONSTRUCTION. As before, we use a PRF $F : \{0,1\}^\lambda \times \{0,1\}^\lambda \to \{0,1\}^\lambda$, and we will sometimes view the output of F as the binary representation of a λ-bit integer. We now describe our PPH $\Gamma^d = (\mathcal{K}_h^d, \mathcal{H}^d, \mathcal{T}^d)$ for the generalized predicate P_d, where $d \geq 1$ is a parameter to adjusted.

- $\mathcal{K}_h^d(1^\lambda)$. This algorithm is identical to $\mathcal{K}_h$ given in Γ.
- $\mathcal{H}^d(\mathsf{hk}, x)$ This algorithm takes as input the hash key hk, an input x. For $i = 0, \ldots, T$ it picks random $r_i \leftarrow \mathbb{Z}_p$, then it samples a random permutation π on $[T]$, and then it computes

  $$(A, B) = (g^{r_0}, g^{r_0 \cdot F(K,x)}) \in \mathbb{G} \times \mathbb{G}.$$

  Then, for $i = 1, \ldots, T$ it computes

  $$(X_i, Y_i) = (\hat{g}^{r_i}, \hat{g}^{r_i \cdot F(K, x+\pi(i))}) \in \hat{\mathbb{G}} \times \hat{\mathbb{G}}.$$

  It outputs $(A_0, B_0, X_1, Y_1, \ldots, X_T, Y_T)$.

Game $\text{IND}^{\mathsf{pph-aug}}_{\Gamma,P}(\mathcal{A})$:

$(\mathsf{hk}, \mathsf{tk}) \xleftarrow{\$} \mathcal{K}_h(1^\lambda)$
$x_1^* \xleftarrow{\$} D\ ;\ x_2^* \xleftarrow{\$} \{x : P(x_1^*, x) = 1\}$
$y_1^* \xleftarrow{\$} D\ ;\ y_2^* \xleftarrow{\$} \{x : P(y_1^*, x) = 1\}$
$h_0^1 \xleftarrow{\$} \mathcal{H}(\mathsf{hk}, x_1^*), h_0^2 \xleftarrow{\$} \mathcal{H}(\mathsf{hk}, x_2^*)$
$h_1^1 \xleftarrow{\$} \mathcal{H}(\mathsf{hk}, y_1^*), h_1^2 \xleftarrow{\$} \mathcal{H}(\mathsf{hk}, y_2^*)$
$b' \xleftarrow{\$} \mathcal{A}^{\text{HASH}}(\mathsf{tk}, x_1^*, x_2^*, y_1^*, y_2^*, h_b^1, h_b^2)$
Return $(b \stackrel{?}{=} b')$

HASH(x):
If $\exists z \in \{x_1^*, x_2^*, y_1^*, y_2^*\}$
$P(z, x) = 1$ or $P(x, z) = 1$
Then $h \leftarrow \perp$
Else $h \xleftarrow{\$} \mathcal{H}(\mathsf{hk}, x)$
Return h

Figure 3: Game $\text{IND}^{\mathsf{pph-aug}}_{\Gamma,P}(\mathcal{A})$.

- $\mathcal{T}^d(\mathsf{tk}, h_1, h_2)$. The test algorithm parses each h_j as

$$(A^j, B^j, X_1^j, Y_1^j, \ldots, X_T^j, Y_T^j)$$

Then it tests if there exists an $i \in \{1, \ldots, T\}$ such that

$$e(A^1, Y_i^2) = e(B^1, X_i^2).$$

If it finds such an i it outputs 1, and otherwise it outputs 0.

And the domain D is $\{0,1\}^\lambda$ and the range R is $\mathbb{G}^2 \times \hat{\mathbb{G}}^{2^{d+1}-2}$.

CORRECTNESS. It is easy to show that Γ_d is computationally correct for the predicate P_d, via the same methods as with Γ, assuming that F is a PRF.

SECURITY. Our ORE construction will require a slightly stronger version of PPH security where the adversary is given, in addition to the hash of a random challenge element x^*, the hash of another element y^* such that $P(x^*, y^*) = 1$. For this definition we must assume P allows for easy sampling of such a y^*, which is the case for our predicate P^*. We call this version of PPH security *augmented-restricted-chosen-input* security, and define the advantage of an adversary $\mathcal{A}$ against PPH scheme Γ via

$$\mathsf{Adv}^{\mathsf{pph-aug}}_{\Gamma,P,\mathcal{A}}(\lambda) = 2\Pr[\text{IND}^{\mathsf{pph-aug}}_{\Gamma,P} = 1] - 1.$$

We say that Γ is *augmented-restricted-chosen-input secure* if for all efficient adversaries $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{pph-aug}}_{\Gamma,P,\mathcal{A}}(\lambda)$ is negligible.

THEOREM 5.2. *For each $d \geq 1$, our PPH Γ^d is augmented-restricted-chosen-input secure, assuming F is a PRF and the SXDH assumption hold with respect to the appropriate groups and pairing.*

The proof of this theorem is very similar to that of Theorem 4.2, despite the augmented security definition. It follows via standard game transitions using the SXDH assumptions, and is omitted from this version of the paper to save space.

5.4 Efficiency Analysis

In this part, we will show the efficiency, including the ciphertext size and computational efficiency of order comparison, for the basic ORE ($d = 1$) and the generalized 0ORE ($d = 2$ and $d = 3$). We measure the ciphertext size in terms of group elements, and we use the expected pairing operation count to measure the computational efficiency of order comparison.

Efficiency	Ciphertext Size	Pairing Operation
$d = 1$	$4n$	$n(n-1)$
$d = 2$	$4n$	$\frac{3}{4} \cdot n(n-1)$
$d = 3$	$\frac{4}{3} \times 4n$	$\frac{7}{9} \cdot n(n-1)$

Table 1: Efficiency comparison of the three cases

We note that when $d = 2$, the ciphertext size is the same as for $d = 1$, and when $d = 3$ the ciphertext size is just 1.33 times larger than for $d = 1$. On the other hand, $d = 2$ is $\frac{3}{4}$ times better than $d = 1$ in terms of pairing operations, and $d = 3$ is $\frac{7}{9}$ times better than $d = 1$. Thus, we see that $d = 2$ is strictly better than $d = 1$, and $d = 3$ has some trade-off in ciphertext size and pairing operation with $d = 1$. For larger d, the leakage continues to improve but efficiency compared to $d = 1$ decreases.

6. ANALYZING THE LEAKAGE

6.1 Impossibility Result for Order-Preserving Encryption

Our impossibility result applies to the weaker non-adaptive indistinguishability version of ORE security, so we recall it here (using the weakest definition makes our result more general). For an ORE scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{C})$, an adversary $\mathcal{A}$, leakage function $\mathcal{L}$, and bit b, the game $\text{IND}^{\mathsf{ore}}_{b,\Pi,\mathcal{L}}(\mathcal{A})$ is defined as follows: The game runs $(\mathsf{sk}, \mathsf{ck}) \xleftarrow{\$} \mathcal{K}(1^\lambda)$ and gives ck to $\mathcal{A}$. Eventually $\mathcal{A}$ responds with two vectors of messages $\mathbf{m}_0, \mathbf{m}_1$ of the same length. If $\mathcal{L}(\mathbf{m}_0) \neq \mathcal{L}(\mathbf{m}_1)$ then the game outputs $\perp$. Otherwise the game computes $\mathbf{c}[i] \xleftarrow{\$} \mathcal{E}(\mathsf{sk}, \mathbf{m}_b[i])$ for each i, and gives $\mathbf{c}$ to $\mathcal{A}$, who finally outputs a bit b'. The game also outputs b'.

We define $\mathsf{Adv}^{\mathsf{ore-ind}}_{\Pi,\mathcal{L}}(\mathcal{A}) =$

$$\frac{1}{2}|\Pr[\text{IND}^{\mathsf{ore}}_{0,\Pi,\mathcal{L}}(\mathcal{A}) = 1] - \Pr[\text{IND}^{\mathsf{ore}}_{1,\Pi,\mathcal{L}}(\mathcal{A}) = 1]$$

DEFINITION 6.1. *An ORE scheme Π is (t, ϵ)-non-adaptively-indistinguishable with leakage $\mathcal{L}$ if for all adversaries running in time at most t, we have*

$$\mathsf{Adv}^{\mathsf{ore-ind}}_{\Pi,\mathcal{L}}(\mathcal{A}) \leq \epsilon.$$

THEOREM 6.2. *Let Π be an order-preserving encryption scheme with domain $[M]$, range $[N]$, and suppose that Π is (t_0, ϵ)-non-adaptively-indistinguishable with leakage $\mathcal{L}_\mathsf{f}$ (defined in Section 3.1), where $0 < \epsilon < 1$, and t_0 is the maximum runtime of a family of explicit adversaries given in the proof. Then we have*

$$N \geq (\frac{M-1}{2})^{\frac{1}{8\epsilon}}(M-1).$$

PROOF. Observe that for any two pairs of messages (m_1, m_2) and (m_1', m_2'), we have that $\mathcal{L}_\mathsf{f}(m_1, m_2) = \mathcal{L}_\mathsf{f}(m_1', m_2')$. Below we will limit ourselves to adversaries that always submit two pairs of messages, and thus they will never cause the game to output $\perp$.

Next, let us define some notation for the proof. For each i define a random variable $X_i = \mathcal{E}(\mathsf{sk}, i)$, where sk is a random output from $\mathcal{K}(1^\lambda)$ (all of the X_i are defined with the same random key). We also define the random variables $\Delta := X_M - X_1$, and $\delta_i = X_{i+1} - X_i$ for $i \in [M-1]$.

LEMMA 6.3. *Let Π be an ORE meeting the hypothesis of the theorem, and let Δ be as defined above, and let $S_j = (M-1)(\frac{M-1}{2})^j$. Then for $j \in \{1, \ldots, \frac{1}{8\epsilon}\}$, we have*

$$\Pr[S_{j-1} \leq \Delta < S_j] \leq 4\epsilon.$$

PROOF. We denote $p = \Pr[\Delta < S_{j-1}]$.

We claim that $\Pr[S_{j-1} \leq \Delta < S_j] \leq 4\epsilon$. Supposing not for contradiction, we construct an adversary $\mathcal{A}_j$ to play $\text{IND}^{\mathsf{ore}}_{b,\Pi,\mathcal{L}_\mathsf{f}}$ as follows:

1. Upon receiving ck, $\mathcal{A}_j$ selects $i \leftarrow [M-1]$ and responds with two pairs of messages $\mathbf{m}_0 = (1, M)$ and $\mathbf{m}_1 = (i, i+1)$.

2. Upon receiving $\mathbf{c} = (c_1, c_2)$, $\mathcal{A}$ outputs 1 if

$$c_2 - c_1 < S_{j-1}.$$

Otherwise $\mathcal{A}_j$ outputs 0.

We consider $\Pr[\text{IND}^{\mathsf{ore}}_{b,\Pi,\mathcal{L}_\mathsf{f}}(\mathcal{A}_j) = 1]$ for $b = 0, 1$. First, it is apparent that

$$\Pr[\text{IND}^{\mathsf{ore}}_{0,\Pi,\mathcal{L}_\mathsf{f}}(\mathcal{A}_j) = 1] = \Pr[\Delta < S_{j-1}] = p.$$

Next we have $\Pr[\text{IND}^{\mathsf{ore}}_{1,\Pi,\mathcal{L}_\mathsf{f}}(\mathcal{A}_j) = 1] =$

$$\begin{aligned}
&\Pr[\delta_i < S_{j-1}] \\
&\quad = \Pr[\delta_i < S_{j-1} | \Delta < S_{j-1}] \Pr[\Delta < S_{j-1}] \qquad (1) \\
&\quad + \Pr[\delta_i < S_{j-1} | S_{j-1} \leq \Delta < S_j] \Pr[S_{j-1} \leq \Delta < S_j] \qquad (2) \\
&\quad + \Pr[\delta_i < S_{j-1} | \Delta \geq S_j] \Pr[\Delta \geq S_j]. \qquad (3)
\end{aligned}$$

We consider (1-3) individually. First, (1) is at least $\Pr[\Delta < S_{j-1}] = p$. Second, for (2), we have

$$\Pr[\delta_i < S_{j-1} | S_{j-1} \leq \Delta < S_j] \geq 1/2$$

because when Δ is bounded by S_j, at most half of the δ_i can be at least $S_{j-1} = S_j/(M-1)$. By the assumption for contradiction, we also have

$$\Pr[S_{j-1} \leq \Delta < S_j] > 4\epsilon,$$

we gives that (2) is at least 2ϵ. Finally (3) is at least 0 trivially. Putting these together we get that $\Pr[\delta < S_{j-1}] > p + \epsilon$.

We now get that the advantage of $\mathcal{A}$ is strictly greater than

$$\frac{1}{2}|p - p - 2\epsilon| = \epsilon.$$

When we take t_0 in the theorem to be larger than the running time of $\mathcal{A}_j$, we get the desired contradiction and the lemma is proved. □

We can now complete the proof of Theorem 6.2. Using the above lemma with all $j = 1, , \ldots, \frac{1}{8\epsilon}$, along with a union bound, we get

$$\Pr[\Delta < (M-1)(\frac{M-1}{2})^{\frac{1}{8\epsilon}}] =\leq \frac{1}{8\epsilon} \cdot 4\epsilon = 1/2$$

Since the probability that Δ is greater than this bound is non-zero, we must that N satisfies the conclusion of the theorem. □

Game χ-$\mathsf{OW}^{\mathsf{ORE}}_\Pi(\mathcal{A})$:

$(\mathsf{sk}, \mathsf{ck}) \xleftarrow{\$} \mathcal{K}(1^\lambda)$
$\vec{m} \leftarrow \chi$
$\vec{c} = \mathcal{E}(\mathsf{sk}, \vec{m})$
$m' = \mathcal{A}(\mathsf{ck}, \vec{c})$
Return 1 if m' hits one of the messages in $\vec{m}$.

Figure 4: Game χ-$\mathsf{OW}^{\mathsf{ORE}}_\Pi(\mathcal{A})$.

6.2 One-Wayness Analysis

In this section, we illustrate a new security notion to analyze quality of ORE schemes under leakage. First, we define the notion χ-one-way security where χ is an efficiently samplable distribution that outputs a vector of messages as the plaintexts. This notion captures how easy/hard the adversary can recover any plaintext given encryptions of the plaintexts sampled by χ. Under this notion, we can paraphrase an analysis in CLWW as that their scheme is uniform-one-way secure, e.g. setting χ to be the uniform distribution. Next, we define a different yet natural distribution χ^* and showed a separation between CLWW (and our first construction) and our more generalized construction. That is, both CLWW and our basic ORE are not χ^*-one-way-secure, but our generalized ORE is χ^*-one-way-secure.

6.2.1 χ-One-wayness

We first introduce the notion of χ-one-way-security. Let χ be an efficiently samplable distribution that outputs a vector of messages $\vec{m}$. Then we define the security notion as follow:

DEFINITION 6.4. *Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{C})$ be an ORE scheme. For an adversary $\mathcal{A}$ we define the game χ-$\mathsf{OW}^{\mathsf{ORE}}_\Pi(\mathcal{A})$ in Figure 4. The advantage of $\mathcal{A}$ is defined as:*

$$\mathbf{Adv}^{\mathsf{ORE}}_{\Pi,\mathcal{A}}(\lambda) = \Pr[\chi\text{-}\mathsf{OW}^{\mathsf{ORE}}_\Pi(\mathcal{A}) = 1].$$

We say that Π is χ-one-way if for all efficient adversaries $\mathcal{A}$, $\mathbf{Adv}^{\mathsf{ORE}}_{\Pi,\mathcal{A}}(\lambda)$ is negligible.

Next we introduce a natural distribution χ^* samples from domain $D = \{0,1\}^n$: χ^* takes 1^n as input, samples $m_0 \xleftarrow{\$} D$ uniformly at random. Then it samples $m_i \xleftarrow{\$} D$ uniformly at random conditioned on $\mathsf{MSDB}(m_0, m_i) = i$ for $i \in [n]$. After that χ^* picks a random permutation π and outputs $(m_0, m_{\pi(1)}, \ldots, m_{\pi(n)})$.

THEOREM 6.5. *Neither CLWW nor our basic ORE is χ^* secure.*

PROOF. According to CLWW's construction, given two ciphers $(\mathcal{E}(m_1), \mathcal{E}(m_2))$, the attacker will know the exact value of i-th bit, where $i = \mathsf{MSDB}(m_1, m_2)$. Hence, by the definition of χ^*, it's apparent that the adversary can recover one message. Thus the construction of CLWW is not χ^*-one way secure.

For our basic ORE, we describe an attack illustrated in figure 5:

According to the attack, we note that for the first bit b, in $S = \{c_1, \ldots c_n\}$, there exist only one cipher c^* encrypted by a message m^* with first bit $1-b$, hence according to the ORE construction, for any other $c' \in S \backslash \{c^*\}$, $P(c^*, c) = P(c^*, c')$. And if $\mathcal{C}(c, c^*) = 1$, which means that $m > m^*$, referring to $b = 1$, and vice verse. For the i-th bit, due to in each step, S will rule out one cipher, the plaintext of the cipher in S shares the same prefix of length $i-1$, so we can view the i-th bit as the first one. Recursively, the attack works. □

Attack on χ-$\mathsf{OW}_{\Pi}^{\mathsf{ORE}}(\mathcal{A})$:
Given $c, (c_1, \ldots, c_n)$
$S \leftarrow \{c_1, \ldots, c_n\}$
For $i \leftarrow 1$ to n
$c^* \leftarrow S$ s.t. $\forall c' \leftarrow S \setminus \{c^*\}$
$P(c^*, c) = P(c^*, c')$
$S \leftarrow S \setminus \{c^*\}$
If $\mathcal{C}(c, c^*) = 1, b_i = 1$
Else $b_i = 0$
Return $b_1 || \ldots || b_n$

Figure 5: Attack on χ-$\mathsf{OW}_{\Pi}^{\mathsf{ORE}}(\mathcal{A})$.

THEOREM 6.6. *The generalized ORE* Π *with* $D = \{0,1\}^n$ *is* χ^**-one way secure if* Π *is simulation based secure with respect to leakage profile* $\mathcal{L}_f$.

PROOF. If Π is simulation based secure with respect to leakage profile $\mathcal{L}_f$, then given adversary a vector of ciphers $\vec{c}$ encrypted by $\vec{m}$, the entropy of any $m^* \in \vec{m}$, computationally, we have:

$$H(m^*|\vec{c}) = H(m^*|\mathcal{L}_{f(\vec{m})}).$$

Hence, the problem can be reduced to the problem of calculating the entropy by given leakage profile.

For our generalized ORE, which encrypts d-bit by d-bit ($d > 1$ and we denote $\ell = n/d, T = 2^d - 1$ for short), we have each cipher contains ℓ PPH hash values, where PPH testing P_d. And for two d-bit blocks B_1, B_2, we define B_1, B_2 have the same weight, if they contain the same number of "1" in their binary representation, for instance, (010) and (100) have the same weight. If $\vec{m} \xleftarrow{\$} \chi^*(1^n)$, then we denote $\bar{m}$ as the first message in $\vec{m}$, then:

LEMMA 6.7. *Let* $\vec{m_1}, \vec{m_2} \xleftarrow{\$} \chi^*(1^n)$ *and* $\bar{m_1}$ *are* $\bar{m_2}$ *are the same except one* d*-bit block, and the differ block* D_1, D_2 *has the same weight, then* $\mathcal{L}_f(\vec{m_1})$ *and* $\mathcal{L}_f(\vec{m_2})$ *are statistically identical.*

PROOF. Without loss of generality, we assume $\bar{m_1}$ and $\bar{m_2}$ are differ in the first block, then in $\vec{m_1}$ and $\vec{m_2}$ there are only t message have distinct first block, and we denote them as $s_1, \ldots, s_d$ and $s'_1, \ldots, s'_d$ respectively. And we denote $s_{d+1}, \ldots, s_n$ and $s'_{d+1}, \ldots, s'_n$ as the rest messages in $\vec{m_1}$ and $\vec{m_2}$. Hence the leakage profile is, for any $i, k \in [d], l_1, l_2 \in [d+1, n], j, k \neq i$:

$$P(s_i, \bar{m_1}) = P(s_i, s_j), P(\bar{m_1}, s_i) = P(\bar{m_1}, s_k)$$
$$P(s'_i, \bar{m_2}) = P(s'_i, s'_j), P(\bar{m_2}, s'_i) = P(\bar{m_2}, s'_k)$$
$$P(s_i, s_{l_1}) = P(s_i, s_{l_2}), P(s'_i, s'_{l_1}) = P(s'_i, s'_{l_2})$$

Since D_1, D_2 have the same weight, the counting on $1(s_i, \bar{m_1})$ and $1(s'_i \bar{m_2})$ are the same. Besides, $\bar{m_1}$ are $\bar{m_2}$ are the same except one d-bit block, it's obvious that the rest of the leakage profile are exactly the same.

Thus, due to the randomness of permutation used in χ^*, the leakage profile $\mathcal{L}_f(\vec{m_1})$ and $\mathcal{L}_f(\vec{m_2})$ are statistically identical □

Applying the result of Lemma 6.7, we see that given the leakage profile $\mathcal{L}_f(\vec{m})$, the $\bar{m}$ can be any message, as long as the corresponding block has the same weight. Hence the entropy:

$$H(\bar{m}|\mathcal{L}_f(\vec{m})) = (\sum_{i=0}^{d} \frac{C_d^i}{2^d} \frac{n}{d} log(C_d^i))n = \sum_{i=0}^{d} \frac{C_d^i log(C_d^i) n}{d \cdot 2^d}$$

For $m_j \xleftarrow{\$} \vec{m}$ where $\mathsf{MSDB}(\bar{m}, m_j) = j, j \in [n]$, within the same analysis we have:

$$H(m_j|\mathcal{L}_f(\vec{m})) = n - j + \sum_{i=0}^{d} \frac{C_d^i \cdot log(C_d^i) j}{d \cdot 2^d} \geq H(\bar{m}|\mathcal{L}_f(\vec{m}))$$

Specifically $d = 2$ then the entropy is $0.25n$, $d = 3$ the entropy is $\frac{\log_2(3)n}{4} \approx 0.39n$, and it increase as d increases. □

6.3 Combinatorial Analysis

Here we initiate the study of the combinatorial structure of our leakage profile. This gives a qualitative feel for the improvement in the leakage and may be useful in future work. For simplicity, we focus on the leakage profile of the basic scheme.

Given n encryptions of plaintext values that each have length m, we view the leakage on these ciphertexts as a function $\mathcal{L}_f : \{0,1\}^{mn} \to \{0,1\}^{n^3}$, and we give an upper bound on the number of possible outputs from the leakage function. A naive upper bound is $2^{O(n^3)}$; we show that this can be tightened to $2^{m^2 \log n}$. Note that each output bit of $\mathcal{L}_f$ depends on only 3 of plaintext values, so we will abuse notation in the rest of the section and treat $\mathcal{L}_f : \{0,1\}^{3n} \to \{0,1\}$, once we've fixed three input values.

Consider the number of ways that we can partition n ordered values into m buckets while preserving their order. This value is $\binom{n+1}{m}$, because we can place each of the "bucket lines" in one of the $n+1$ gaps between the plaintext. We partition our plaintext values by placing value x in bucket k whenever $2^k < x < 2^{k+1}$ (for $k \in \{0, \ldots m-1\}$). We will use $b(x)$ to denote the bucket that element x is assigned to. Consider the output of $\mathcal{L}_f$ in each of the following cases.

case 1: $b(x_1) \leq b(x_2) < b(x_3)$. In this case, $\mathcal{L}_f(x_1, x_2, x_3) = 1$, since the most significant 1 in x_3 comes before the most significant 1 in either x_1 or x_2.

case 2: $b(x_1) < b(x_2) = b(x_3)$. In this case, $\mathcal{L}_f(x_1, x_2, x_3) = 0$.

case 3: $b(x_1) = b(x_2) = b(x_3)$. In this case, the value of $\mathcal{L}_f(x_1, x_2, x_3)$ is still undetermined.

In the first two cases, we stress that the output of $\mathcal{L}_f$ is fully determined by *any* triplet satisfying those cases, so fixing the partition of elements to buckets suffices for fixing the output of $\mathcal{L}_f$ on all such inputs. For all triplets satisfying the final case, we can recurse, stripping off the most significant bit, and partitioning again into $m-1$ buckets. Clearly after $m-2$ recursive steps, all remaining input values will satisfy the first 2 cases, since the inputs are only 2 bits long. Since there are $O(n^m) = O(2^{m \log n})$ possible partitions in the first step, and we recurse at most m times, we have that there are at most $O(2^{m^2 \log n})$ possible input configurations, each mapping to a unique output of $\mathcal{L}_f$.

Acknowledgements

We thank Dov Gordon for the analysis in Section 6.3, and Mark Zhandry for helpful comments.

7. REFERENCES

[1] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. Order preserving encryption for numeric data. In *SIGMOD*, 2004.

[2] A. Arasu, S. Blanas, K. Eguro, R. Kaushik, D. Kossmann, R. Ramamurthy, and R. Venkatesan. Orthogonal security with cipherbase. In *CIDR*, 2013.

[3] A. Arasu, K. Eguro, R. Kaushik, and R. Ramamurthy. Querying encrypted data (tutorial). In *ICDE*, 2013.

[4] S. Bajaj and R. Sion. Trusteddb: A trusted hardware-based database with privacy and data confidentiality. *TKDE*, 26(3):752–765, 2014.

[5] M. Bellare, A. Boldyreva, and A. O'Neill. Deterministic and efficiently searchable encryption. In A. Menezes, editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pages 535–552. Springer, Heidelberg, Aug. 2007.

[6] A. Boldyreva, N. Chenette, Y. Lee, and A. O'Neill. Order-preserving symmetric encryption. In A. Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 224–241. Springer, Heidelberg, Apr. 2009.

[7] A. Boldyreva, N. Chenette, and A. O'Neill. Order-preserving encryption revisited: Improved security analysis and alternative solutions. In P. Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 578–595. Springer, Heidelberg, Aug. 2011.

[8] D. Boneh, K. Lewi, M. Raykova, A. Sahai, M. Zhandry, and J. Zimmerman. Semantically secure order-revealing encryption: Multi-input functional encryption without obfuscation. In E. Oswald and M. Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 563–594. Springer, Heidelberg, Apr. 2015.

[9] R. Canetti. Towards realizing random oracles: Hash functions that hide all partial information. In B. S. Kaliski Jr., editor, *CRYPTO'97*, volume 1294 of *LNCS*, pages 455–469. Springer, Heidelberg, Aug. 1997.

[10] D. Cash, P. Grubbs, J. Perry, and T. Ristenpart. Leakage-abuse attacks against searchable encryption. In *CCS*, 2015.

[11] D. Cash, S. Jarecki, C. S. Jutla, H. Krawczyk, M.-C. Rosu, and M. Steiner. Highly-scalable searchable symmetric encryption with support for boolean queries. In R. Canetti and J. A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 353–373. Springer, Heidelberg, Aug. 2013.

[12] Y.-C. Chang and M. Mitzenmacher. Privacy preserving keyword searches on remote encrypted data. In *ACNS*, 2005.

[13] M. Chase and S. Kamara. Structured encryption and controlled disclosure. In M. Abe, editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 577–594. Springer, Heidelberg, Dec. 2010.

[14] N. Chenette, K. Lewi, S. A. Weis, and D. J. Wu. Practical order-revealing encryption with limited leakage. Cryptology ePrint Archive, Report 2015/1125, 2015. `http://eprint.iacr.org/2015/1125`.

[15] N. Chenette, K. Lewi, S. A. Weis, and D. J. Wu. Practical order-revealing encryption with limited leakage. In *FSE*, 2016. To appear.

[16] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky. Searchable symmetric encryption: improved definitions and efficient constructions. In *CCS*, 2006.

[17] J. L. Dautrich Jr and C. V. Ravishankar. Compromising privacy in precise query protocols. In *EDBT*, 2013.

[18] E.-J. Goh et al. Secure indexes. *IACR Cryptology ePrint Archive*, 2003:216, 2003.

[19] V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute-based encryption for fine-grained access control of encrypted data. In A. Juels, R. N. Wright, and S. Vimercati, editors, *ACM CCS 06*, pages 89–98. ACM Press, Oct. / Nov. 2006. Available as Cryptology ePrint Archive Report 2006/309.

[20] H. Hacigümüş, B. Iyer, C. Li, and S. Mehrotra. Executing sql over encrypted data in the database-service-provider model. In *Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data*, SIGMOD '02, pages 216–227, New York, NY, USA, 2002. ACM.

[21] B. Hore, S. Mehrotra, M. Canim, and M. Kantarcioglu. Secure multidimensional range queries over outsourced data. *VLDBJ*, 21(3):333–358, 2012.

[22] M. S. Islam, M. Kuzu, and M. Kantarcioglu. Access pattern disclosure on searchable encryption: Ramification, attack and mitigation. In *NDSS*, 2012.

[23] M. S. Islam, M. Kuzu, and M. Kantarcioglu. Inference attack against encrypted range queries on outsourced databases. In *CODASPY*, 2014.

[24] S. Kamara. How to search on encrypted data, 2015. `https://cs.brown.edu/~seny/slides/encryptedsearch-full.pdf`.

[25] S. Kamara and T. Moataz. Sql on structurally-encrypted databases. Cryptology ePrint Archive, Report 2016/453, 2016. `http://eprint.iacr.org/`.

[26] C. Liu, L. Zhu, M. Wang, and Y.-a. Tan. Search pattern leakage in searchable encryption: Attacks and new construction. *Information Sciences*, 265:176–188, 2014.

[27] A. Mandal and A. Roy. Relational hash: Probabilistic hash for verifying relations, secure against forgery and more. In R. Gennaro and M. J. B. Robshaw, editors, *CRYPTO 2015, Part I*, volume 9215 of *LNCS*, pages 518–537. Springer, Heidelberg, Aug. 2015.

[28] P. Mohassel and M. Franklin. Efficiency tradeoffs for malicious two-party computation. In M. Yung, Y. Dodis, A. Kiayias, and T. Malkin, editors, *PKC 2006*, volume 3958 of *LNCS*, pages 458–473. Springer, Heidelberg, Apr. 2006.

[29] M. Naveed, S. Kamara, and C. V. Wright. Inference attacks on property-preserving encrypted databases. In *CCS*, 2015.

[30] O. Pandey and Y. Rouselakis. Property preserving symmetric encryption. In D. Pointcheval and T. Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 375–391. Springer, Heidelberg, Apr. 2012.

[31] R. A. Popa, C. M. S. Redfield, N. Zeldovich, and H. Balakrishnan. Cryptdb: Protecting confidentiality with encrypted query processing. In *SOSP*, 2011.

[32] D. X. Song, D. Wagner, and A. Perrig. Practical techniques for searches on encrypted data. In *SP*, 2000.