A Formal Information-Theoretic Leakage Analysis of Order-Revealing Encryption

Mireya Jurado, Catuscia Palamidessi, Geoffrey Smith

A Formal Information-Theoretic Leakage Analysis of Order-Revealing Encryption

Mireya Jurado
Florida International University
Miami, USA
mjura011@fiu.edu

Catuscia Palamidessi
Inria and *École Polytechnique*
Palaiseau, France
catuscia@lix.polytechnique.fr

Geoffrey Smith
Florida International University
Miami, USA
smithg@cis.fiu.edu

***Abstract*—Order-Revealing Encryption (ORE) allows deriving the order of two plaintexts to facilitate database functions such as range queries and sorting. Ideally, nothing is observable to an adversary beyond the order of the messages. Unfortunately, Ideal ORE is challenging to implement, and a variation of it has then been developed. This variation, referred to as CLWW ORE, reveals the first differing bit position between every two plaintexts, in addition to the order.**

We provide a formal leakage analysis of these two ORE variations by applying the information-theoretic quantitative information flow (QIF) framework. We evaluate two threat models: (1) the Bayes scenario in which an adversary wishes to guess the secret entirely and (2) a bucketing scenario in which an adversary is content to simply guess the range of the plaintext. We provide security implications, usage guidelines, and a mitigation technique that improves the security of Ideal ORE. We find that while Ideal and CLWW ORE perform similarly under the Bayes scenario, CLWW ORE is fundamentally insecure under the bucketing scenario.

***Keywords*-Quantitative Information Flow, Order-Revealing Encryption, Formal Security Models, Cloud Computing Security**

I. Introduction

Cloud-hosted databases are attractive for reasons of flexibility, reliability, and economy, but they raise obvious security concerns. Even if a cloud server can be trusted to maintain a database's integrity, there is still the worry that an "honest but curious" server might compromise the confidentiality of sensitive database entries. For this reason, it would be desirable to encrypt such entries with a symmetric encryption scheme whose key is known to the user but not to the database server.

Strong encryption, such as IND-CPA, provides the desired security guarantees, but it does not allow the server to determine *any* property of the underlying plaintexts. As a consequence, the server would not be able to execute database queries. For this reason, cryptographers have proposed various forms of *property-revealing encryption*, which are deliberately weakened to allow certain properties of plaintexts to be efficiently computed from their ciphertexts.

In this paper we focus on *order-revealing encryption* (ORE) [1]. An ORE scheme provides an algorithm that, given two ciphertexts c and c' coming from plaintexts v and v', decides whether $v < v'$ or $v = v'$ or $v > v'$. Crucially, ORE enables a cloud server to perform range queries on encrypted data. Indeed, consider a database column containing ORE-encrypted values of some kind for n users, and suppose that we wish to retrieve the rows of all users whose value is between some lower bound ℓ and upper bound u. To achieve this goal, it suffices to encrypt ℓ and u and send them to the server, which can use the comparison algorithm on the encrypted bounds and the encrypted values in the column, retrieving those rows whose value is between ℓ and u. There is a pressing demand for encryption schemes allowing this kind of cloud services, and order-revealing encryption schemes are actively deployed in enterprise solutions [2].

Since with ORE encryption, the server never sees any plaintexts at all, one might hope that the confidentiality of the data is ensured. Unfortunately, as pointed out by Naveed, Kamara, and Wright [3], ORE is not exempt from the risk of plaintext disclosure. Indeed, suppose that there are k possible values, ranging from 0 to $k-1$. ORE enables the server to count the number of distinct values in the column, and if this number happens to be k, then the server can deduce all the entries in the column.

The disclosure described above is a risk already present in the most secure ORE scheme, namely *Ideal ORE*, which leaks *nothing* about the column entries other than their ordering information. One may argue that such a situation would be extremely rare, and that therefore *Ideal ORE* is pretty secure. Unfortunately, however, cryptographers have not yet found an efficient way to implement Ideal ORE — so far implementations require cryptographic multilinear maps [1] which are too inefficient for practical use. To address this problem, Chenette, Lewi, Weis, and Wu [4] proposed a scheme, called CLWW ORE (from the initials of the authors), that is efficient but which leaks more than Ideal ORE. Specifically, given two ciphertexts, CLWW ORE also leaks the *most-significant bit position* at which the two underlying plaintexts differ. This is useful information for the adversary — for instance, if the first difference is in a low-order bit, then the two entries must be close in value, while if the difference is in a high-order bit, then the two entries might be far apart. The authors of [4] claim that their scheme "reveals slightly more information than just the ordering of the underlying messages" — but what, precisely, does "slightly more" mean here? They argue that their construction explicitly models the leakage and provides "a precise quantification of any information leaked by the scheme". In our opinion the scheme does indeed provide a concrete leakage profile, but it does not account for the amount

of information that can be inferred by a potential attacker.

In order to compare the leakage of CLWW ORE with that of Ideal ORE — that we regard as a baseline for ORE schemes — and assess whether the gain in efficiency is worth the loss of security, it is clearly important to measure the leakage of information in a precise way and according to a meaningful metric. We undertake this challenge and propose an analysis of both Ideal ORE and CLWW ORE by means of *quantitative information flow* (QIF) [5].

QIF is an information-theoretic framework in which a system is modeled as an *stochastic channel* that takes in input a secret X and outputs an observable Y. The measure of "leakage" of the system represents how much Y lets an optimal adversary deduce about X. More precisely, the leakage is measured by comparing the *prior vulnerability* of X when the adversary knows only its prior distribution, to the *posterior vulnerability* of X when she[1] also knows Y. The precise definition of vulnerability depends on a parameter that represents the operational scenario, comprising the adversary's capabilities and goals. An important point about QIF analysis is that it is an instance of what can be called *universal security*: it gives strong security guarantees by showing that *all* adversaries within a certain class can achieve at most a certain level of success with respect to some goal.[2]

In our analyses of ORE, we take the secret input X to be a column of n database values, each in the set $\{0, 1, \ldots, k-1\}$ for some $k \geq 1$. We assume that each entry in the column is chosen independently according to some distribution δ. The prior on X will be therefore the distribution determined by the product of δ with itself n times. The observable output Y is the column of ORE encryptions of the entries in X that can be observed from a database "snapshot".[3]

We consider two kinds of operational scenarios. In the first one, the adversary aims to guess all the entries of X exactly. This situation is represented by the so-called *Bayes vulnerability*. In the second one, the adversary settles for guessing the entries approximately, i.e., within a certain range. The corresponding measure is called *bucketing vulnerability*. We argue that this second case represents a realistic objective for the adversary, notably in case of quantitative data, like salary, age, etc. We carry on our analysis under the assumption of a uniform prior, which gives the maximum leakage in the case of the Bayes vulnerability [6]. Furthermore, the *Miracle theorem* [7] ensures that the Bayes leakage on the uniform prior is an upper bound to all other kinds of leakage (i.e., for any other gain function).

The results of our analysis are that, while under the first threat model the security of the two schemes is comparable, under the second one CLWW ORE leaks catastrophically more than Ideal ORE.

[1]We arbitrarily use "she/her" pronouns for the adversary.

[2]In contrast, finding a new attack (e.g. using a machine-learning algorithm) is an instance of *existential (in)security* — it shows that there *exists* an adversary within a certain class that achieves at least a certain level of success with respect to some goal.

[3]Thus we do not consider the additional leakage that might result from range queries where the user requests records between a lower and upper boundary.

A. Summary of contributions

The contributions of our work are as follows:

1) We provide efficient techniques to compute posterior Bayes vulnerability on a uniform prior for both Ideal ORE and CLWW ORE, for arbitrary column length n and number k of possible values. We find that the posterior Bayes vulnerability of both Ideal ORE and CLWW ORE depends heavily on the "density" of the column — roughly, if the number n of column entries is no bigger than the number k of possible values, meaning that the column is fairly "sparse", then the posterior Bayes vulnerability is small. But if n is much larger than k, then the posterior Bayes vulnerability approaches 1, meaning that the adversary can almost surely guess the entire column correctly. Moreover we find that, with respect to Bayes vulnerability, CLWW ORE is indeed not too much worse than Ideal ORE. Finally, we give a simple formula that upper bounds the posterior Bayes vulnerability of Ideal ORE, thus providing easily understood "usage guidelines".
2) We consider a family of bucketing vulnerabilities, in which the adversary just wants to guess the column entries approximately. This is a meaningful measure in those scenarios in which the adversary is interested in the magnitude of the values, and does not care too much about the low-order bits. For instance, this could be the case when the entries are salaries, assets, etc. We give efficient techniques for computing the various posterior bucketing vulnerabilities (depending on how many of the bits the adversary is trying to guess) for both Ideal ORE and CLWW ORE. Here we find an enormous difference: assuming that the column is not too dense, we find that as the number of bits being guessed decreases, the posterior bucketing vulnerability of Ideal ORE increases slowly, while the posterior bucketing vulnerability of CLWW ORE increases explosively. Because the bucketing vulnerability scenario is so natural, we conclude that CLWW ORE is fundamentally insecure and should not be used.
3) We propose a mitigation technique to improve the security of ORE. The technique consists of padding each entry with some number of random bits before encrypting it. We find that this technique greatly decreases the posterior Bayes and bucketing vulnerability of Ideal ORE, enabling it to be used safely even in the case of a "dense" column where n is much larger than k. Unfortunately, this random padding does not help at all in the case of CLWW ORE.

B. Related Work

Introduced by Agrawal et al. [8] in 2004, order-preserving encryption (OPE) ensures that the encryption of two ordered plaintexts produces ciphertexts that preserve that order.

Boldyreva et al. [9] introduced an achievable security notion for OPE based on pseudorandom functions, which we consider here. In 2015, Boneh et al. introduced a generalization called order-revealing encryption (ORE) [1]. Instead of preserving the order in the ciphertexts, ORE applies a publicly computable comparison function on the ciphertexts that outputs the order of the underlying plaintexts. Importantly, we consider ORE a category encompassing most OPE schemes since the order is maintained in the ciphertext. (There are some schemes that prevent equality testing by the server. For example, both Liu and Wang [10] and Dyer et al. [11] construct schemes that add random noise to the initial plaintext so that if two plaintexts are equal, their ciphertext order is randomized.) There are many order-preserving and order-revealing schemes [8], [12], [1], [4], [11], [13], [14], [15], [16], [17], [10], [18], [19], [20], but fundamentally, an *Ideal ORE* scheme only leaks order. As explained in the introduction, there are no known efficient implementations of Ideal ORE, which motivated Chenette, Lewi, Weis, and Wu to construct a scheme that trades security for efficiency [4].

Many inference attacks targeting property-preserving or order-revealing encryption have been presented [8], [21], [22], [23], [24]. Durak, DuBuisson and Cash [25] empirically evaluate Ideal and CLWW ORE (which they refer to as the most-significant-differing bit profile) in a multi-column setting under the assumption that the adversary receives no auxiliary information. While we limit our analysis to the single-column setting, we make the stronger assumption that the adversary has full knowledge of a prior distribution on secret values. Grubbs et al. [26] also find that CLWW ORE performs poorly in an experimental attack in a single-column setting. They call for a formal analysis of ORE leakage, which this work answers. Bogatov, Lollio, and Reyzon [27] find that CLWW ORE is one of the fastest ORE schemes, although it has relatively worse security as it leaks the most-significant differing bits. We characterize this leakage precisely using novel combinatorics and we find that the gains in speed may not justify the security cost.

Our analysis does not evaluate the success of a particular attack. Instead, we calculate the vulnerability of the secret with respect to an *optimal* adversary. For example, experimental research tests how well a specific machine learning or graph-based attack strategy accomplishes an adversarial goal. Our analysis does not rely on a particular method but instead evaluates what the best possible adversary is able to accomplish. While previous works illustrated security flaws with Ideal and CLWW ORE using experimental methodology, we provide a formal analysis of these flaws using the QIF framework.

A work that is similar to ours in spirit is [28]. That paper applies the QIF methodology to the analysis of the leakage of deterministic encryption. It considers three different operational scenarios, including the one of our paper, namely the case in which the attacker is interested in guessing the entire column. Like ORE, deterministic encryption is a property-revealing encryption, however that kind of leakage is very different, and the results of [28] do not bear any implication for the leakage of ORE.

C. Outline of the paper

The rest of the paper is organized as follows: The next section recalls background information about QIF and CLWW ORE, Section III gives our Bayes analyses of Ideal ORE and CLWW ORE, Section IV gives our bucketing analyses of Ideal ORE and CLWW ORE, Section V discusses our mitigation technique, Section VI provides some discussion of possible future directions, and Section VII concludes.

Due to space constraints, all proofs are in the appendix.

II. Background

The intent of this section is to present an introduction to the QIF framework and some background about Ideal ORE and CLWW ORE. Readers who are familiar with these concepts are encouraged to move to Section III.

A. Quantitative Information Flow Framework

In order to precisely model and analyze the leakage of these ORE schemes, we apply the quantitative information flow (QIF) framework. The definitive resource on this framework with definitions, theorems, and proofs can be found in [5]. For this document, we outline fundamental concepts.

QIF is an information-theoretic framework that models a system taking a secret input from $\mathcal{X}$ and producing some observable output in $\mathcal{Y}$, according to some probability distribution. We assume that the input comes with its own probability distribution, and we will denote by X and Y the random variables associated to the input and the output, respectively. The system can be statically modeled as an information-theoretic channel matrix M, where $\mathsf{M}_{x,y}$ contains the conditional probability $p(y|x)$ of $y : \mathcal{Y}$ given $x : \mathcal{X}$. We make the following assumptions: (1) the adversary knows the probability distribution on secrets and (2) the adversary knows how the channel works and can compute all the $\mathsf{M}_{x,y}$. With this information, she can update her knowledge about X to a posterior distribution $p(x|y)$. Since each output y also has a probability $p(y)$, the channel matrix M provides a mapping from any prior π to *distributions on posterior distributions*, which we call a hyper-distribution and denote $[\pi \triangleright \mathsf{M}]$.

The g-leakage framework, introduced in [7], defines the *vulnerability* of X with respect to specific operational scenarios. An adversary is described by a set $\mathcal{W}$ of *guesses* (or *actions*) that she can make about the secret, and by a *gain function* $g(w, x)$ expressing the gain of selecting the guess w when the real secret is x. An optimal adversary will choose a guess that maximizes her expected gain with respect to π. From this gain function, a secret's vulnerability can be determined.

We distinguish between *prior* and *posterior* g-vulnerability. Given a prior distribution on secrets π, prior g-vulnerability, denoted $V_g(\pi)$, represents the adversary's *expected gain* of her optimal guess based only on π, i.e., before observing to the channel output. Formally:

$$V_g(\pi) := \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \pi_x \cdot g(w, x)\,. \tag{1}$$

In the posterior case, the adversary observes the output of the system which allows her to improve her guess and consequent expected gain. Posterior g-vulnerability, denoted $V_g[\pi \triangleright \mathsf{M}]$, is defined as follows:

$$V_g[\pi \triangleright \mathsf{M}] := \sum_{y \in \mathcal{Y}} \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \pi_x \cdot \mathsf{M}_{xy} \cdot g(w, x)\,. \qquad (2)$$

The choice of the gain function g allows the modeling of a variety of adversarial scenarios. The simplest case is the *identity* gain function, given by $\mathcal{W} = \mathcal{X}$, $g_1(w, x) = 1$ iff $x = w$ and 0 otherwise. This gain function models an adversary who tries to guess the secret exactly in one try; V_1 is the *Bayes-vulnerability*, which corresponds to the complement of the Bayes error [7].

However, the interest in g-vulnerability lies in the fact that many more adversarial scenarios can be captured by a proper choice of g. Several gain functions are discussed in [7], while [29] shows that *any* vulnerability function satisfying basic axioms can be expressed as V_g for a properly constructed g.

Of particular interest for this paper is the *bucketing vulnerability*, which in its basic form is characterized by $\mathcal{W}$ being a partition of $\mathcal{X}$, corresponding to the buckets, and a gain function returning 1 whenever $x \in w$, and 0 otherwise. In our case, the secret is a vector $\vec{x} = (x_1, x_2, \ldots x_n)$ of values in $\{0, 1, \ldots, k-1\}$, and the guess is a vector of buckets $\vec{w} = (w_1, w_2, \ldots w_n)$, one for each value, where each w_i is an element of a fixed partition of $\{0, 1, \ldots, k-1\}$. Our bucketing gain function g_{buck} is then defined as follows:

$$g_{\text{buck}}(\vec{w}, \vec{x}) \;:=\; \begin{cases} 1 & \text{if } \forall i \in \{1, 2, \ldots, n\}.\ x_i \in w_i \\ 0 & \text{otherwise} \end{cases}$$

We measure the channel leakage by comparing the prior and posterior g-vulnerability, which quantifies how much a specific channel M *increases* the vulnerability of the system. This comparison can be done additively or multiplicatively, but in this paper we only consider the multiplicative case.[4]

Definition 1 (Multiplicative g-Leakage)**.** *Given prior probability distribution π, a gain function g, and channel M, the* multiplicative g-leakage *is*

$$\mathcal{L}_g^{\times}(\pi, \mathsf{M}) := \frac{V_g[\pi \triangleright \mathsf{M}]}{V_g(\pi)}$$

□

Crucially for this work, in [30] it was proved that the posterior Bayes vulnerability of a deterministic channel under a uniform prior is simply the number of possible outputs divided by the number of possible inputs.

Proposition 1 ([30])**.** *Let $\boldsymbol{\vartheta}$ be a uniform prior distribution on an N-element set $\mathcal{X}$ and let M be a deterministic channel with $|\mathcal{Y}|$ possible output values. Then $V_1[\boldsymbol{\vartheta} \triangleright \mathsf{M}] = |\mathcal{Y}|/|\mathcal{X}|$ and $\mathcal{L}_1^{\times}(\boldsymbol{\vartheta}, \mathsf{M}) = |\mathcal{Y}|$.*

[4]In the original paper, the multiplicative version of g-leakage was defined as the log of the definition given here. In recent literature, however, the log is not used anymore. Anyway, the two definitions are equivalent for comparing systems, since log is a monotonic function.

B. Ideal ORE and CLWW ORE

Following the approach of [28], an information-theoretic analysis of cryptography is made simpler by analyzing an "ideal object," conjectured to be computationally indistinguishable from the cryptographic scheme. For Ideal ORE, the ideal object is a random permutation that is indistinguishable from random values subject to the constraint that an applied function reveals order between the values. The result is a probabilistic channel that maps a column of plaintext values $(x_1, x_2, \ldots, x_n)$ to a column of random independent values $(v_1, v_2, \ldots, v_n)$ subject to the constraint that the order of the original plaintext values can be determined: $x_i \leq x_j$ iff $f(v_i) \leq f(v_j)$ where f is a public function determining the order.

We can make our analysis easier when we consider that this probabilistic channel is equivalent to a deterministic channel that maps a column of plaintext values to an ordered partition of indices where all indices in a block are equal. For example, a column $(1, 2, 2, 5, 1)$ maps to $\{1, 5\} < \{2, 3\} < \{4\}$.

Unfortunately, it seems that schemes either do not achieve this ideal notion of security or are inefficient for practical use [1]. Hence researchers have considered schemes that leak somewhat more, aiming to achieve a good balance between security and efficiency. One such scheme is CLWW ORE [4]. It additionally leaks, for each pair of ciphertexts, the bit position at which their corresponding plaintexts first differ. Given the binary representation of a message $b_1 \cdots b_n$, for every index i, CLWW ORE applies a secure pseudorandom function (PRF) to the first $i-1$ bits then adds $b_i \pmod M$, where M is an integer greater than or equal to 3. The output is a tuple $(u_1, u_2, \ldots, u_n)$. Two messages that differ at index i will differ at u_i, thereby revealing the index of the first significant differing bit.

CLWW ORE provides an important security property: assuming that a secure PRF produces computationally indistinguishable outputs, CLWW ORE is computationally indistinguishable from a channel that reveals only (1) ordering information and (2) the first differing bit position for each pair of column entries.

III. Bayes vulnerability analysis of Ideal ORE and CLWW ORE

In this section, we analyze Ideal ORE and CLWW ORE with respect to Bayes vulnerability. We first make precise the two channels that we wish to analyze. In both cases, we take the secret input to be a database column X of n items, each independently chosen in the range $0, 1, \ldots, k-1$ according to a distribution δ, which hence induces a prior distribution π. We assume that $n \geq 1$ and $k \geq 1$.

A. The channel matrices

In the case of Ideal ORE, the channel output is the information revealed by the ciphertexts when the column is encrypted using Ideal ORE, which by definition reveals only the ordering information among the column entries. This information can be described compactly as a *partition* of the

indices $\{1, 2, \ldots, n\}$ into i blocks of indices containing equal values, where we have $1 \leq i \leq \min\{n, k\}$, together with a total ordering on the blocks.

For example, the channel matrix I when $n = 3$ and $k = 3$ is shown in Table I. Given values 0, 1, and 2, and a column size of 3, there are 27 possible columns that can be generated, represented by the rows of the matrix. For instance, the row $(2, 0, 0)$ represents the column where the first index has a value of 2, while the second and third indices have a value of 0. Given these 27 possible channel inputs, there are only 13 possible outputs that reveal the ordering of the indices. The column $(2, 0, 0)$ deterministically outputs the ordered blocks $\{2, 3\} < \{1\}$ which means that the values at second and third index are equal but less than the value in the first index. Since this column only produces this one output, $p(y|x) = 1$ thus the entry in the matrix is 1.

In the case of CLWW ORE, the output of the "ideal object" (which is computationally indistinguishable from the CLWW cryptographic scheme) is the ordering information among the column entries, together with the first bit positions at which each pair of entries differs.

If we let $\ell = \lceil \log_2 k \rceil$, then the values in $0, 1, \ldots, k-1$ can be written as ℓ-bit strings. We write the first differing bit position of strings a and b as $a\#b$, using 0 to indicate that the strings are equal. For example, $1101\#1111 = 2$ and $0110\#1001 = 4$ and $1101\#1101 = 0$.

Interestingly, however, the adversary does not actually benefit from being told the differing bit positions of every *pair* of plaintexts: if those plaintexts are written in increasing order, then it suffices to reveal just the differing bit positions of *adjacent* plaintexts. This follows from the following theorem.

Theorem 2. *If $a < b < c$, then we have $a\#c = \max\{a\#b, b\#c\}$.*

A proof is given in Appendix A. (Note that the ordering condition on the plaintexts is necessary: we have $1000\#0101 = 4$ and $0101\#1001 = 4$, but $1000\#1001 = 1$.)

This allows us a compact representation of the additional information output by CLWW ORE. If the column contains i distinct values, written in increasing order as $v_1, v_2, \ldots, v_i$, then the additional information can be written as a "report" R of the first differing bit positions of the adjacent values:

$$R \;=\; [v_1\#v_2, v_2\#v_3, \ldots v_{i-1}\#v_i]\,.$$

For example, with values $0001, 0111, 1010, 1011$, the report R is $[3, 4, 1]$.

Table II shows the channel matrix C for CLWW ORE when $n = 3$ and $k = 3$. In comparison with Ideal ORE, it has 19 outputs, rather than 13, sometimes enabling the adversary to make better guesses. For example, an adversary observing output $\{3\} < \{1, 2\}$ under Ideal ORE has three candidate columns: $(1, 1, 0)$, $(2, 2, 0)$, and $(2, 2, 1)$. However, under CLWW ORE she also observes either report $[1]$ or report $[2]$; if she observes $[1]$, she knows that the column *must* be $(1, 1, 0)$.

B. Computing posterior Bayes vulnerability

The main challenge for computing Bayes leakage is to compute the posterior Bayes vulnerability. For small values of n and k, this can be done with high assurance using a QIF-aware language Kuifje [31]. Indeed, posterior g-vulnerability can always be computed, in principle. In the case of Bayes vulnerability the time required becomes $O(|\mathcal{Y}| \cdot |\mathcal{X}|)$ — that is, in time proportional to the size of the channel matrix. However, both matrices we consider have k^n rows and an exponential number of columns, making the computation infeasible. Fortunately, we know from [6] that the maximum leakage for Bayes vulnerability is obtained in the case of uniform prior, and it is immediate to calculate the posterior Bayes vulnerability of a deterministic channel under the uniform prior: it is simply the number of possible outputs, $|\mathcal{Y}|$, divided by the number of possible inputs, $|\mathcal{X}|$ (cfr. Proposition 1). So, the uniform prior assumption allows us an efficient computation of the upper bound on the Bayes leakage on any prior (and *on any kind of leakage on any prior*, thanks to the Miracle theorem [7]).

The number of possible inputs $|\mathcal{X}|$ is simply k^n. To compute the number of possible outputs in the case of Ideal ORE, there is well-developed combinatorial theory that we can apply, namely *Stirling numbers of the second kind* and *ordered Bell numbers*.

Each output partitions the indices $\{1, 2, \ldots, n\}$ into i blocks, $1 \leq i \leq \min\{n, k\}$, where the blocks contain the indices that contain equal values. The number of such partitions is well known: it is by definition the Stirling number of the second kind $\left\{{n \atop i}\right\}$.[5] Moreover, the output also imposes a total order among the i blocks; there are $i!$ ways to do this. (See Table I for an explicit enumeration of the 13 possible outputs when $n = 3$ and $k = 3$.) Hence we have the following equation:

$$\mathcal{L}_1^{\times}(\boldsymbol{\vartheta}, \mathsf{I}) \;=\; \sum_{i=1}^{\min\{n,k\}} \left\{{n \atop i}\right\} i!\,. \tag{3}$$

Note that the expression on right-hand side of the above equality is the *ordered Bell number* $a(n)$. Turning our attention now to posterior Bayes vulnerability, we have

$$\mathcal{L}_1^{\times}(\boldsymbol{\vartheta}, \mathsf{I}) \;=\; \frac{V_1[\boldsymbol{\vartheta} \triangleright \mathsf{I}]}{V_1(\boldsymbol{\vartheta})}$$

which implies that

$$V_1[\boldsymbol{\vartheta} \triangleright \mathsf{I}] \;=\; \mathcal{L}_1^{\times}(\boldsymbol{\vartheta}, \mathsf{I}) \cdot V_1(\boldsymbol{\vartheta}) \;=\; \frac{\sum_{i=1}^{\min\{n,k\}} \left\{{n \atop i}\right\} i!}{k^n}\,. \tag{4}$$

since $V_1(\boldsymbol{\vartheta}) = k^{-n}$.

The posterior Bayes vulnerability of Ideal ORE can therefore be calculated as follows:

$$\mathit{Bayes}_I(n, k) = \frac{\sum_{i=1}^{\min\{n,k\}} \left\{{n \atop i}\right\} i!}{k^n} \tag{5}$$

[5] Stirling numbers of the second kind can be computed efficiently thanks to the following recurrence relation for $n \geq i \geq 1$:

$$\left\{{n \atop i}\right\} \;=\; \left\{{n-1 \atop i-1}\right\} + i\left\{{n-1 \atop i}\right\}$$

and initial conditions for $n \geq 1$: $\left\{{0 \atop 0}\right\} = 1$ and $\left\{{n \atop 0}\right\} = 0$ and $\left\{{n \atop 1}\right\} = 1$.

I	$\{1,2,3\}$	$\{3\} < \{1,2\}$	$\{1,2\} < \{3\}$	$\{2\} < \{1,3\}$	$\{1,3\} < \{2\}$	$\{1\} < \{2,3\}$	$\{2,3\} < \{1\}$	$\{3\} < \{2\} < \{1\}$	$\{2\} < \{3\} < \{1\}$	$\{3\} < \{1\} < \{2\}$	$\{1\} < \{3\} < \{2\}$	$\{2\} < \{1\} < \{3\}$	$\{1\} < \{2\} < \{3\}$
(0, 0, 0)	1	0	0	0	0	0	0	0	0	0	0	0	0
(0, 0, 1)	0	0	1	0	0	0	0	0	0	0	0	0	0
(0, 0, 2)	0	0	1	0	0	0	0	0	0	0	0	0	0
(0, 1, 0)	0	0	0	0	1	0	0	0	0	0	0	0	0
(0, 1, 1)	0	0	0	0	0	1	0	0	0	0	0	0	0
(0, 1, 2)	0	0	0	0	0	0	0	0	0	0	0	0	1
(0, 2, 0)	0	0	0	0	1	0	0	0	0	0	0	0	0
(0, 2, 1)	0	0	0	0	0	0	0	0	0	0	1	0	0
(0, 2, 2)	0	0	0	0	0	1	0	0	0	0	0	0	0
(1, 0, 0)	0	0	0	0	0	0	1	0	0	0	0	0	0
(1, 0, 1)	0	0	0	1	0	0	0	0	0	0	0	0	0
(1, 0, 2)	0	0	0	0	0	0	0	0	0	0	0	1	0
(1, 1, 0)	0	1	0	0	0	0	0	0	0	0	0	0	0
(1, 1, 1)	1	0	0	0	0	0	0	0	0	0	0	0	0
(1, 1, 2)	0	0	1	0	0	0	0	0	0	0	0	0	0
(1, 2, 0)	0	0	0	0	0	0	0	0	0	1	0	0	0
(1, 2, 1)	0	0	0	0	1	0	0	0	0	0	0	0	0
(1, 2, 2)	0	0	0	0	0	1	0	0	0	0	0	0	0
(2, 0, 0)	0	0	0	0	0	0	1	0	0	0	0	0	0
(2, 0, 1)	0	0	0	0	0	0	0	0	1	0	0	0	0
(2, 0, 2)	0	0	0	1	0	0	0	0	0	0	0	0	0
(2, 1, 0)	0	0	0	0	0	0	0	1	0	0	0	0	0
(2, 1, 1)	0	0	0	0	0	0	1	0	0	0	0	0	0
(2, 1, 2)	0	0	0	1	0	0	0	0	0	0	0	0	0
(2, 2, 0)	0	1	0	0	0	0	0	0	0	0	0	0	0
(2, 2, 1)	0	1	0	0	0	0	0	0	0	0	0	0	0
(2, 2, 2)	1	0	0	0	0	0	0	0	0	0	0	0	0

TABLE I: Ideal ORE channel matrix I where $n = 3$ and $k = 3$

Now we consider the posterior Bayes vulnerability of CLWW ORE. As in Ideal ORE, encrypting a column X reveals to the adversary that X comprises i distinct values, for some i with $1 \leq i \leq \min\{n, k\}$; moreover, it reveals the placement of those values among the indices $1, 2, \ldots, n$; and finally it reveals the ordering of those values.

If we write those values in increasing order as $v_1, v_2, \ldots, v_i$, then the adversary's only challenge is to determine how those values are chosen from $0, 1, \ldots, k-1$. They can have been chosen in $\binom{k}{i}$ different ways, and in Ideal ORE the adversary has no clue what they are. But CLWW ORE gives the adversary additional information, namely the "report" R of the first differing bit positions of the adjacent values:

$$R \;=\; [v_1 \# v_2, v_2 \# v_3, \ldots v_{i-1} \# v_i]\,.$$

The key question, then, is how many different reports are possible for each i. It is interesting to note that if it happened that distinct sequences of values always produced distinct reports, then there would be $\binom{k}{i}$ possible reports, and leakage would be total.[6] In reality, however, distinct sequences of values often produce the same report. For example, the values $0001, 0111, 1010, 1011$ and $0000, 0100, 1000, 1001$ both produce the same report $R = [3, 4, 1]$. To count the number of possible reports, we define the following function:

Definition 2. *For $1 \leq i \leq k$, let function $r(k, i)$ be the number of possible reports if a sequence of i values is selected from $\{0, 1, 2, \ldots, k-1\}$.*

Given function r, we see that the multiplicative Bayes leakage of CLWW ORE on a uniform prior is given by

$$\mathcal{L}_1^{\times}(\vartheta, \mathsf{C}) \;=\; \sum_{i=1}^{\min\{n,k\}} \left\{ {n \atop i} \right\} i!\, r(k, i)\,. \tag{6}$$

The posterior Bayes vulnerability is therefore

$$\mathit{Bayes}_C(n, k) = \frac{\sum_{i=1}^{\min\{n,k\}} \left\{ {n \atop i} \right\} i!\, r(k, i)}{k^n} \tag{7}$$

But determining the value of $r(k, i)$ is not straightforward.

Certain boundary conditions are immediate: $r(k, 1) = 1$ (since then R is empty) and $r(k, k) = 1$ (since then there is only one possible sequence of values). Also, it is convenient to define $r(k, i) = 0$ if $i > k$, since in that case there are no possible sequences of values. But the general case of $r(k, i)$ seems quite difficult to determine.

For small values of k and i, $r(k, i)$ can be calculated by exhaustive trial, but as k and i grow, this becomes computa-

[6] Since in that case the number of possible outputs is equal to the number of possible secret values, we obtain an interesting mathematical identity for all $n \geq 1$ and $k \geq 1$:

$$\sum_{i=1}^{\min\{n,k\}} \left\{ {n \atop i} \right\} i! \binom{k}{i} \;=\; k^n\,.$$

C	[] {1,2,3}	[1] {3} < {1,2}	[2] {3} < {1,2}	[1] {1,2} < {3}	[2] {1,2} < {3}	[1] {2} < {1,3}	[2] {2} < {1,3}	[1] {1,3} < {2}	[2] {1,3} < {2}	[1] {1} < {2,3}	[2] {1} < {2,3}	[1] {2,3} < {1}	[2] {2,3} < {1}	[1,2] {3} < {2} < {1}	[1,2] {2} < {3} < {1}	[1,2] {3} < {1} < {2}	[1,2] {1} < {3} < {2}	[1,2] {2} < {1} < {3}	[1,2] {1} < {2} < {3}
(0, 0, 0)	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(0, 0, 1)	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(0, 0, 2)	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(0, 1, 0)	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
(0, 1, 1)	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
(0, 1, 2)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
(0, 2, 0)	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
(0, 2, 1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
(0, 2, 2)	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
(1, 0, 0)	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
(1, 0, 1)	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
(1, 0, 2)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
(1, 1, 0)	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(1, 1, 1)	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(1, 1, 2)	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(1, 2, 0)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
(1, 2, 1)	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
(1, 2, 2)	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
(2, 0, 0)	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
(2, 0, 1)	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
(2, 0, 2)	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
(2, 1, 0)	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
(2, 1, 1)	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
(2, 1, 2)	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
(2, 2, 0)	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(2, 2, 1)	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(2, 2, 2)	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TABLE II: CLWW ORE channel matrix C where $n = 3$ and $k = 3$

k \ i	1	2	3	4	5	6	7	8	9	10	11	12
1	1											
2	1	1										
3	1	2	1									
4	1	2	2	1								
5	1	3	4	3	1							
6	1	3	5	5	3	1						
7	1	3	6	8	7	4	1					
8	1	3	6	9	10	8	4	1				
9	1	4	9	15	19	18	12	5	1			
10	1	4	10	18	25	27	22	13	5	1		
11	1	4	11	22	34	42	41	31	17	6	1	
12	1	4	11	23	38	51	56	50	35	18	6	1

TABLE III: $r(k, i)$, the number of possible reports when a sequence of i values is selected from 0, 1, ..., $k - 1$

tionally infeasible. Table III shows the first values of $r(k, i)$ in a 12×12 table.

We have discovered, however, that $r(k, i)$ is governed by a remarkable recurrence relation, which allows it to be computed efficiently.

Theorem 3. *Given $k \geq 2$, let z be the number of 0 bits at the end of the binary representation of $k - 1$. (Equivalently, z is the largest integer such that $k - 1$ is divisible by 2^z.) Then for $1 \leq i \leq k$ we have*

$$r(k, i) = r(k - 1, i) + r(k - 1, i - 1) - r(k - 1 - 2^z, i - 1) .$$

(Note that if $k - 1$ is a power of 2 then $k - 1 - 2^z = 0$, and we define $r(0, i)$ to be 0 for all i.)

The proof and further discussion of reports are in Appendix A.

C. Security Implications

Given a recurrence relation for $r(k, i)$, we can construct graphs that compare the posterior Bayes vulnerability of Ideal and CLWW ORE.

Let us first examine how posterior Bayes vulnerability grows as a function of n with $k = 100$ as depicted in Figure 1. Ideal and CLWW ORE are graphed on the same axis, along with the prior Bayes vulnerability. We see that the posterior Bayes vulnerability of CLWW ORE is consistently higher than the posterior Bayes vulnerability of Ideal ORE.

When $n = 100$, we see that the posterior Bayes vulnerability for Ideal ORE is very small. Although all k values could be represented in the column and leak the secret completely, under a uniform prior this is extremely unlikely so the secret remains secure. As n increases, the difference between Ideal ORE and CLWW ORE grows before converging again: when $n = 300$, under Ideal ORE, the adversary has a very small

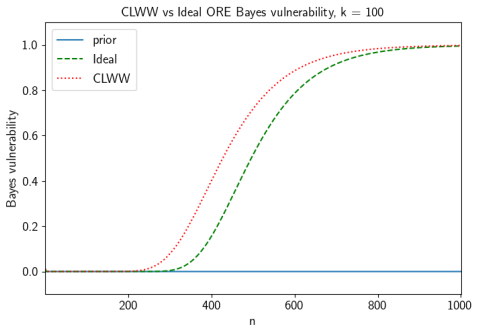

Fig. 1: CLWW & Ideal ORE Bayes vulnerability when $k = 100$ as a function of n.

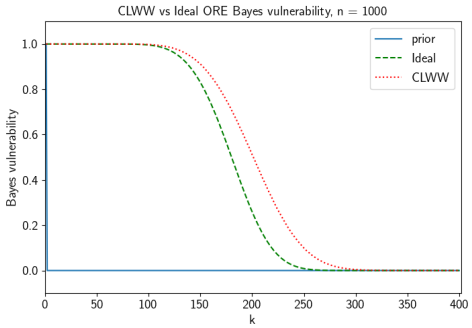

Fig. 2: CLWW & Ideal ORE Bayes vulnerability when $n = 1000$ as a function of k.

chance of guessing the secret correctly at 0.0044, but under CLWW ORE, the probability increases over 10-fold to 0.0783.

As a function of k, Figure 2 shows how the posterior Bayes vulnerabilities of CLWW and Ideal ORE decrease with a column size $n = 1000$. When $k = 1$, the prior and posterior Bayes vulnerability for Ideal and CLWW ORE are at 1: given only one possible value, the adversary will always guess correctly. As the set of secret values increases, we see that the adversary's probability of success gradually decreases. Ideal ORE remains safer than CLWW ORE but they decrease together. Note that around $k = 300$, the adversary is unlikely to guess the column entries, even though the column is not particularly sparse.

Finally let us examine the Bayes vulnerability as n and k grow together shown in Figure 3. As with the previous graph, when the size of the column and the set of plaintext values are both 1, the prior and posterior Bayes vulnerability for both ORE schemes is 1. As the size of the column grows along with the set of plaintext values, the vulnerability begins to decrease. We see that when $n = 20$ and $k = 20$, the posterior Bayes vulnerability is very low for both CLWW and Ideal ORE. This is a surprising result. It would seem that a column of size n with n possible values would be very risky as all n values could be represented and leak the entire column completely. We find, however, that under a uniform prior this is very unlikely to happen and therefore the posterior Bayes vulnerability remains low.

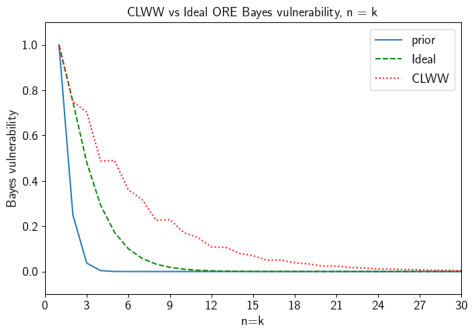

Fig. 3: CLWW & Ideal ORE Bayes vulnerability when $n = k$.

D. Usage Guidelines

One observation that we can make is that if n is much larger than k, then the posterior Bayes vulnerability approaches 1. The reason is that in this case the column is likely to include all k of the possible values, leaking the ordering among entries, and allowing the adversary to determine the entire column exactly. On the other hand, when $k \geq n$ (meaning intuitively that the column is not too dense) then we find empirically that posterior Bayes vulnerability seems usually to be "small".

Remarkably, we can make this observation precise and rigorous, thereby establishing simple usage guidelines for Ideal ORE. We know from (3) that if $k \geq n$ then the multiplicative Bayes leakage is the ordered Bell number $a(n)$; remarkably, this number has an asymptotic approximation. Defining

$$f(n) := \frac{n!}{2(\ln 2)^{n+1}} \quad (n = 1, 2, 3, \ldots)$$

we have $a(n) \sim f(n)$, which means that the relative error goes to 0 as n goes to infinity. More precisely, it follows from Theorem 1 in [32], that :

$$\left| \frac{a(n)}{f(n)} - 1 \right| \leq 2\,\zeta(n+1) \left(\frac{\log 2}{2\pi} \right)^{n+1}$$

where $\zeta(m)$ is the Riemann zeta function that, for m natural number greater than or equal to 2, is sandwiched between 1.7 and 1, it is monotonically decreasing and it approximates 1 rather quickly. Hence $\left|{}^{a(n)}/_{f(n)} - 1\right|$ goes to 0 exponentially fast as n goes to infinity.

Since the posterior Bayes vulnerability on a uniform prior is the multiplicative Bayes leakage times the prior Bayes vulnerability, which is k^{-n}, then when $k = n$ the posterior Bayes vulnerability is approximated by $p(n)$, defined by

$$p(n) := \frac{f(n)}{n^n}$$

(Note furthermore that the posterior Bayes vulnerability will be smaller if $k > n$.) As n increases, $p(n)$ decreases rather quickly, and indeed we have the following theorem.

Theorem 4. *$p(n)$ is monotonic decreasing. Indeed, for all $n \geq 1$, we have $p(n) > 1.386 \cdot p(n+1)$.*

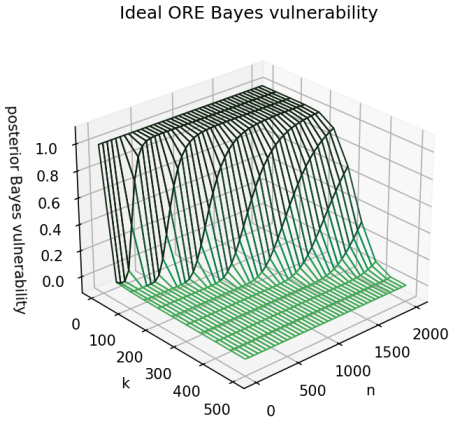

Fig. 4: Posterior Bayes vulnerability as n ranges from 1 to 2000 and k ranges from 1 to 500.

The proof is given in Appendix A.

Given this theorem, we can establish useful usage guidelines in the form of upper bounds on posterior Bayes vulnerability for Ideal ORE under a uniform prior. For instance, we have the following theorem.

Theorem 5. *If $k \geq n \geq 1$, then we have*

$$\mathit{Bayes}_I(n,k) \leq (3/4)^{n-1} \cdot (n/k)^n \ .$$

The proof is given in Appendix A. As an illustration, if $n = 5$ and $k = 12$, then

$$\mathit{Bayes}_I(n,k) < (3/4)^4 \cdot (5/12)^5 < 0.004 \ .$$

Since $k = 12$ is greater than $n = 5$, the posterior Bayes vulnerability is very small, at less than 0.004. (The precise vulnerability is 0.00217).

Fundamentally, this usage guideline means that a database administrator can safely apply Ideal ORE if $k \geq n$ provided a Bayes threat model and a uniform prior. As shown in Figure 4, provided 500 people ($n = 500$) and at least 500 possible values, Ideal ORE is safe to deploy.

IV. Bucketing vulnerability of Ideal ORE and CLWW ORE

Bayes vulnerability measures an adversary's probability of guessing the entire column X exactly. But an adversary might have the more modest goal of guessing the column's entries only approximately — this is very natural if the entries are salaries, whose exact value to the penny is of no great importance. To this end, suppose that k factors as $b \cdot s$. Then we can divide the values in $\{0, 1, \ldots, k-1\}$ into b *buckets*, each of size s, where bucket 0 contains the values from 0 to $s-1$, bucket 1 contains the values from s to $2s-1$, and so forth. And we can consider an adversary who just wants to guess the correct bucket for each entry.

We hence consider *bucketing vulnerability*, which results from a gain function that requires the adversary to guess a sequence of n buckets, each in the range $0, 1, \ldots, b-1$, giving a gain of 1 if the buckets are all correct for X, and 0 otherwise.

Bucketing makes sense whenever k factors into $b \cdot s$, but its effect is particularly easy to understand in the case where s is a power of 2. For then when entries are written in binary, of length $\ell = \lceil \log_2 k \rceil$, the significance is that the adversary cares only about the first $\lceil \log_2 b \rceil$ bits of each entry, and not about the last $\log_2 s$ bits.

A. Bucketing Vulnerability Computation

As with Bayes vulnerability, naïve calculation is prohibitively expensive. However, there is an efficient way to calculate the posterior bucketing vulnerability of Ideal ORE on a uniform prior.

Theorem 6. *Given column X of size n, with values in $\{0, 1, \ldots, k-1\}$ and $k = b \cdot s$, where b is the number of buckets and s is the size of each bucket, the posterior bucketing vulnerability of Ideal ORE on a uniform prior, which we denote by $\mathit{Bucket}_I(n,k,b,s)$, can be computed as follows:*

$$\mathit{Bucket}_I(n,k,b,s) \ = \ \frac{\sum_{i=1}^{\min\{n,k\}} \left\{ {n \atop i} \right\} i!\, e(i,b,s)}{k^n}$$

where

$$e(i,b,s) = \binom{s}{\lfloor \frac{i}{b} \rfloor + 1}^{i \bmod b} \cdot \binom{s}{\lfloor \frac{i}{b} \rfloor}^{b-(i \bmod b)} .$$

A proof is given in Appendix B.

Turning now to the posterior bucketing vulnerability of CLWW ORE, we restrict our attention to the case when s is a power of 2, since the implications then are so catastrophic. For in this case CLWW ORE allows the adversary to determine the ordering between the *buckets* of any two entries: given the ciphertexts for two ℓ-bit entries v and v', she simply checks whether $v \# v' \leq \log_2 s$. For if so their buckets are equal, and otherwise they are ordered in the same way that v and v' are.[7] It follows that CLWW ORE's posterior bucketing vulnerability when s is a power of 2 can be computed as its posterior *Bayes* vulnerability when entries are restricted to $\{0, 1, \ldots, b-1\}$ (rather than $\{0, 1, \ldots, k-1\}$).

Theorem 7. *If $k = b \cdot s$ and s is a power of 2, then posterior bucketing vulnerability of CLWW ORE on a uniform prior can be computed as follows:*

$$\mathit{Bucket}_C(n,k,b,s) \ = \ \mathit{Bayes}_C(n,b) \, .$$

[7]Note in contrast that Ideal ORE does *not* allow such deductions, since knowing that $v < v'$ does not tell whether their buckets are equal or not.

Also note that CLWW ORE does not allow these deductions when s is not a power of 2. For example, suppose that $k = 6$, $b = 2$, and $s = 3$, so that the two buckets consist of $\{000, 001, 010\}$ and $\{011, 100, 101\}$. If two values v and v' satisfy $v \# v' = 3$, then their buckets might be equal (e.g. 011 and 101) or unequal (e.g. 000 and 100). Similarly, if $v \# v' = 2$, then their buckets might be equal (e.g. 001 and 010) or unequal (e.g. 000 and 011). Similarly, if $v \# v' = 1$, then their buckets might be equal (e.g. 000 and 001) or unequal (e.g. 010 and 011).

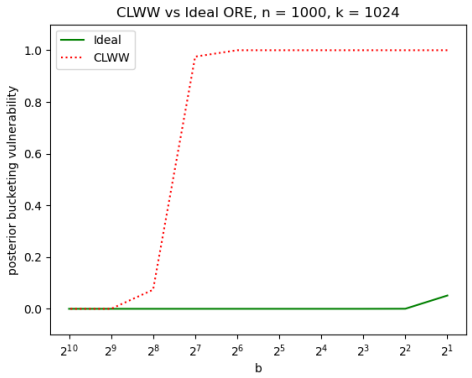

Fig. 5: CLWW and Ideal ORE bucketing vulnerability when n = 1000, k = 1024 as a function of the number of buckets b.

B. Security Implications

The implications of Theorem 7 for the security of CLWW ORE are truly catastrophic. For that theorem shows that whatever the values of n and k may be (assuming that k is a power of 2), a bucketing adversary can in effect make k be as small as she likes (though at the cost of coarser and coarser approximations) by choosing b to be smaller and smaller. Since we know that the case when n is much larger than k is insecure for ORE, this result is terrible for CLWW ORE.

Figure 5 illustrates the vast difference in posterior bucketing vulnerability of Ideal and CLWW ORE. Here we see the adversary's probability of successfully guessing the first bits of every entry in the column when $n = 1000$ and $k = 1024$. Ideal ORE, indicated by the green line, remains fairly secure as the number of buckets decrease, while CLWW ORE degrades dramatically.

When $b = 2^{10}$, the adversary must guess all 10 bits of every value in the column correctly on the first try, corresponding to Bayes vulnerability. Intuitively, this task is very difficult and her probability of success is very low. However, at $b = 2^7$, the red dotted line corresponding to CLWW ORE bucketing vulnerability jumps to 0.9752. The adversary needs to only guess the first 7 bits for CLWW ORE to be unsafe under these conditions. Under a bucketing threat, it is clear the CLWW ORE is catastrophically insecure compared to Ideal ORE.

Figure 6 showing the posterior bucketing vulnerability for different values of b and k, provides further evidence that Ideal ORE is robust against the bucketing scenario, while CLWW ORE is not.

Figures 6(a) and 6(b) show posterior bucketing vulnerability of Ideal and CLWW ORE where $k = 128$ under $b = 8$, $b = 32$, and $b = 128$, reflecting an adversary attempting to guess the first 3 bits, the first 5 bits, and the entire 7 bit secret. In Figure 6(a), we see that Ideal ORE is incredibly robust regardless of the number of buckets; an adversary attempting to guess the first 3 and first 5 bits, represented by the red and blue lines, does not fare much better than an adversary attempting to guess all of the bits. In contrast, Figure 6(b) shows an adversary guessing the first 3 and 5 bits under CLWW ORE has a high probability of success.

This same phenomenon is shown in Figures 6(c) and 6(d) where $k = 1024$ under $b = 16$, $b = 128$, and $b = 1024$, reflecting an adversary interested in guessing the first 4 bits, the first 7 bits, and all 10 bits. In Figure 6(c) on the left, the green line corresponding to Bayes vulnerability indicates that Ideal ORE is safe up to $n = 5000$; all three lines are very close, regardless of the value of b. However in Figure 6(d), the vulnerability corresponding to $b = 16$ and $b = 128$ grows to 1 quickly, indicating that correctly guessing the first 7 out of 10 bits quickly becomes an easy adversarial task.

V. A Mitigation Strategy

An objection could be raised to our "sufficiently sparse" usage guideline: the values of n and k may be seen as parameters that are *given* and not chosen by a database administrator. In response, we have developed a mitigation strategy that can dramatically improve the security of Ideal ORE. A user simply needs to append random bits to the end of each entry before encrypting it with Ideal ORE; this allows him to make k as large as he likes, thereby making the column as sparse as he likes. This mitigation strategy is effective in decreasing posterior vulnerability of Ideal ORE under both the Bayes and bucketing scenarios.

For example, suppose n = 50 and $k = 8$. An adversary will do very well as it is highly likely that she will observe 8 distinct values among the 50 column entries, allowing her to determine the entire column exactly. Notice also that she will observe many equal entries in the column. Now suppose that the user appends 3 random bits to each entry, so that the number of possible values becomes $8 \cdot 2^3 = 64$. Now this adversary certainly fails — she cannot possibly observe 64 distinct values among the 50 column entries! (Notice furthermore that she will now observe very few equal entries in the column.)

This strategy ensures that two unequal plaintexts retain their order while the order of two equal plaintexts is randomized. Given augmented plaintexts encrypted with Ideal ORE, the adversary can only see the ordering on the augmented plaintexts—*not* the ordering on the original plaintexts.

While this mitigation strategy makes ciphertexts larger, its impact on functionality seems relatively small. The random bits can be added and erased by a local proxy, making the padding transparent to the user, and range queries can still be done — we just need to pad the lower bound with 0s and the upper bound with 1s, ensuring that the desired ciphertexts will be retrieved regardless of how they are padded. Still, some queries are prevented by the inability to determine equality; for example, the server can no longer retrieve the most frequent values in the column.

The effect of mitigation on the posterior Bayes vulnerability of Ideal ORE is shown in Figure 7. Suppose that $n = 1000$. Let secrets be 6 bits long, meaning that $k = 64$. As we know, that is a situation where Ideal ORE is insecure. Indeed the posterior Bayes vulnerability is about 0.999991, as represented by the red dotted line. Intuitively, it is highly likely that all

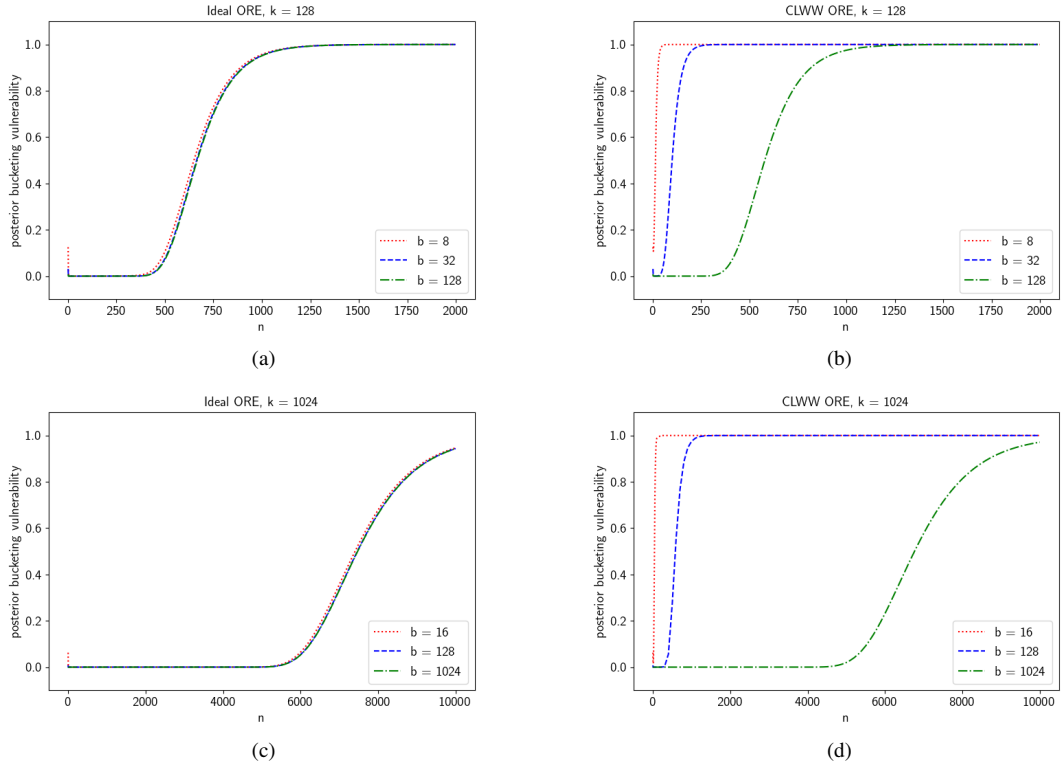

Fig. 6: Posterior bucketing vulnerability under 3 different buckets of (a) Ideal ORE and (b) CLWW ORE with $k = 128$, and (c) Ideal ORE and (d) CLWW ORE with $k = 1024$

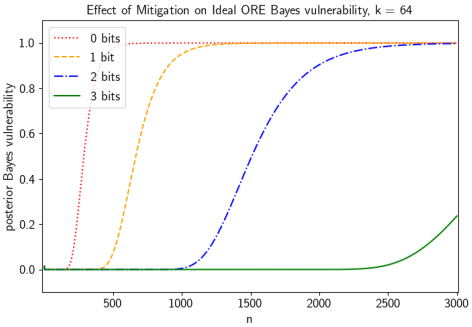

Fig. 7: Ideal ORE Bayes vulnerability when $k = 64$ from 0 to 3 bits of mitigation.

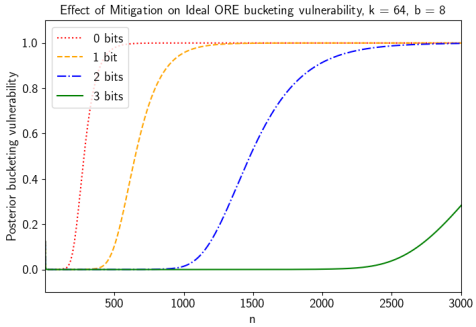

Fig. 8: Ideal ORE bucketing vulnerability when $k = 64$ and $b = 8$ from 0 to 3 bits of mitigation.

64 values occur among the 1000 column entries, allowing the adversary to determine the entire column. With 3 random padding bits, however, the situation is very different. It is now as if secrets were 9 bits long ($k = 512$), but the adversary cares only about the first 6 bits ($b = 64$). Now the posterior Bayes vulnerability is exactly the same as the posterior bucketing vulnerability with $n = 1000$, $k = 512$, and $b = 64$, which is 2.52×10^{-26}, indicated by the solid green line. Intuitively, now very few of the 1000 column entries will be equal, giving the adversary no good way to determine the bucket boundaries.

The effect of mitigation on the posterior bucketing vulnerability of Ideal ORE is shown in Figure 8. Adding bits allows us to increase k while the number of buckets b remains the same. Again, by blurring equality, the adversary is not able to easily determine bucket boundaries and therefore the posterior bucketing vulnerability remains lower as more bits are appended to the plaintext. Every bit added to the plaintext prior to encryption allows the column to be larger without meaningfully sacrificing security.

Unfortunately, the mitigation strategy we propose does nothing to improve either the Bayes or bucketing vulnerability of CLWW ORE. Unlike the Ideal ORE case, the mitigation

does not prevent the adversary from determining equality. We assume that the adversary knows how many bits are appended to the plaintext; if the first significant differing bit between two padded plaintexts falls within the padded bits, then the adversary knows that the corresponding plaintexts are equal.

VI. Future Directions

In this paper, we have analyzed the leakage of Ideal ORE and CLWW ORE, under a uniform prior, in both the Bayes and bucketing scenarios. Future work could (1) analyze the leakage of these channels under different priors or gain functions, or (2) analyze different channels that make different assumptions about what the adversary can observe. Here we give some preliminary thoughts about non-uniform priors, single-index gain functions, and range queries.

A. Non-uniform priors

Thus far this discussion has been restricted to uniform priors. Although the posterior vulnerability under a non-uniform prior can always be computed in principle via Equation (2), this becomes computationally infeasible for large values of n and k, necessitating the development of new techniques. Here we just look at two small examples of the effect of a non-uniform prior. Suppose that $n = 4$ and $k = 11$. Under Ideal ORE and a uniform prior, the posterior Bayes vulnerability is 0.0051. But if the first value (0) has probability $1/2$, while the remaining values (1–10) have probability $1/20$, then the posterior Bayes vulnerability jumps to 0.102. If instead the middle value (5) has probability $1/2$, while the remaining values (0–4, 6–10) have probability $1/20$, then the posterior Bayes vulnerability increases to 0.14025. So for Ideal ORE, posterior Bayes vulnerability depends on *which* values are most likely.

B. Single-index scenarios

In addition to the Bayes and bucketing scenarios, there are other goals that an adversary may have. For example, an adversary could be rewarded for the number of entries guessed correctly, or she might wish to guess the value corresponding to a single index in the column. There are two variations of this latter scenario: 1) the "free" variation in which she is satisfied to guess the value at any one index and 2) the "forced" scenario in which she must guess about one specific index, perhaps corresponding to a high-value person. To illustrate the difference between the "free" and the "forced" scenarios, consider an example with $n = 4$ and $k = 4$.

a) Free scenario: We have to consider various cases, depending on how many different values occur in the column; generally, the adversary's best strategy is to guess about an extreme value, either one that is very large or very small.

If all 4 values occur, which happens with probability $6/64$, then the entries can be matched to the k values: the adversary can guess about any index and her probability of success is 1. If exactly 3 different values occur, which happens with probability $36/64$, then the adversary's best strategy is to either guess that the largest entry is 3 or that the smallest is 0; with either choice, she has a $3/4$ probability of success.

TABLE IV: Summary of the four possible cases when $n = 4$ and $k = 4$ under the free single-index scenario.

Number of values	Prob.	Best guess	Guess prob.
4	$6/64$	Any	1
3	$36/64$	Large = 3, small = 0	$3/4$
2	$21/64$	Large = 3, small = 0	$1/2$
1	$1/64$	Any	$1/4$

By proceeding in this way, for each of the four output categories we obtain the event's probability and the adversary's success probability. These are summarized in Table IV. Overall, the probability to guess correctly the target entry is $6/64 \cdot 1 + 36/64 \cdot 3/4 + 21/64 \cdot 1/2 + 1/64 \cdot 1/4 = 175/256$. This is the posterior free vulnerability.

b) Forced scenario: The vulnerability changes when the adversary is operating under the forced variation. Continuing the example, let the adversary be required to guess about the first entry in the column. The difference w.r.t. the free scenario is only in the case in which 3 different values occur in the column. In such a situation, with probability $2/3$, she must guess about either the largest or smallest entries, giving success probability $3/4$. However, with probability $1/3$, she must guess about the unfavorable middle entry, giving success probability of only $1/2$. Her overall success probability is therefore $163/256$, i.e., $3/64$ less than the free adversary.

C. Range Queries

A quite different direction is to augment the Ideal ORE channel by allowing the adversary to observe the results of a sequence of range queries with randomly-chosen bounds, as in [33] and [34].[8] Such range queries will be likely to reveal information about the entries. For instance, if a range query retrieves one entry but not another, then the two entries are definitely unequal. On the other hand, if two entries are always both retrieved or both not retrieved by a large number of randomly-chosen range queries, then they are likely to be equal. Such leakage is of course uninteresting in the case of Ideal ORE, which already makes equality directly observable. But it appears to severely weaken the effectiveness of the mitigation strategy in Section V, which works by adding random padding to obscure equality among entries.

VII. Conclusion

In this work, we provided a leakage analysis of Ideal and CLWW ORE under two meaningful operational scenarios. We firstly evaluated the Bayes scenario corresponding to an adversary wishing to guess the entire column in one try. We then evaluated the bucketing scenario under which an adversary wishes to correctly guess the range of every value in the column. Through efficient computation techniques using novel combinatorics, we provided security implications for both ORE variations. Under the Bayes scenario, we found that

[8] Those works however assume that (contrary to the case with ORE) the column's ciphertexts themselves leak nothing.

Ideal ORE and CLWW ORE perform comparably, but under the bucketing scenario, the security of CLWW ORE degrades quickly and it is therefore unsafe to use. We proved a simple and surprising usage guideline: Ideal ORE is safe to use when $n \leq k$ under a uniform prior and a Bayes scenario. Finally, we provided a mitigation strategy that improves the security of Ideal ORE under both Bayes and bucketing scenarios.

ACKNOWLEDGEMENTS

We are grateful to the anonymous reviewers for their helpful comments. This work was partially supported by the National Science Foundation under grant CSN-1749014 and by the U.S. Department of Homeland Security under Grant Award Number 2017-ST-062-000002. The work of Catuscia Palamidessi was supported by the European Research Council (ERC) under the Horizon 2020 research and innovation programme, grant agreement № 835294. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the U.S. Department of Homeland Security.

REFERENCES

[1] 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 *Advances in Cryptology - EUROCRYPT 2015*, E. Oswald and M. Fischlin, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 563–594.

[2] A. Boldyreva and P. Grubbs, "White paper: The cloud encryption handbook: Encryption schemes and their relative strengths and weaknesses," McAfee, Santa Clara, California, Tech. Rep., April 2018.

[3] M. Naveed, S. Kamara, and C. V. Wright, "Inference attacks on property-preserving encrypted databases," in *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '15. New York, NY, USA: ACM, 2015, pp. 644–655. [Online]. Available: http://doi.acm.org/10.1145/2810103.2813651

[4] N. Chenette, K. Lewi, S. A. Weis, and D. J. Wu, "Practical order-revealing encryption with limited leakage," in *Fast Software Encryption*, T. Peyrin, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 474–493.

[5] M. S. Alvim, K. Chatzikokolakis, A. McIver, C. Morgan, C. Palamidessi, and G. Smith, *The Science of Quantitative Information Flow*, ser. Information Security and Cryptography. Cham, Switzerland: Springer International Publishing, 2020.

[6] C. Braun, K. Chatzikokolakis, and C. Palamidessi, "Quantitative notions of leakage for one-try attacks," in *Proceedings of the 25th Conf. on Mathematical Foundations of Programming Semantics*, ser. Electronic Notes in Theoretical Computer Science, vol. 249. Elsevier B.V., 2009, pp. 75–91. [Online]. Available: http://hal.archives-ouvertes.fr/inria-00424852/en/

[7] M. S. Alvim, K. Chatzikokolakis, C. Palamidessi, and G. Smith, "Measuring information leakage using generalized gain functions," in *2012 IEEE 25th Computer Security Foundations Symposium*, 2012, pp. 265–279.

[8] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, "Order preserving encryption for numeric data," in *Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '04. New York, NY, USA: ACM, 2004, pp. 563–574. [Online]. Available: http://doi.acm.org/10.1145/1007568.1007632

[9] A. Boldyreva, N. Chenette, Y. Lee, and A. O'Neill, "Order-preserving symmetric encryption," in *Advances in Cryptology - EUROCRYPT 2009*, A. Joux, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 224–241.

[10] D. Liu and S. Wang, "Programmable order-preserving secure index for encrypted database query," in *2012 IEEE Fifth International Conference on Cloud Computing*, 2012, pp. 502–509.

[11] J. Dyer, M. Dyer, and K. Djemame, "Order-preserving encryption using approximate common divisors," *Journal of Information Security and Applications*, vol. 49, p. 102391, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2214212619301103

[12] A. Boldyreva, N. Chenette, and A. O'Neill, "Order-preserving encryption revisited: Improved security analysis and alternative solutions," in *Advances in Cryptology – CRYPTO 2011*, P. Rogaway, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 578–595.

[13] H. Kadhem, T. Amagasa, and H. Kitagawa, "Mv-opes: Multivalued-order preserving encryption scheme: A novel scheme for encrypting integer value to many different values," *IEICE Transactions on Information and Systems*, vol. E93.D, no. 9, pp. 2520–2533, 2010.

[14] F. Kerschbaum and A. Schroepfer, "Optimal average-complexity ideal-security order-preserving encryption," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 275–286. [Online]. Available: https://doi.org/10.1145/2660267.2660277

[15] E. Khoury, M. Medlej, C. A. Jaoude, and C. Guyeux, "Novel order preserving encryption scheme for wireless sensor networks," in *2018 IEEE Middle East and North Africa Communications Conference (MENACOMM)*, 2018, pp. 1–6.

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

[17] K. Lewi and D. J. Wu, "Order-revealing encryption: New constructions, applications, and lower bounds," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 1167–1178. [Online]. Available: https://doi.org/10.1145/2976749.2978376

[18] Z. Liu, X. Chen, J. Yang, C. Jia, and I. You, "New order preserving encryption model for outsourced databases in cloud environments," *Journal of Network and Computer Applications*, vol. 59, pp. 198 – 207, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1084804514001350

[19] R. A. Popa, F. H. Li, and N. Zeldovich, "An ideal-security protocol for order-preserving encoding," in *2013 IEEE Symposium on Security and Privacy*, 2013, pp. 463–477.

[20] I. Teranishi, M. Yung, and T. Malkin, "Order-preserving encryption secure beyond one-wayness," in *Advances in Cryptology – ASIACRYPT 2014*, P. Sarkar and T. Iwata, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 42–61.

[21] V. Bindschaedler, P. Grubbs, D. Cash, T. Ristenpart, and V. Shmatikov, "The tao of inference in privacy-protected databases," *Proc. VLDB Endow.*, vol. 11, no. 11, pp. 1715–1728, July 2018. [Online]. Available: https://doi.org/10.14778/3236187.3236217

[22] D. Cash, P. Grubbs, J. Perry, and T. Ristenpart, "Leakage-abuse attacks against searchable encryption," in *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '15. New York, NY, USA: ACM, 2015, pp. 668–679. [Online]. Available: http://doi.acm.org/10.1145/2810103.2813700

[23] M. S. Islam, M. Kuzu, and M. Kantarcioglu, "Access pattern disclosure on searchable encryption: Ramification, attack and mitigation." in *Network and Distributed System Security Symposium (NDSS'12)*, 2012.

[24] Y. Zhang, J. Katz, and C. Papamanthou, "All your queries are belong to us: The power of file-injection attacks on searchable encryption," in *25th USENIX Security Symposium (USENIX Security 16)*. Austin, TX: USENIX Association, August 2016, pp. 707–720.

[25] F. B. Durak, T. M. DuBuisson, and D. Cash, "What else is revealed by order-revealing encryption?" in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 1155–1166. [Online]. Available: https://doi.org/10.1145/2976749.2978379

[26] P. Grubbs, K. Sekniqi, V. Bindschaedler, M. Naveed, and T. Ristenpart, "Leakage-abuse attacks against order-revealing encryption," in *2017 IEEE Symposium on Security and Privacy (SP)*, 2017, pp. 655–672.

[27] D. Bogatov, G. Kollios, and L. Reyzin, "A comparative evaluation of order-revealing encryption schemes and secure range-query protocols," *Proc. VLDB Endow.*, vol. 12, no. 8, p. 933–947, Apr. 2019. [Online]. Available: https://doi.org/10.14778/3324301.3324309

[28] M. Jurado and G. Smith, "Quantifying information leakage of deterministic encryption," in *Proceedings of the 2019 ACM SIGSAC Conference on Cloud Computing Security Workshop*, ser. CCSW'19. New York, NY, USA: Association for Computing Machinery, 2019, p. 129–139. [Online]. Available: https://doi.org/10.1145/3338466.3358915

[29] M. S. Alvim, K. Chatzikokolakis, A. McIver, C. Morgan, C. Palamidessi, and G. Smith, "Axioms for information leakage," in *2016 IEEE 29th Computer Security Foundations Symposium (CSF)*, 2016, pp. 77–92.

[30] G. Smith, "On the foundations of quantitative information flow," in *Proceedings of the 12th International Conference on Foundations of Software Science and Computation Structures (FOSSACS 2009)*, ser. LNCS, L. de Alfaro, Ed., vol. 5504. York, UK: Springer, 2009, pp. 288–302.

[31] M. Bognar, J. Gibbons, A. McIver, C. Morgan, and T. Schrijvers, *kuifje: A Quantitative Information Flow aware programming language*, 2019, provided by private distribution. [Online]. Available: https://hackage.haskell.org/package/kuifje

[32] A. Sklar, "On the factorization of squarefree integers," *Proceedings of the American Mathematical Society*, vol. 3, no. 5, pp. 701–705, 1952. [Online]. Available: http://www.jstor.org/stable/2032169

[33] G. Kellaris, G. Kollios, K. Nissim, and A. O'Neill, "Generic attacks on secure outsourced databases," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 1329–1340. [Online]. Available: https://doi.org/10.1145/2976749.2978386

[34] M.-S. Lacharité, B. Minaud, and K. G. Paterson, "Improved reconstruction attacks on encrypted data using range query leakage," in *2018 IEEE Symposium on Security and Privacy (SP)*, 2018, pp. 297–314.

APPENDIX

A. Proofs and Discussion from Section III

Theorem 1. *If $a < b < c$, then we have $a\#c = \max\{a\#b, b\#c\}$.*

Proof: Let j be the first bit position at which a, b, and c do not all agree. Because $a < b < c$, their respective bits at position j must either be $0, 0, 1$ or $0, 1, 1$. In the former case we have $a\#c = b\#c = k > a\#b$, and in the latter case we have $a\#c = a\#b = k > b\#c$. In either case we have the desired conclusion. ∎

Definition 2. *For $1 \leq i \leq k$, let function $r(k, i)$ be the number of possible reports if a sequence of i values is selected from $\{0, 1, 2, \ldots, k-1\}$.*

We can firstly observe that all of the reports counted by $r(k-1, i)$ should also be counted by $r(k, i)$. For example, with values 0000, 0001, 0010 ($k = 3$) and all three values represented ($i = 3$), there is only one report $R = [1, 2]$. But if we increase $k = 4$ such that the values can now include 0011, an additional report is possible: $R = [2, 1]$.

A manual examination of the reports themselves provides insight into the general recurrence relation of $r(k, i)$. Table V enumerates all of the reports R that are *newly* introduced at each k: if we let $R(k, i)$ denote the *set* of reports possible for i values in the range $0, 1, \ldots, k-1$, then the (k, i) entry of Table V is $R(k, i) - R(k-1, i)$. Because we wish to examine the additional reports that become possible as k increases, we count $[1, 2]$ as the single new report for $k = 3, i = 3$ and [2, 1] as the report for $k = 4, i = 3$.

Studying Table V is valuable for trying to understand the how the set of possible reports varies with k and i. For instance, the $(10, 4)$ entry of Table III tells us that $r(10, 4) = 18$, and Table V displays what those 18 reports are: we start at the $(10, 4)$ entry (with 3 reports) and continue up column 4, finding additional reports (6, 1, 3, 2, 2, and 1), for a total indeed of 18 reports.

More interestingly, Table V gives clues as to how the reports themselves are constructed. Looking for example at the new reports shown in row 12, we can observe that each new report is formed by appending a 1 to the end of one of the reports in the previous column of row 11. The behavior at row 11 is different, however: there the new reports are formed by appending a 2 to the end of one of the reports in the previous column of row 10 or row 9. And the behavior at row 13 is still different: there the new reports are formed by appending a 3 to the end of one of the reports in the previous column of rows 12, 11, 10, or 9.

But there is a single pattern that is consistent with all of these. If we let b be the number of 0 bits at the end of the binary representation of $k-1$, then the new reports at row k are formed by appending $b+1$ to the end of one of the reports in the previous column of any of the 2^b preceding rows. (Note that 12, 11, and 13 correspond to cases where b is 0, 1, and 2, respectively.) We prove this pattern in Theorem 2, which, crucially, gives a recurrence that allows $r(k, i)$ to be computed efficiently.

Theorem 2. *Given $k \geq 2$, let z be the number of 0 bits at the end of the binary representation of $k-1$. (Equivalently, z is the largest integer such that $k-1$ is divisible by 2^z.) Then for $1 \leq i \leq k$ we have*

$$r(k, i) = r(k-1, i) + r(k-1, i-1) - r(k-1-2^z, i-1)\ .$$

(Note that if $k-1$ is a power of 2 then $k-1-2^z = 0$, and we define $r(0, i)$ to be 0 for all i.)

Proof: Our crucial task is to characterize precisely $R(k, i) - R(k-1, i)$, the set of reports that can be generated by length-i ascending sequences of values in $\{0, 1, \ldots, k-1\}$ but *cannot* be generated by such sequences of values in $\{0, 1, \ldots, k-2\}$. For brevity, we refer to such reports as "new".

A first key fact about the new reports is that they can *only* be generated by sequences that end with $k-1$ and whose penultimate value p satisfies $k-1-2^z \leq p \leq k-2$. Equivalently, if a report R is generated by a sequence S that does not end in this way, then R is not new. To prove this, we first note that if S does not end with $k-1$, then certainly R is not new, since it is in $R(k-1, i)$. To prove the claimed bounds on the penultimate value p, we begin by noting that the binary representation of $k-1$ is $w10^z$ for some bit-string w, while the binary representations of the 2^z claimed possible values for p are given by $w0(0 \cup 1)^z$. So if $p < k-1-2^z$, then its binary representation is in $w'(0 \cup 1)^{z+1}$, where w' represents a smaller number than does w. As a result, $p \ \#\ (k-1) = p \ \#\ (k-1-2^z)$, since both are determined just by comparing w and w'. But this means that the sequence formed by changing the last value in S from $k-1$ to $k-1-2^z$ also generates report R. Hence R is not in fact new.

TABLE V: Reports table

	1	2	3	4	5	6
1	[]					
2		[1]				
3		[2]	[1, 2]			
4			[2, 1]	[1, 2, 1]		
5		[3]	[1, 3] [2, 3]	[2, 1, 3] [1, 2, 3]	[1, 2, 1, 3]	
6			[3, 1]	[1, 3, 1] [2, 3, 1]	[1, 2, 3, 1] [2, 1, 3, 1]	[1, 2, 1, 3, 1]
7			[3, 2]	[2, 3, 2] [3, 1, 2] [1, 3, 2]	[1, 2, 3, 2] [1, 3, 1, 2] [2, 1, 3, 2] [2, 3, 1, 2]	[2, 1, 3, 1, 2] [1, 2, 3, 1, 2] [1, 2, 1, 3, 2]
8				[3, 2, 1]	[1, 3, 2, 1] [2, 3, 2, 1] [3, 1, 2, 1]	[1, 2, 3, 2, 1] [2, 3, 1, 2, 1] [1, 3, 1, 2, 1] [2, 1, 3, 2, 1]
9		[4]	[3, 4] [2, 4] [1, 4]	[2, 1, 4] [3, 1, 4] [1, 2, 4] [1, 3, 4] [2, 3, 4] [3, 2, 4]	[2, 1, 3, 4] [2, 3, 1, 4] [1, 3, 2, 4] [1, 2, 3, 4] [2, 3, 2, 4] [2, 1, 4, 4] [1, 3, 1, 4] [3, 1, 2, 4] [3, 2, 1, 4]	[3, 1, 2, 1, 4] [1, 2, 3, 1, 4] [2, 1, 3, 2, 4] [1, 2, 1, 3, 4] [2, 3, 1, 2, 4] [2, 1, 3, 1, 4] [2, 3, 2, 1, 4] [1, 2, 3, 2, 4] [1, 3, 2, 1, 4] [1, 3, 1, 2, 4]
10			[4, 1]	[2, 4, 1] [1, 4, 1] [3, 4, 1]	[1, 2, 4, 1] [2, 1, 4, 1] [3, 1, 4, 1] [3, 2, 4, 1] [1, 3, 4, 1] [2, 3, 4, 1]	[2, 1, 3, 4, 1] [1, 2, 1, 4, 1] [3, 1, 2, 4, 1] [3, 2, 1, 4, 1] [1, 2, 3, 4, 1] [2, 3, 1, 4, 1] [1, 3, 1, 4, 1] [1, 3, 2, 4, 1] [2, 3, 2, 4, 1]
11			[4, 2]	[4, 1, 2] [3, 4, 2] [1, 4, 2] [2, 4, 2]	[3, 1, 4, 2] [3, 4, 1, 2] [1, 4, 1, 2] [1, 2, 4, 2] [1, 3, 4, 2] [2, 1, 4, 2] [2, 4, 1, 2] [3, 2, 4, 2] [2, 3, 4, 2]	[3, 1, 4, 1, 2] [2, 1, 4, 1, 2] [2, 3, 1, 4, 2] [2, 3, 2, 4, 2] [2, 3, 4, 1, 2] [1, 2, 4, 1, 2] [1, 2, 3, 4, 2] [1, 2, 1, 4, 2] [1, 3, 4, 1, 2] [3, 2, 1, 4, 2] [1, 3, 2, 4, 2] [1, 3, 1, 4, 2] [2, 1, 3, 4, 2] [3, 2, 4, 1, 2] [3, 1, 2, 4, 2]
12				[4, 2, 1]	[4, 1, 2, 1] [2, 4, 2, 1] [1, 4, 2, 1] [3, 4, 2, 1]	[3, 1, 4, 2, 1] [3, 4, 1, 2, 1] [3, 2, 4, 2, 1] [1, 4, 1, 2, 1] [2, 1, 4, 2, 1] [1, 2, 4, 2, 1] [2, 4, 1, 2, 1] [1, 3, 4, 2, 1] [2, 3, 4, 2, 1]
13			[4, 3]	[4, 1, 3] [3, 4, 3] [2, 4, 3] [4, 2, 3] [1, 4, 3]	[4, 1, 2, 3] [3, 2, 4, 3] [1, 3, 4, 3] [2, 3, 4, 3] [3, 4, 2, 3] [4, 2, 1, 3] [3, 4, 1, 3] [1, 4, 2, 3] [1, 2, 4, 3] [2, 1, 4, 3] [2, 4, 2, 3] [3, 1, 4, 3] [1, 4, 1, 3] [2, 4, 1, 3]	[1, 2, 3, 4, 3] [1, 4, 1, 2, 3] [1, 4, 2, 1, 3] [1, 2, 4, 2, 3] [1, 2, 1, 4, 3] [1, 2, 4, 1, 3] [3, 2, 1, 4, 3] [3, 2, 4, 1, 3] [4, 1, 2, 1, 3] [1, 3, 1, 4, 3] [1, 3, 4, 1, 3] [2, 4, 1, 2, 3] [2, 4, 2, 1, 3] [3, 1, 4, 1, 3] [2, 3, 2, 4, 3] [1, 3, 2, 4, 3] [1, 3, 4, 2, 3] [2, 3, 4, 2, 3] [3, 1, 2, 4, 3] [3, 1, 4, 2, 3] [2, 1, 4, 1, 3] [2, 3, 1, 4, 3] [2, 3, 4, 1, 3] [3, 2, 4, 2, 3] [2, 1, 3, 4, 3] [3, 4, 1, 2, 3] [3, 4, 2, 1, 3] [2, 1, 4, 2, 3]
14				[4, 3, 1]	[4, 1, 3, 1] [3, 4, 3, 1] [1, 4, 3, 1] [2, 4, 3, 1] [4, 2, 3, 1]	[2, 1, 4, 3, 1] [1, 2, 4, 3, 1] [2, 4, 1, 3, 1] [1, 3, 4, 3, 1] [2, 4, 2, 3, 1] [3, 4, 2, 3, 1] [3, 1, 4, 3, 1] [4, 2, 1, 3, 1] [2, 3, 4, 3, 1] [3, 4, 1, 3, 1] [1, 4, 2, 3, 1] [1, 4, 1, 3, 1] [3, 2, 4, 3, 1] [4, 1, 2, 3, 1]
15				[4, 3, 2]	[4, 2, 3, 2] [3, 4, 3, 2] [1, 4, 3, 2] [4, 1, 3, 2] [2, 4, 3, 2] [4, 3, 1, 2]	[1, 4, 1, 3, 2] [1, 4, 3, 1, 2] [2, 3, 4, 3, 2] [3, 4, 3, 1, 2] [4, 2, 1, 3, 2] [3, 2, 4, 3, 2] [4, 1, 2, 3, 2] [1, 4, 2, 3, 2] [2, 4, 3, 1, 2] [2, 4, 1, 3, 2] [4, 2, 3, 1, 2] [1, 3, 4, 3, 2] [2, 4, 2, 3, 2] [3, 4, 2, 3, 2] [3, 4, 1, 3, 2] [3, 1, 4, 3, 2] [2, 1, 4, 3, 2] [4, 1, 3, 1, 2] [1, 2, 4, 3, 2]
16					[4, 3, 2, 1]	[2, 4, 3, 2, 1] [4, 3, 1, 2, 1] [4, 2, 3, 2, 1] [1, 4, 3, 2, 1] [4, 1, 3, 2, 1] [3, 4, 3, 2, 1]

At this point we know that all new reports are generated by sequences that end with $p, k-1$, where $k-1-2^z \leq p \leq k-2$. (Note that this implies that all new reports must end with $z+1$.) But are all reports generated by such sequences new? Not necessarily. With $k = 8$ and $i = 3$, the sequence 4, 6, 7 (in binary, 100, 110, 111) generates report $[2, 1]$. But this report is not new, since it is also generated by 0, 2, 3 (in binary, 000, 010, 011).

The precise condition that is needed is that the report R is formed by appending $z+1$ to the end of a sub-report R', where R' is in

$$R(k-1, i-1) - R(k-1-2^z, i-1) \ .$$

That is, R' can *only* be generated by sequences that end with a value p satisfying $k-1-2^z \leq p \leq k-2$. (Note that this condition fails in the counterexample above, since report $[2]$ is generated by 0, 2.) We claim that any R satisfying this condition must be new. For if not, then R can be generated by a sequence S whose final value is less than $k-1$, which we recall has the form $w10^z$ for some bit-string w. But since report R ends with $z+1$, we know that the last two values of S must then have binary representations of the form $w'0(0 \cup 1)^b$ and $w'1(0 \cup 1)^z$, respectively, where w' represents a smaller number than does w. It follows that the penultimate value of S is less than $k-1-2^z$. Hence the sub-report R' (formed by deleting the final 1 from R) is generated by a sequence S' (formed by deleting the final value of S) whose final value is

less than $k-1-2^z$; hence R' fails the condition.

We have hence proved that the set of new reports is equal to the result of appending $z+1$ to each of the reports in $R(k-1,i-1) - R(k-1-2^z,i-1)$. The size of that set is $r(k-1,i-1)-r(k-1-2^z,i-1)$, since $R(k-1-2^z,i-1) \subseteq R(k-1,i-1)$, and hence our desired recurrence is proved. ∎

Theorem 3. *$p(n)$ is monotonic decreasing. Indeed, for all $n \geq 1$, we have $p(n) > 1.386 \cdot p(n+1)$.*

Proof: We have

$$\begin{aligned}\frac{p(n)}{p(n+1)} &= \frac{\frac{n!}{2n^n(\ln 2)^{n+1}}}{\frac{(n+1)!}{2(n+1)^{n+1}(\ln 2)^{n+2}}} \\ &= \frac{(n+1)^n \ln 2}{n^n} \\ &= \left(1+\frac{1}{n}\right)^n \ln 2\end{aligned}$$

Next, using the binomial theorem we have

$$\begin{aligned}\left(1+\frac{1}{n}\right)^n &= \sum_{i=0}^{n} \binom{n}{i} 1^{n-i} \frac{1}{n^i} \\ &\geq \binom{n}{0}\frac{1}{n^0} + \binom{n}{1}\frac{1}{n^1} = 2\end{aligned}$$

(We are using the fact that, since $n \geq 1$, the sum has at least two terms.) Hence the ratio is at least $2\ln 2 > 2 \cdot 0.693 = 1.386$. ∎

Theorem 4. *If $k \geq n \geq 1$, then we have*

$$\mathit{Bayes}_I(n,k) \leq (3/4)^{n-1} \cdot (n/k)^n \ .$$

Proof: We start with the case when $k = n$, in which case the upper bound simplifies to $(3/4)^{n-1}$. To deal with arbitrarily large n, we first observe that by Theorem 3 we have for $n \geq 3$ that $p(n) \leq (3/4)^{n-3} \cdot p(3)$, since the reciprocal of 1.386 is less than $3/4$. Hence $p(n) \leq (3/4)^{n-1}$, since $p(3) = 0.4813440926 < 0.5625 = (3/4)^2$.

It follows that, for all $n \geq 1$, the same bound holds for $V_1[\vartheta \triangleright \mathrm{I}]$. Turning to the case when $k > n$, observe that increasing k to be larger than n does not increase the multiplicative Bayes leakage, by (3). But it does decrease the prior Bayes vulnerability by precisely a factor of $(n/k)^n$. ∎

B. Proofs from Section IV

Theorem 5. *Given column X of size n, with values in $\{0,1,\ldots,k-1\}$ and $k = b \cdot s$, where b is the number of buckets and s is the size of each bucket, the posterior bucketing vulnerability of Ideal ORE on a uniform prior, which we denote by $\mathit{Bucket}_I(n,k,b,s)$, can be computed as follows:*

$$\mathit{Bucket}_I(n,k,b,s) = \frac{\sum_{i=1}^{\min\{n,k\}} \left\{ {n \atop i} \right\} i!\, e(i,b,s)}{k^n}$$

where

$$e(i,b,s) = \binom{s}{\lfloor \frac{i}{b} \rfloor + 1}^{i \bmod b} \cdot \binom{s}{\lfloor \frac{i}{b} \rfloor}^{b-(i \bmod b)} .$$

Proof: Recall that the Ideal ORE channel reveals to the adversary that the column X comprises i distinct values, for some i with $0 \leq i \leq \min\{n,k\}$, and moreover reveals the ordering of those values and their placement among the n column indices. Hence the adversary's only uncertainty concerns the choice of the i values from $\{0,1,\ldots,k-1\}$. In the case of a uniform prior (as we are assuming here) all $\binom{k}{i}$ such choices are equally likely.

But a bucketing adversary is not concerned with the sequence $v_1, v_2, \ldots, v_i$ of exact values, but only with their buckets $\lfloor \frac{v_1}{s} \rfloor$, $\lfloor \frac{v_2}{s} \rfloor$, …, $\lfloor \frac{v_i}{s} \rfloor$. The key question is which such sequences of buckets are most likely. This in turn depends on the question of in how many ways can a sequence of buckets $b_1, b_2, \ldots, b_i$ (each in $\{0,1,\ldots,b-1\}$) be extended to a sequence of full values (each in $\{0,1,\ldots,k-1\}$). Our first thought may be that each bucket b_j can be extended in s possible ways, namely to $s \cdot b_j + s_j$ for some s_j with $0 \leq s_j \leq s-1$. But this is correct only if the buckets are all distinct, since otherwise the resulting sequence of values will not necessarily be in strictly increasing order.

An example may be helpful here. Let $k = 16$, $b = 4$, and $s = 4$, so that we are dealing with 4-bit values and an adversary that cares only about their first 2 bits. With $i = 3$, consider the bucket sequence `00`, `10`, `10`. We can indeed extend `00` with $s = 4$ possible 2-bit sequences, but we must extend the two occurrences of `10` with distinct 2-bit sequences that are in increasing order. The number of ways that we can do this is $\binom{s}{2} = 6$. Hence the total number of extensions of this bucket sequence is $4 \cdot 6 = 24$. In contrast, the bucket sequence `00`, `10`, `11` can be extended in $4 \cdot 4 \cdot 4 = 64$ ways, making it a better guess.

The general principle is that a best sequence of buckets needs to be as *balanced* as possible among the b buckets. Suppose for instance that some bucket occurs 3 times in a sequence, while another occurs 5 times. Such a sequence cannot be optimal. For (considering just those two buckets) the number of possible extensions is $\binom{s}{3} \cdot \binom{s}{5}$. But it we change one of the occurrences of the second bucket to the first bucket, so that both have 4 occurrences, then the number of possible extensions becomes $\binom{s}{4} \cdot \binom{s}{4}$, which is larger.

Such reasoning leads to the conclusion that any best sequence of buckets must consist of either $\lfloor \frac{i}{b} \rfloor + 1$ or $\lfloor \frac{i}{b} \rfloor$ occurrences of each bucket: the number of buckets with $\lfloor \frac{i}{b} \rfloor + 1$ occurrences is $i \bmod b$ and the number with $\lfloor \frac{i}{b} \rfloor$ occurrences is $b - (i \bmod b)$. Hence the total number of possible extensions of such a bucket sequence is given by $e(i,b,s)$.

Finally, we see that the number of columns X for which the optimal adversary will guess all the buckets correctly is given by $\sum_{i=1}^{\min\{n,k\}} \left\{ {n \atop i} \right\} i!\, e(i,b,s)$. Since the number of possible columns is k^n, the posterior bucketing vulnerability is as claimed. ∎