The Tao of Inference in Privacy-Protected Databases

Vincent Bindschaedler
UIUC
bindsch2@illinois.edu

Paul Grubbs
Cornell Tech
pag225@cornell.edu

David Cash
Rutgers University
david.cash@cs.rutgers.edu

Thomas Ristenpart
Cornell Tech
ristenpart@cornell.edu

Vitaly Shmatikov
Cornell Tech
shmat@cs.cornell.edu

Abstract

To protect database confidentiality even in the face of full compromise while supporting standard functionality, recent academic proposals and commercial products rely on a mix of encryption schemes. The common recommendation is to apply strong, semantically secure encryption to the "sensitive" columns and protect other columns with property-revealing encryption (PRE) that supports operations such as sorting.

We design, implement, and evaluate a new methodology for inferring data stored in such encrypted databases. The cornerstone is the *multinomial attack*, a new inference technique that is analytically optimal and empirically outperforms prior heuristic attacks against PRE-encrypted data. We also extend the multinomial attack to take advantage of correlations across multiple columns. These improvements recover PRE-encrypted data with sufficient accuracy to then apply machine learning and record linkage methods to infer the values of columns protected by semantically secure encryption or redaction.

We evaluate our methodology on medical, census, and union-membership datasets, showing for the first time how to infer full database records. For PRE-encrypted attributes such as demographics and ZIP codes, our attack outperforms the best prior heuristic by a factor of 16. Unlike any prior technique, we also infer attributes, such as incomes and medical diagnoses, protected by strong encryption. For example, when we infer that a patient in a hospital-discharge dataset has a mental health or substance abuse condition, this prediction is 97% accurate.

1 Introduction

Frequent large-scale breaches [2,14,94,95] compromise millions of sensitive data records, including credit card numbers, purchase transactions, email messages, and health-care records. Several academic proposals [3, 9, 39, 43, 56, 79] and commercial products [13, 51, 66, 69, 76, 86] aim to mitigate the problem with specialized techniques that (1) protect the data even in the event of a whole-system compromise, while (2) enabling standard applications to operate on this data in its original database, whenever possible.

A popular approach is *encrypted databases*, in which a client (or a proxy acting on his behalf) encrypts the data before uploading it to the database. For efficiency, these systems employ various flavors of "property-revealing" encryption (PRE). PRE schemes reveal certain properties of the underlying plaintexts, e.g., equality in the case of deterministic encryption or relative order in the case of order-preserving encryption. Revealing this information enables essential database operations such as searching and sorting to be performed on the encrypted database entries without decrypting them.

Efficiency in encrypted databases comes at a cost. Recent inference attacks against PRE [22, 42, 60, 68] showed that, in some situations, an adversary can recover plaintexts with good accuracy. These inference attacks use public auxiliary information to obtain an estimate of the expected distribution of plaintexts, then target each database column separately.

In response to these attacks, academics and practitioners now suggest (e.g., [81]) increased protection based on "sensitivity analysis": apply PRE to the less sensitive database columns (enabling efficient database operations on these columns) and strong encryption to the more sensitive columns. The database administrator decides which columns warrant stronger protection. From an attacker's perspective, this is equivalent to redacting or segmenting the sensitive columns, a common practice in medical databases [39, 72, 73]

None of the previously proposed inference attacks work against strongly encrypted or redacted data. It may therefore appear that the state-of-the-art privacy protection methods based on "sensitivity analysis" ensure confidentiality of the sensitive database columns. After all, how can an adversary recover the data that is not even in the database?

Our contributions. We develop a new inference framework to demonstrate that even the state-of-the-art protections described above cannot prevent an adversary from inferring full database records in realistic scenarios. Unlike prior heuristics, our framework exploits correlations across database columns and across databases. Our key technical tool is a new, optimal inference attack against PRE-encrypted columns.

Consider a database in which some columns have been encrypted using PRE, while the "sensitive" columns have been strongly encrypted or redacted. At a high level, our inference framework will (1) recover as many PRE-encrypted values as possible, then (2) use the recovered values to infer the redacted columns. Our attacks use public auxiliary data sources that provide the estimates not just of the distributions for individual columns but also of joint distributions, as well as partial records of specific individuals.

Inference attacks on PRE-encrypted data. We start by developing a new inference technique against deterministic and order-preserving encryption, the two PRE schemes that are actually used in practice. Our approach provides a theoretical basis for attacks and greatly outperforms prior inference heuristics against order-preserving encryption [22, 42, 68].

Consider attacking a single PRE-encrypted column. We treat the attacker's task as a Bayesian inference problem whose objective is to identify the most probable plaintext-to-ciphertext mapping function that fits the constraints of the type of encryption (i.e., a permutation for deterministic encryption, an order-preserving injective function for order-preserving encryption), conditioned on the observed leakage from the ciphertexts and on the priors from the auxiliary data. We call this a *multinomial attack* because we model the vector of plaintexts as sampled according to the multinomial distribution. Subject to this assumption, our new attack is optimal: it is a maximum likelihood estimator (MLE) and thus maximizes the expected accuracy of inference.

We show how to efficiently instantiate our MLE for deterministic (DET) and order-preserving (ORE) encryption. For the former, our attack finds the same solution as frequency analysis [60], matching its optimality. For the latter, our attack is optimal, too, obviating the search for better heuristics. Our approach offers additional benefits over prior work, such as the ability to calculate confidence estimates on inferences.

Even an optimal single-column attack does not work well if, for example, the plaintexts of a column are almost uniformly distributed—even if the column is strongly correlated with another, non-uniformly distributed column. To take advantage of such correlations across PRE-encrypted columns, we extend the multinomial attack to multiple columns, departing significantly from the single-column viewpoint of prior work.

To avoid the curse of dimensionality, our multi-column attack proceeds column by column. We start with a column for which the single-column attack should perform well, then pick another, correlated column as the target. To attack it, we use (a) the plaintext-ciphertext mappings recovered for the first column, and (b) the joint distribution, estimated from the auxiliary data, over just these two columns. Having recovered two columns, we proceed to the next target, and so on, following a dependency graph between columns. We explain how to build column dependency graphs that perform well in practice, using empirical estimates of the mutual information between columns (derived from the auxiliary data).

Our multi-column multinomial attack greatly outperforms single-column approaches. For example, in our case study on the ACS dataset (subsection 7.3), it recovers PRE-encrypted demographic attributes 2 times better than our own single-column attack and **16 times better than the best prior technique**. In the case study on the FOP dataset (subsection 7.5), it recovers ZIP codes 7 times better than the single-column attack and **17 times better than the best prior technique**.

Inference attacks on redacted data. After accurately inferring the values of the PRE-encrypted columns using our single- and multi-column multinomial attacks, we proceed to infer the values of "sensitive" columns—which cannot be attacked using prior techniques. We employ two methods that exploit cross-database correlations: record linkage and machine learning.

If the target and (plaintext) auxiliary databases contain records of the same individuals and overlap on some attributes, we can use record linkage. Classic record linkage [67,70,87,88] works on plaintext databases. Instead, we use the attributes recovered by the multinomial attack on the target database as *quasi-identifiers* to join the target and auxiliary databases.

The auxiliary database may provide only statistical information, as opposed to the records of specific individuals from the target database. In this case, we cast the problem of inferring the redacted values as a prediction task where these values are the target and the attributes recovered from the PRE-encrypted columns are the features. The prediction model (a classifier or a regression model) is trained on the auxiliary data.

Case studies. We evaluate our framework on the HCUP National Inpatient Sample (NIS) dataset with 7,000,000 hospital stay records; the U.S. Census American Community Survey (ACS) dataset with 2,500,000 records; and the Fraternal Order of Police (FOP) dataset with the names and addresses of 600,000 police officers. In our experiments, we protect the datasets using appropriate techniques, e.g., diagnoses, incomes, and addresses are all redacted. All attacks are simulated and do not extract any information that is not already public.

For the HCUP-NIS dataset, we recover all PRE-encrypted attributes and then infer the redacted medical diagnosis codes. Accurately inferring the condition of even a single patient is a privacy breach, thus our inference algorithm maximizes accuracy rather than coverage and only predicts the condition of the patients for whom it is most confident. When predicting that a patient has a mental health or substance abuse condition, its accuracy is 97%; for mood disorders, accuracy is 96%. For the ACS dataset, we recover over 90% of PRE-encrypted attributes and infer annual income to within \$8.4K. We also demonstrate that the attack would have succeeded even if the auxiliary data had been obsolete and much smaller. For the FOP dataset, we infer the exact home addresses of approximately 1 out of 13 police officers, and as many as 1 out of 7 in Michigan and Pennsylvania (equivalent to 7,200 officers in Pennsylvania).

2 Database Privacy Protection

We abstractly model a database DB as an $m \times k$ matrix of values $p_j[i]$, for $1 \le i \le m$ and $1 \le j \le k$. The database schema specifies for column j the set P_j to which $p_j[i]$ must belong. For example, P_1 may hold names which are strings up to some length and P_2 may hold salaries which are integers between 0 and 1,000,000.

The data owner (*client*) applies the privacy protections from in Section 2.2 to the data before uploading it to the database server. For each encrypted column, the client chooses an independent secret key and does not share it with the server.

2.1 Threat model

We focus on the common "smash-and-grab" attack that compromises the entire system hosting the database but does not establish a persistent attacker presence. We model this as a passive adversary who obtains a single snapshot of the server's entire disk and memory, including all server-side secrets (and therefore defeating at-rest and full-disk encryption) but not clients' secrets. Recently, Grubbs et al. showed that a single snapshot of an encrypted database contains a wealth of information about past queries [41], thus our threat model is a very conservative underestimate. Real-world compromises may reveal much more than the attacks described in this paper.

The adversary's goal is to recover as many rows of DB as possible. We assume that the adversary has access to *auxiliary data*—informally, public databases that are correlated with the compromised database. In subsequent sections, we will detail our assumptions about the auxiliary data and its quality. For the purposes of this paper, we assume that the adversary does not perform active attacks such as inserting known plaintexts into the database (c.f., [40,42]).

2.2 Privacy protection techniques

A *privacy schema* associates to each column a technique for protecting the confidentiality of the data in that column.

Redaction. Sensitive data elements can be replaced with random or otherwise unique identifiers chosen independently of the plaintext values. For example, *data segmentation*—advocated for electronic health records (EHR) of individual patients [39,72,73]—removes values or replaces them with random values sampled from the same distribution.

Another approach is to use strong encryption that achieves (at least) indistinguishability under chosen-plaintext attack (IND-CPA) security. Each element $p_j[i]$ in the sensitive column j is replaced by $c_j[i] = \mathcal{E}_K(p_j[i])$ for $1 \leq i \leq m$, where key K is not stored at the database server.

In our threat model, strong encryption is equivalent to segmentation because the ciphertext carries no (computationally learnable) information about the plaintext. We refer to both segmented and strongly encrypted columns as "redacted."

Property-revealing encryption. Redaction of columns severely limits the utility of a database. For example, strong encryption restricts the server to only insertion and retrieval queries; most processing must be performed by clients or client proxies. Property-revealing encryption (PRE) is an attractive alternative that intentionally reveals some properties of plaintexts in order to support certain operations on the ciphertexts without the decryption key. The two PRE schemes that are used in practice are deterministic encryption DE and order-revealing encryption ORE.

DE is a form of symmetric encryption that ensures that encrypting the same message twice produces the same ciphertext. Therefore, DE leaks plaintext equality. This enables efficient exact-match search: a client re-encrypts the word and submits the resulting ciphertext as a query to the database. Our treatment of DE covers a wide variety of specific realizations, such as using a block cipher for short messages or a deterministic length-preserving mode for longer messages [45,46]; format-preserving [4,5] and -transforming encryption [24,63,64]; and deterministic authenticated encryption [85].

(Ideal) ORE leaks plaintext equality and plaintext order. We survey ORE schemes in Section 9. Our results apply to most of these schemes, including [6,11,78].

Choosing privacy schema. Selecting the privacy schema for a given database involves delicate decisions that must balance confidentiality goals with database utility. For example, a schema that strongly encrypts all columns will achieve high confidentiality, but the database server will not be able to perform any database operations (beyond insertion and retrieval) without involving clients or client proxies.

In practice, schemas are chosen in a *utility-first* manner: determine the operations for each column that must be performed by the database server, then choose the most secure scheme that still allows the server to perform those operations. For example, Microsoft's Always Encrypted database encryption guidelines [66] recommend "us[ing] deterministic encryption for columns that will be used as search or grouping parameters." This approach is often the only viable way to satisfy legacy constraints and business needs.

Another approach, espoused by some academics [81], is to perform *column sensitivity analysis.* This tasks the security architect with making judgments about which columns are sensitive enough to warrant strong encryption (or, equivalently, segmentation). We are not aware of any framework or guidelines for determining whether a given column is sensitive or not. As we concretely demonstrate in Section 7.2, naive approaches to sensitivity analysis, such as measuring correlations between columns, can easily miss columns that are *not* strongly correlated with the sensitive column yet enable accurate inference of the latter.

We assume that the adversary knows the privacy schema. Even if the database encrypts column labels and randomizes column order (c.f., [79]), discovering the mapping between a known database schema and individual encrypted columns is almost always straightforward [68]. The lengths of the ciphertexts may reveal the column's domain, or else the adversary can find the closest match between the number of distinct values in a column and domain sizes $|P_1|, \ldots, |P_k|$.

3 Overview

We start in Section 4 by developing a new framework, based on Bayesian inference, for attacking individual PRE-encrypted columns. It can be instantiated to attack both DE and ORE. The resulting attacks are optimal in a well-defined sense and, in particular, our attack against ORE-encrypted columns outperforms prior heuristic approaches [42, 68].

Next, in Section 5, we develop the first ever multi-column attack against PRE-encrypted columns. It exploits correlations between columns and outperforms the best known single-column attacks. For example, attacking the state and ZIP code attributes simultaneously recovers ORE-encrypted ZIP codes with 36% accuracy whereas the best prior work [42] only achieved 2% accuracy on the same dataset.

In Section 6, we develop a whole-database inference attack. Unlike all previous heuristics, it targets "sensitive" columns (e.g., medical diagnosis codes) protected by redaction, i.e., strong encryption or segmentation. Our attacks uses record linkage and machine learning techniques to infer the values of the redacted columns from those of the PRE-encrypted columns (recovered by our single- and multi-column attacks) and the cross-column correlations. This breaks the last defense encrypted databases use against inference attacks, namely, "column sensitivity analysis."

4 Inferring Single PRE Columns

In this section, we formalize a new approach to inference attacks against a single column of data encrypted using DE or ORE. The adversary has access to (1) auxiliary information providing (an estimate of) the distribution of plaintext values, and (2) an observed ciphertext vector. In the DE case, the vector leaks only frequency information (which ciphertexts correspond to the same plaintext); in the ORE case, the vector leaks both frequency information and ordering. Our attack thus applies to any ORE scheme that leaks frequency (including at least [11, 58, 78]). We make the most conservative assumption about leakage and treat the ORE scheme as "ideal," but for specific ORE schemes that leak more, we can extend the attack by adapting the techniques of Grubbs et al. [42].

Our attacks against DE and ORE are instances of a more general Bayesian inference framework. We first describe it, then instantiate it for DE and ORE. The resulting attacks are optimal in a well-defined sense and also allow the adversary to compute his confidence in the inferred values.

4.1 General attack framework

We first design a general framework for attacking PRE-encrypted columns via Bayesian inference, then instantiate it for specific PRE schemes. Let ρ be the distribution of plaintext values for the target column. There are m possible plaintext values: $[m] \equiv \{1, 2, \ldots, m\}$. Let ρ_i be the probability (or expected proportion) of the i^{th} value; $\sum_{i=1}^{m} \rho_i = 1$. The ciphertext vector $\vec{c}$ contains n ciphertexts, each of which is a valid encryption of a plaintext in $[m]$. Encryption is a bijection, thus there are at most m distinct ciphertext values. For simplicity we assume there are exactly m distinct ciphertexts in the sample; the general case is a simple extension. Without loss of generality, relabel them to $[m]$, then the vector $\vec{c}$ can be regarded as a histogram over $[m]$. Let c_j be the number of times ciphertext $j \in [m]$ appears in $\vec{c}$.

Let $f : [m] \to [m]$ denote a bijective plaintext-ciphertext mapping, $\mathbf{f}$ the random variable describing the choice of f made by the PRE key generation algorithm, and F the support of $\mathbf{f}$, i.e., all possible functions. We assume that the attacker knows how f is chosen. The goal of the attack is to compute

$$\mathbb{P}\mathrm{r}\{\mathbf{f}=f \mid \vec{c}\ ; \rho\} = \frac{\mathbb{P}\mathrm{r}\{\vec{c} \mid \mathbf{f}=f\ ; \rho\} \cdot \mathbb{P}\mathrm{r}\{\mathbf{f}=f\ ; \rho\}}{\mathbb{P}\mathrm{r}\{\vec{c}\ ; \rho\}} \tag{1}$$

which is simply the posterior probability $\mathbf{f}$ given $\vec{c}, \rho$. Directly computing $\mathbb{P}\mathrm{r}\{\vec{c}\ ; \rho\}$ is infeasible in general: it scales with $|F|$, which is exponential in m. The attack will instead target just the most likely decryption of $\vec{c}$ by computing

$$f_{\max} = \arg\max_{f}\ \ \mathbb{P}\mathrm{r}\{\mathbf{f} \mid \vec{c}\ ; \rho\} = \arg\max_{f}\ \ \mathbb{P}\mathrm{r}\{\vec{c} \mid \mathbf{f}=f\ ; \rho\} \cdot \mathbb{P}\mathrm{r}\{\mathbf{f}=f\ ; \rho\} \tag{2}$$

and then applying $f_{\max}^{-1}$ to each ciphertext in $\vec{c}$. The second equality uses Bayes' rule and then drops the denominator, which cannot affect the maximization. Equation 2 is the maximum likelihood estimator (MLE) for $\mathbf{f}$ if $\mathbb{P}\mathrm{r}\{\mathbf{f}=f\ ; \rho\}$ is uniform. The actual distribution will be indistinguishable from uniform if the PRE scheme is indistinguishable from a random permutation or random order-preserving function, (q.v., [6, 38]).

Thus Equation 2 simplifies to $\arg\max_f \ \mathbb{P}\mathrm{r}\{\vec{c} \mid \mathbf{f} = f \ ; \rho\}$ and with all but negligible probability, the maximized value is equal to the true $f_{\max}$.

To estimate the likelihood, we model it as the density of a multinomial distribution, giving

$$\mathbb{P}\mathrm{r}\{\vec{c} \mid \mathbf{f} = f \ ; \rho\} = \mathbb{P}\mathrm{r}\{c_1, c_2, \ldots, c_n \mid \mathbf{f} = f \ ; \rho\} = K_c \prod_{i=1}^{m} \rho_i^{c_{f(i)}} \tag{3}$$

where K_c is a normalization constant that depends on $\vec{c}$ but not on f. Thus, it is not required to directly compute the posterior.

Subject to our assumption that data is aptly modeled by the multinomial distribution, the above approach is optimal: the computed $f_{\max}$ maximizes the expected recovery rate.

A benefit of our Bayesian approach is that it allows the attacker to compute confidence values on the plaintext-ciphertext mappings. This is significant because low-confidence recovered values are unlikely to be useful in practice. We explain in Section 4.5 how to compute confidence values.

The general framework needs to be instantiated to attack specific encryption schemes. This involves fixing a particular random variable $\mathbf{f}$ and associated support F and specifying an efficient way to compute Equation 3.

4.2 Attacking deterministic encryption

If the encryption scheme is DE, F is the set of all permutations on $[m]$. When m is large, evaluating Equation 3 for all f is infeasible, since there are $m!$ such functions. However, if we take the log of Equation 3, we get $\sum_{i=1}^{m} c_{f(i)} \log(\rho_i)$, i.e., the log of the likelihood of ciphertext j being an encryption of plaintext i is $c_j \log(\rho_i)$. Thus to find f that maximizes Equation 1 for DE, we solve a linear assignment problem. The solution is the maximum-weight bipartite matching of the induced graph where the weight on edge (i, j) is $c_j \log(\rho_i)$.

Interestingly, other edge-weighting schemes used in prior attacks on DE do *not*, in general, maximize the likelihood. One such alternative scheme was used in the ℓ_p attack of [68]. It weighs edge (i, j) with the ℓ_p-distance between the empirical probability of ciphertext j and ρ_i. In Appendix B, we give a concrete counterexample for the case where $p = 1$.

Lacharité and Paterson [60] showed that frequency analysis is also a maximum-likelihood estimator for the permutation f. In the single-column case, the multinomial attack and frequency analysis are equivalent for recovering DE-encrypted values (they find the same f). However, frequency analysis cannot incorporate information about other correlated values, so when such information is available our approach is superior.

4.3 Attacking order-revealing encryption

If the encryption scheme is ORE, F is the set of order-preserving injective functions. As above, we find the MLE f by computing a maximum-weight matching in the bipartite graph where the weight of edge (i, j) is $c_j \log(\rho_i)$. However, we only consider candidate matchings that correspond to order-preserving injective functions.[1]

Grubbs et al. [42] point out that such matchings are "non-crossing": if the matching contains edge (i, j), it cannot contain any edge (α, β) where $\alpha < i$ but $\beta > j$. We use their maximum-weight non-crossing matching algorithm to return a candidate f in $\mathcal{O}(m^2)$ time. However, Grubbs et al. use the L1 distance as a heuristic to weight the edges of the bipartite graph. In our case, log-likelihood edge weights ensure that our attack actually maximizes the likelihood function, thus, in contrast to [42], our attack is optimal. This is not merely a theoretical advantage. In Sections 4.4 and Section 7, we demonstrate that our attack significantly outperforms [42] on realistic datasets.

4.4 Comparison with prior heuristics

We now empirically demonstrate that our multinomial attack is not only theoretically optimal, but also outperforms the best prior heuristic, the non-crossing attack of Grubbs et al. [42], which in turn outperforms [68].

[1]Note a subtlety here: if an encryption of every possible plaintext appears in $\vec{c}$, only one such function exists. This was not the case for any dataset in our case studies.

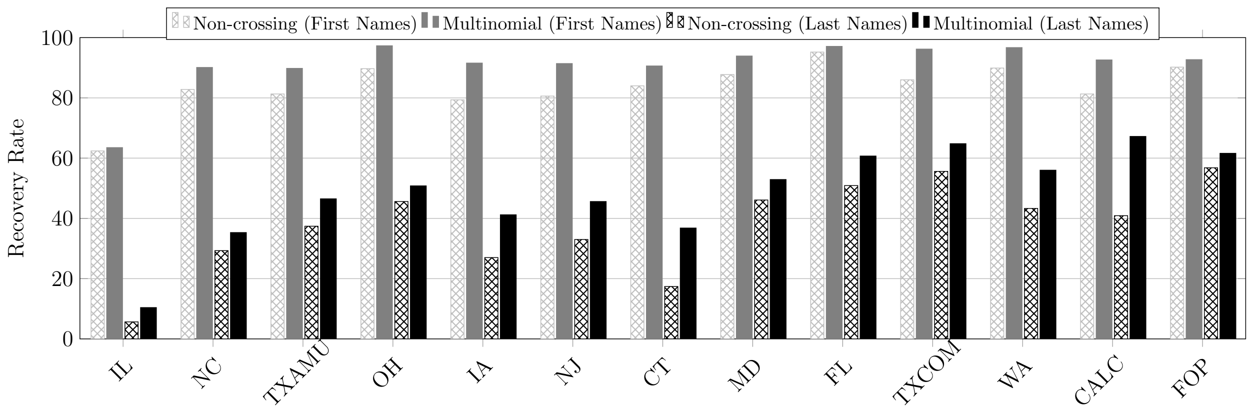

Figure 1: Recovery rates of the non-crossing attack [42] and our multinomial attack on first and last names, for various datasets.

We use the same datasets as [42]—including publicly available datasets of U.S. federal state and government employees and the FOP dataset described in subsection 7.1—to simulate attacks against ORE-encrypted first and last names, a common type of data to which ORE is applied in practice. As in prior work, we use the first and last names from these datasets. The smallest dataset contains about 1,000 records, the largest over 250,000.

Figure 1 shows the recovery rates, defined as the fraction of records in each dataset for which the exact value of the attribute (first or last name) is recovered. Our multinomial attack performs strictly better than the non-crossing attack. On some datasets, it improves first name recovery by 10%. The improvement in last name recovery is more substantial: on some datasets, our attack performs twice as well.

The multinomial attack is also substantially better than the non-crossing attack in recovering low-frequency plaintexts. For example, the average *unique* recovery rate (the fraction of distinct elements recovered without multiplicity) for last names was only 8% for the non-crossing attack, but 14% (almost twice as good) for the multinomial attack. The best unique recovery rate of the multinomial attack for any dataset was 20% vs. 7% for the non-crossing attack on the same dataset.

In section 7, we will demonstrate that on large, realistic datasets, our single- and multi-column multinomial attack outperforms the best prior heuristic by up to 16 times.

4.5 Measuring confidence and uncertainty

Another advantage of our framework is that it can provide confidence values for the attack. Consider an attacker who performs an inference attack on a PRE-encrypted column. The attack produces some plaintext-to-ciphertext mapping but the attacker does not know whether this mapping is correct or not. Prior techniques provide no help in answering this question.

There are two causes for failing to find the correct mapping: (1) the most probable mapping is not the correct one, or (2) the auxiliary data is wrong and does not reflect the ciphertext distribution. First, consider case (1). The multinomial attack returns a mapping $f_{\max}$ that maximizes the posterior (Equation 1). The posterior value of $f_{\max}$, which we denote as $p(f_{\max})$, reflects the probability that it is the correct mapping, as opposed to some other f'. The posterior value itself can be used as a confidence value: the closer $p(f_{\max})$ is to 1, the more likely it is the correct mapping.

Therefore, the attacker needs to compute $p(f_{\max})$, but computing the full posterior is often intractable (Section 4). Instead, we propose two approximation strategies.

Approximation via top K mappings. Given another probable mapping f', $\frac{p(f_{\max})}{p(f')}$ quantifies how much more likely $f_{\max}$ is than f'. This ratio can be computed without fully computing the posterior distribution, enabling a simple approximation of $p(f_{\max})$.

Concretely, we compute the top K mappings (for some small integer $K > 1$) using the appropriate graph algorithm. For example, for DE, we can use algorithms such as [10] or [37]. Informally, we first find the best matching using the Hungarian algorithm, then find additional mappings iteratively by partitioning the solution space.

After computing the top K mappings, we obtain the mapping weights $w_1 \geq w_2 \geq \ldots \geq w_K$, where w_1 is the weight of $f_{\max}$. Given that each weight is the log of the likelihood, we can compute the posterior ratio of $f_{\max}$ and the i^{th} best mapping as $e^{w_1 - w_i}$, assuming a flat prior is used. We can then estimate $p(f_{\max})$ as: $e^{w_1} \cdot [\sum_{i=1}^{K} e^{w_i}]^{-1}$.

Approximation via the permanent. The permanent of an $n \times n$ matrix M is defined as

$$\sum_{\sigma \in \mathbf{Perms}(n)} \prod_{i=1}^{n} M_{i,\sigma(i)}$$

where $\mathbf{Perms}(n)$ is the set of all permutations of n items. It is known that if the matrix M is the adjacency matrix of a weighted bipartite graph, the permanent of M is the sum of the weights of all possible perfect matchings in the graph. Since the set of possible permutations corresponds 1-to-1 to the set of perfect matchings in the bipartite graph where each edge is weighted with the (log-)likelihood, approximating the permanent of the graph's adjacency matrix gives (up to a constant factor) an approximation of the denominator of the MLE of f for DE.

There are several Monte Carlo approximation algorithms for the permanent [44, 53]. For the multinomial attack on ORE, not all perfect matchings correspond to the candidate functions in F—$f \in F$ are exactly the *non-crossing* matchings. We are not aware of any function of a bipartite graph's adjacency matrix that sums only the non-crossing matchings, but an exact or approximate algorithm for such a function would give a way to compute the confidence values for ORE attacks. Extending the algorithms of [44, 53] to give a Monte Carlo approximation for the posterior denominator in the ORE case is an interesting open problem for future work.

Detecting bad auxiliary data. If an inference was based on bad auxiliary data, we cannot rely on $p(f_{\max})$, but we can detect this through a hypothesis test.

The null hypothesis is that the distribution of the ciphertexts really comes from a multinomial distribution with parameters given by the auxiliary data. This is a multinomial test [83], but there are two difficulties. First, we do not know the correct mapping, but we can simply use $f_{\max}$. Second, using an exact multinomial test is computationally infeasible in practice when n, the number of observed ciphertexts, is large.

Fortunately, we can approximate the multinomial test with a χ^2-test. Specifically, under the null hypothesis, the expected number of plaintexts having the i^{th} value is $n\rho_i$, and given $f_{\max}$, the corresponding number of observations is $c_{f_{\max}(i)}$. Thus, the test statistic is $\sum_{i=1}^{m} \frac{(n\rho_i - c_{f_{\max}(i)})^2}{n\rho_i}$, where the corresponding p-value is computed using the χ^2-CDF with $m-1$ degrees of freedom.

An example of computing confidence values. To demonstrate the importance of confidence values, we perform the following experiment. We take the ACS dataset, specifically the `SEX` column which has two values: male (value 1) or female (value 2). Using the 2012 dataset as the auxiliary data, the probability distribution for this column is 0.485, 0.515. If this column is encrypted by DE and we denote the two corresponding ciphertexts as a and b, then the attacker has to choose between two mappings: (1) $a \leftrightarrow$ male, $b \leftrightarrow$ female, and (2) $a \leftrightarrow$ female, $b \leftrightarrow$ male.

We take the 2013 ACS dataset as the target and calculate confidence values, which depend on the number and distribution of ciphertexts and the auxiliary information. Concretely, we vary the number n of observed ciphertext and calculate the confidence value in each case. Figure 2 shows the results for $n = 10, 25, 50, 75, 100, 500, 1000, 5000$. The y-axis corresponds to the confidence values. The correct mapping is (2), shown in (light) yellow in the figure. Due to randomness, the more probable mapping for $n = 10, 25, 50$ is incorrect, but the confidence value is closer to 0.5 than 1 in these cases. As n increases to 1000 and beyond, the confidence values increase and the attacker becomes nearly certain that the inference is correct. An attacker who cannot compute confidence values cannot tell whether he is in the situation depicted in top left plot of Figure 2 or in the bottom right plot.

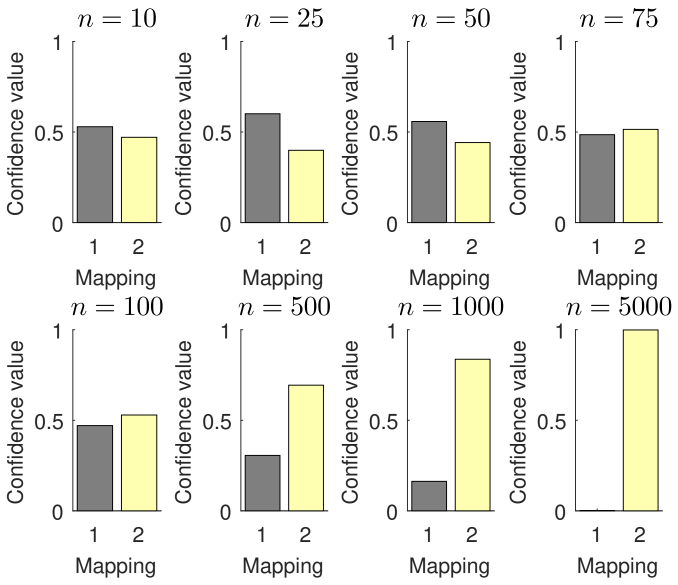

Figure 2: Confidence values for the multinomial attack on column `SEX` of the ACS dataset. The correct mapping is in (light) yellow.

5 Inferring Multiple PRE Columns

Prior inference techniques against PRE-encrypted databases target each column independently and fail in a variety of situations. For example, they do not work against a column whose values are uniformly distributed, even if these values are strongly correlated with (and can thus be inferred from) another column's plaintext values.

In this section, we extend our multinomial attack to multiple columns. In subsection 7.3, we show how this dramatically increases recovery ratios versus our single-column multinomial attack, which in turn outperforms previous work.

5.1 Exploiting cross-column correlations

A naive approach would extend the Bayesian inference task to the entire set of PRE-encrypted columns, using the joint prior on these columns as estimated from the auxiliary data. Unfortunately, this suffers from the curse of dimensionality: as the number of columns (the dimension) grows, the ability of any reasonable auxiliary data source to provide accurate estimates of the full joint prior diminishes.

Another potential approach is to use partitioning. First, apply a single-column attack to recover some column's plaintexts, then partition the database into separate databases (one for each of the recovered plaintext values), then attack other columns with a single-column attack but treating each partition separately and using only the relevant auxiliary data. For example, consider a two-column database with states and first names. The partitioning approach could first try to recover the state in each row, then attack first names separately using per-state auxiliary information about first names. The problem with partitioning is that the same first name may appear in rows with different states, thus the attack will yield inconsistent results (i.e., map a ciphertext to two different first names).

Instead, we extend our multinomial inference to directly account for correlations across columns. We start with a single target column. For expositional simplicity, we assume that it depends on one other column, but the attack easily generalizes to an arbitrary number $k \geq 1$ of dependent columns.

Let f_t denote the (unknown) encryption function for the target column, and let f_d denote the (recovered) encryption function for the dependent column. Assume that the target column has m values, and the dependent column m_d values. Using the chain rule for probability, $\mathbb{P}\mathrm{r}\{\mathbf{f_t} = f_t, \mathbf{f_d} = f_d \mid \vec{c}\ ; \rho\}$ is equal to $\mathbb{P}\mathrm{r}\{\mathbf{f_d} = f_d \mid \vec{c}\ ; \rho\} \cdot \mathbb{P}\mathrm{r}\{\mathbf{f_t} = f_t \mid \vec{c}\ ; \rho, f_d\}$.

The attack consists in estimating $\mathbb{P}\mathrm{r}\{\mathbf{f_t} = f \mid \vec{c}\ ; \rho, f_d\}$, assuming we have already discovered the correct mapping for the dependent column, f_d. The inference is:

$$\mathbb{P}\mathrm{r}\{\mathbf{f_t} = f \mid \vec{c}\ ; \rho, f_d\} = \frac{\mathbb{P}\mathrm{r}\{\vec{c} \mid \mathbf{f_t} = f\ ; \rho, f_d\} \cdot \mathbb{P}\mathrm{r}\{\mathbf{f_t} = f\ ; \rho\}}{\mathbb{P}\mathrm{r}\{\vec{c}\ ; \rho, f_d\}}\ , \tag{4}$$

where we assume the prior to be independent of f_d.

The main technical challenge for finding the best mapping f_t is how to compute the term $\mathbb{P}\mathrm{r}\{\vec{c} \mid \mathbf{f_t} = f\ ; \rho, f_d\}$. Here, $\vec{c}$ is a vector of pairs of ciphertexts and ρ is the joint distribution over the two columns. Without loss of generality, we assume that the first column is the target and the second is the dependent. Then $\rho_{i,j}$ is the joint probability or proportion of records with the i^{th} plaintext value for the target column and the j^{th} plaintext value for the dependent column. Similarly, let $c_{i,j}$ denote the number of records with the i^{th} ciphertext value for the target and the j^{th} ciphertext value for the dependent.

We model the likelihood as a multinomial distribution with outcome probabilities ρ given by the (known) value of f_d:

$$\mathbb{P}\mathrm{r}\{\vec{c} \mid \mathbf{f_t} = f\ ; \rho, f_d\} = \mathbb{P}\mathrm{r}\{c_{1,1}, \ldots, c_{m,m_d} \mid \mathbf{f_t} = f\ ; \rho, f_d\} = K_c \prod_{i,j} \rho_{i,j}^{c_{f(i),f_d(j)}}\ . \tag{5}$$

Here K_c is a constant that depends only on $\vec{c}$ (as before). The product is taken over all pairs of plaintexts for both the target and dependent columns, but the unknown part (i.e., f) is only over the target column mapping.

To find f that maximizes Equation 4, we solve the appropriate bipartite matching problem. Taking the log of the likelihood, the weight of edge (i, j) is $\sum_k c_{j,k} \log(\rho_{i,f_d(k)})$. For DE, we find the maximum-weight bipartite matching by solving the associated linear assignment problem. For ORE, we find the maximum-weight non-crossing matching. The running time of the two algorithms is the same as for a single column, but the complexity of computing edge weights increases by a factor of m_d.

5.2 Discovering correlated columns

To apply the techniques described above, the attacker must discover dependent columns for a given target. He can use the auxiliary data to search for them using some correlation measure such as conditional entropy, i.e., the amount of uncertainty about the target column given knowledge of the dependent column. Let $D = \{\mathbf{d_1}, \ldots, \mathbf{d_\ell}\}$ be the set of random variables corresponding to columns. If random variables $\mathbf{d_j}$ and $\mathbf{d_i}$ ($i, j \in [\ell]$) represent the target and dependent columns, respectively, and $H(\cdot)$ is the entropy function, the attacker wants to find a dependent column that minimizes $H(\mathbf{d_j}|\mathbf{d_i}) = H(\mathbf{d_i}, \mathbf{d_j}) - H(\mathbf{d_i})$.

5.3 Attacking multiple columns

A pragmatic approach is to first use a single-column attack against the columns for which they are likely to be successful and then, for each remaining column, identify one or more dependent columns among those recovered by the single-column attacks. Once a particular target column is recovered by the multi-column attack, it can be used as a dependent column for another column. The attack against the entire database can thus be represented as a directed acyclic dependency graph (DAG). In this graph, a node is associated to each column and an edge (i, j) indicates that column i will be used as a dependent to attack column j. Columns with no incoming edges are attacked using the single-column attack. The order in which the other columns are attacked is obtained via topological sort.

In Appendix A, we sketch an alternative approach to multi-column attacks that uses Markov-chain Monte Carlo (MCMC) techniques to attack multiple columns simultaneously. It may outperform the approach described above in settings where deciding which columns are correlated is difficult. In all of our case studies, however, determining cross-column correlations was straightforward. We leave implementing the MCMC approach and comparing it to the one presented above as an open problem for future work.

6 Inferring Redacted Columns

Because all previous inference heuristics target database columns one at a time, none of them can recover any information about columns that are redacted, i.e., strongly encrypted or segmented. This observation is the basis of "column sensitivity analysis" [81], proposed by the advocates of encrypted databases as the defense against the original inference attack of [68]. Encrypt the sensitive columns using strong encryption, encrypt the rest of the columns using PRE, and the sensitive columns will be protected even if the PRE columns fall to an inference attack.

The security of "sensitivity analysis" implicitly relies on data being independent across columns, which is unlikely to hold in practice. We explore two techniques, both of which leverage the values recovered from the PRE columns in combination with the auxiliary data, to infer the values of the strongly encrypted (or, in general, redacted) columns.

Record linkage. If the auxiliary dataset includes records of the same individuals as the target database, the attacker can use classic record linkage [29, 87]. The values of the PRE columns recovered by the multinomial attack serve as quasi-identifiers. The attacker performs an SQL-style inner join of these columns with the auxiliary data. In each row of the resulting table, the value of the redacted column (which comes from the auxiliary database) should be identical to the value of the same column in the target database. In subsection 7.5, we successfully apply this technique to a real dataset using real auxiliary data obtained from public sources.

Due to errors and incomplete data in the auxiliary dataset, record linkage may not recover the redacted column for every row. Our experiments in subsection 7.5 are very conservative and require the exact match on all quasi-identifier columns without any fuzziness (e.g., if the first name is "William" in the target database and "Will" in the auxiliary database, they will not match). Our reported recovery rates are therefore a lower bound on what a real adversary might achieve, especially using techniques for partial matching on noisy attributes [67].

Machine learning. When the auxiliary data yields only statistical information, as opposed to records of specific individuals, a more general approach is needed. The attacker can use the auxiliary dataset to train a prediction model in which the PRE-protected and unencrypted columns are the features and the redacted column is the prediction target. In our experiments in subsection 7.3, we use classifiers to predict categorical columns and regression to predict numerical columns.

Given a target database, the attacker applies the model to each row, predicting the value of the redacted column. This step critically depends on the previous single- and multi-column PRE attacks because they produce the features needed to apply the model. The accuracy of the recovery on the redacted column can be as much as the cross-validation accuracy of the model on the auxiliary dataset, provided the plaintexts recovered by the PRE attacks are not too noisy.

Directly using the prediction to infer the redacted value is not the best strategy if the correlations are weak or if the target column is binary with a highly skewed distribution, i.e., one label has a very low frequency. The latter case is common in medical databases. If the target column is a binary medical condition (e.g., some form of cancer), then most records have 0 in this column and very few (e.g., 0.1%) have 1. Simply predicting this binary value for each record is not a good strategy because (1) always predicting 0 (without even looking at the record!) already yields 99.9% accuracy, and (2) only the value of 1 is sensitive (i.e., the patient has this condition).

In this scenario, the attacker should define the (minority) sensitive value as the positive class and make predictions only for a small subset of records. Because **only positive labels leak information**, this strategy deliberately yields low coverage due to the asymmetry of the reward. By contrast, predicting the negative label for all records yields full coverage and high accuracy, yet is essentially useless.

The question then is how to predict the positive label so as to maximize the number of true positives (i.e., accuracy). Our classifier scores each record with respect to the positive label and then predicts positive values for the k highest-scoring records, where k is small enough that we expect more than k records with positive labels in the dataset, yet large enough that the results are statistically meaningful (e.g., $k = 100$ or 1000 for a dataset of millions of records).

Attribute	Type	Encryption
State	C	DE
Gender	B	DE
First Name	S	ORE
Last Name	S	ORE
ZIP code	S	ORE
Address	S	Redacted

Figure 3: FOP data elements, data types (C = categorical, B = binary, S = string) and corresponding encryption type.

7 Case Studies

We now evaluate our inference framework on several realistic case studies. In each scenario, we populate a database DB with one of the datasets described in subsection 7.1. These datasets are publicly available (one is from an actual database compromise leaked online), but all of them are representative of the data that encrypted databases intend to protect. For each DB, we apply an appropriate privacy schema (involving both encryption and redaction), producing an encrypted database EDB. Then, assuming an adversary has obtained EDB, as well as some auxiliary data, we ask: how much information about DB can the adversary infer? The ultimate goal is to recover entire plaintexts with high accuracy, but even partial leakage can be extremely damaging when the database holds healthcare records, financial data, or home addresses.

In each case study, the first step is *auxiliary data processing.* It produces an estimate ρ of the distributions of plaintext values in the columns of the target database DB. The second step is *inferring PRE columns.* It uses ρ and the multinomial attacks from Sections 4 and 5 to generate a database $\mathrm{DB}'_{\mathrm{pred}}$, where the PRE-encrypted columns are filled with the inferred values. The third step is *inferring redacted values.* It uses ρ and $\mathrm{DB}'_{\mathrm{pred}}$ to infer the strongly encrypted and segmented columns of DB and produce the final inferred database $\mathrm{DB}_{\mathrm{pred}}$, where all cells are filled with the inferred values or "don't know."

7.1 Datasets

Hospital patient data. The Nationwide Inpatient Sample (NIS) of the Healthcare Cost and Utilization Project (HCUP) [48], from the Agency for Healthcare Research and Quality (AHRQ), is a database of U.S. hospital discharge data. It is available for purchase after completing an online training course describing the Data Use Agreement (DUA). We strictly abide by this DUA. We do not perform re-identification attacks on this data, nor do we present new techniques that would enable others to perform such attacks.

We use the 2012 and 2013 versions of the database with $7,296,968$ and $7,119,563$ records, respectively. We pre-process the data by cleaning it (e.g., removing records with missing attribute values) and extracting a subset of the attributes, obtaining $6,538,031$ and $6,352,914$ records, respectively. Attributes include demographic information (age, sex, race), medical information (length of stay, disposition of patient, major operating room procedures, etc.), and individual Clinical Classifications Software (CCS) [49] category codes, which are fine-grained binary attributes corresponding to specific medical conditions or sets of related conditions. See Table 1 for a full list of attributes.

Census data. The American Community Survey (ACS) dataset consists of millions of (anonymized and re-sampled) U.S. Census records. It is publicly available for download from the US Census Bureau website [91]. We use the 2012 and 2013 versions of the dataset with $3,113,030$ and $3,132,795$ records, respectively. We pre-process the data by cleaning it (e.g., removing records with missing attribute values) and extracting a subset of the attributes, obtaining $2,563,935$ and $2,582,042$ records, respectively. Attributes include demographic information such as age and sex, as well as the individual's total yearly income. See Table 2 for a full list of attributes.

Police union member data. The Fraternal Order of Police (FOP) is the oldest and largest law enforcement union in the U.S. In 2015, their database servers were breached and a dump of their membership database was released online, with the full names, addresses, birthdates, and other sensitive information of over 600,000 current and retired FOP members. We use a publicly available copy of this dataset. See Figure 3 in for a full list of attributes.

Attribute	Description	Type	Encryption
`AGE`	Age (in years)	N	ORE
`FEMALE`	Female indicator	B	DE
`RACE`	Race	C	DE
`DIED`	Patient died	B	DE
`ELECTIVE`	Elective admission	B	DE
`HOSPBRTH`	In-hospital birth	B	DE
`NEOMAT`	Neonatal / maternal code	C	DE
`LOS`	Length of stay	N	ORE
`DISPUNIFORM`	Disposition of patient	C	DE
`NCHRONIC`	Number of chronic conditions	N	ORE
`ORPROC`	Major operating room procedure	B	DE
`PAY1`	Expected primary payer	C	DE
`ZIPINC_QRTL`	Median household income quartile for patient's ZIP code	C	DE
`TOTCHG`	Total charges	N	ORE
`MORTALITY`	Risk of mortality	C	DE
`SEVERITY`	Severity of illness	C	DE

Table 1: HCUP-NIS data elements, data types (C = categorical, B = binary, N = numerical) and corresponding encryption type. Omitted are the 262 binary-valued columns `CAT1`,`CAT2`,... for CCS diagnosis codes. For schema (A) these are all DE encrypted except a distinguished target `CAT` diagnosis code, which is redacted. For schema (B) all 262 `CAT` codes are redacted.

Auxiliary data. Our attacks rely on auxiliary data to obtain the distributions of values for a single attribute, correlations between attributes, and records of known individuals.

Standard sources of auxiliary data include publicly available datasets. For example, in our attack on the FOP dataset, we use U.S. voter registration lists, which are available for many states either for free, or for a nominal fee. We used databases from 7 states (Colorado, Connecticut, Florida, Michigan, North Carolina, Oklahoma, Washington) that are available online (e.g., [71]). The size of each dataset is shown in Table 3. We also purchased the Pennsylvania voter database from the Pennsylvania Department of State [19]. Pennsylvania is one of the most common states in the FOP dataset, which will be relevant in one of our attack scenarios. The exact information in the voter registration records varies from state to state, but all states collect full name, residential address including ZIP code, and birth year for each registered voter. Four states collect full birth date, and all but one collect gender. We also use per-state FOP membership statistics for six of the eight states, obtained directly from the websites of the state FOP lodges [30–35].

For our attacks on the hospital-discharge and ACS data, we use the 2012 dataset as the auxiliary data for attacking the 2013 dataset. None of our attacks exploit or assume that some records in the 2012 dataset refer to the same individuals as the 2013 dataset. Instead, we use the auxiliary data solely as the source of *statistical* information, such as correlations between demographic attributes or distributions of medical attributes. These scenarios model realistic attacks: a plaintext database is compromised, the breach motivates the data owner to deploy an encrypted database, some time later this encrypted database is compromised again. This pattern of multiple compromises occurs in practice. For example, Yahoo was repeatedly breached in 2013 and 2014.

In Section 7.4, we demonstrate that our attacks are robust even if the auxiliary data is noisier or more obsolete.

7.2 Scenario 1: Inferring sensitive medical conditions

This case study is based on the 2013 HCUP-NIS dataset described in subsection 7.1. We simulate an encrypted database using two different privacy schemas. In schema (A), the column corresponding to a specific medical condition (i.e., a CCS diagnosis code/category such as delirium dementia) is marked as sensitive and redacted; the other columns are less sensitive and encrypted with PRE. In schema (B), all columns corresponding to medical conditions (i.e., all CCS diagnoses) are marked as sensitive and redacted; the other columns are encrypted with PRE as shown in Table 1.

Attribute	**Description**	**Type**	**Encryption**
`AGE`	Age (in years)	N	ORE
`SEX`	Sex	B	DE
`RACE`	Race	C	DE
`REL`	Relationship code	C	DE
`MS`	Marital status	C	DE
`OCCFC`	Occupation code	C	DE
`COGD`	Cognitive difficulty	B	DE
`DIS`	Disability	C	DE
`ENG`	English language ability	C	DE
`LANX`	Language other than English spoken	C	DE
`EDU`	Educational attainment	C	DE
`FOD1`	Field of degree 1	C	DE
`FOD2`	Field of degree 2	C	DE
`WORKCLASS`	Class of worker	C	DE
`HPW`	Usual work hours per week	N	ORE
`WLW`	When last worked	C	DE
`TTTW`	Travel time to work	N	ORE
`CITZ`	Citizenship status	C	DE
`ST`	State	C	DE
`POW`	Place of work	C	DE
`POB`	Place of birth	C	DE
`PUMA`	Public use microdata area code	C	DE
`INC`	Person's total yearly income	N	Redacted

Table 2: ACS data elements, data types (C = categorical, B = binary, N = numerical) and corresponding encryption type.

State	**# Records**
Florida	13,677,763
Pennsylvania	8,722,571
North Carolina	7,559,168
Michigan	7,285,525
Washington	4,595,216
Colorado	3,600,733
Connecticut	2,367,707
Oklahoma	2,027,711

Table 3: Number of records in voter databases used as auxiliary data for FOP case study.

Auxiliary data. The goal of this experiment is not to perform a re-identification attack, but to use the statistics extracted from the auxiliary data to help infer the values in the encrypted database. We use the auxiliary data (in this scenario, the 2012 version of the dataset—see subsection 7.1 for why this is realistic) to construct a classifier that models the relationship between PRE-encrypted and sensitive columns.

We use MATLAB to train a `GentleBoost` ensemble using 40 tree-based weak learners. We specify for each feature column whether it contains numerical or categorical values and train the classifier with the default parameters on at least 1 million records. For both privacy schemas, the classifier predicts the value of a binary column associated with some medical condition. Predicting a value of 1 (the patient has the condition) is called a *positive prediction*.

For schema (A), we iterate over all single-code CCS categories and train the classifiers for each using the selected column as the target and the other columns as the features. This corresponds to the case where information on specific medical conditions (e.g., those related to mental health) is redacted, as in prior work on segmentation [39, 43].

For schema (B), we iterate over all single-code CCS categories and multi-code chapters and train the classifiers for each using the selected column as the target and only the non-CCS diagnosis columns (the first

16 columns from Table 1) as the features. This is an extremely conservative privacy schema, in which the information redacted from the database far exceeds that considered in prior work [39, 43]. The attacker's job is significantly harder in this case.

Inferring PRE columns. We use the single-column multinomial attack from section 4 against each PRE-encrypted column. We start with the estimated probability distribution ρ over the values of that column, where ρ_i is the proportion of records with the i^{th} value in the auxiliary dataset. We then compute c_j, the number of occurrences of the j^{th} ciphertext value in the target for every j. Finally, we perform the attack using ρ and $\vec{c}$ with the appropriate graph assignment.

For DE columns, we use a MATLAB implementation of the Hungarian Algorithm to find the best matching, i.e., pairs (j, i) such that the j^{th} ciphertext is mapped to the i^{th} plaintext. For ORE, we use our own MATLAB implementation of the maximum-weight non-crossing matching [42].[2]

For both privacy schemas, the attack almost fully recovers the correct plaintexts all PRE columns. The plaintexts of DE columns are recovered with 100% accuracy, i.e., we correctly find the entire permutation for each such column. The reason is that these columns have a highly non-uniform distribution over small domains. For example, all CCS diagnosis columns are binary-valued with 0 mode.

The accuracy of recovery for the numerical columns is also high. Columns `AGE`, `LOS`, and `NCHRONIC` have values from a relatively small domain and the multinomial attack recovers 100% of the plaintexts. By contrast, `TOTCHG` has values from a large sparse domain (e.g., few patients have the exact same total charges). The attack does not always recover the exact value, but the median error for `TOTCHG` is $1,052$, thus the recovered plaintexts are close to the true value.

The accuracy is so high for our single-column attack that there is no need to resort to the multi-column attack.

Inferring sensitive columns. As explained in Section 6, if the fraction of records with the positive label in the sensitive column is low (e.g., 1%), overall accuracy is not the right metric. For example, always predicting that the patient does *not* have the condition has high overall accuracy, yet is useless.

Instead, we aim to maximize *accuracy over the positive predictions.* For each sensitive column, we apply the classifier trained on the auxiliary data to the partial records recovered by the multinomial attack on the PRE columns. The classifier scores them w.r.t. the positive labels, we pick 100 records with the highest scores and only output predictions for those. Accuracy is the fraction of those 100 that are true positives.

Of 262 medical conditions, positive labels for 40 can be inferred with 75% or higher accuracy. Inferring even a single medical condition is a serious privacy breach, thus failure to infer certain conditions (false negatives) should not be considered a flaw of our methodology, as long as our false positives are low. Table 4 shows the results for the 10 conditions with the highest accuracy for the privacy schema (A). The table also shows the prevalence of each condition, i.e., the proportion of records with the positive label for that condition in the database. To estimate the variance in accuracy, we repeat the score-and-predict experiment 10 times and report the mean accuracy and standard deviation. Each experiment operates on a random subset of the partial records recovered by the PRE attack. We can successfully infer many sensitive medical conditions, including hepatitis, leukemias, chronic kidney disease, and mood disorders.

Table 5 shows the results for the privacy schema (B). 10 of the 40 conditions can still be inferred, including chronic kidney disease and mood disorders, albeit with slightly lower accuracy. This is significant, because none of the 16 columns available to the classifier in this scenario are sensitive; they provide only general or aggregate information. This demonstrates the limits of redaction: **even strongly encrypted or segmented medical conditions can be successfully inferred from the non-sensitive columns, which, in turn, are inferred using the multinomial attack**.

An attacker may also group semantically similar categories and try to infer the whole group, e.g., find patients who suffer from *some* mental illness as opposed to a specific one. Table 6 shows the results for groups of diagnoses guided by the CCS hierarchy. We found 7 groups (out of 20) that can be inferred with accuracy of 75% or more.

Discussion. For the CCS categories other than those in Tables 4, 5 and 6, the attack accuracy is below 75%. Some cancer-related categories are especially difficult to infer, e.g., for cancer of bone and connective

[2]For column `TOTCHG` we use a sorting attack instead [68]. This column is sparse with a large domain, thus frequency information is difficult to leverage and, further, the multinomial attack is computationally intensive. As a limiting case, when the numbers of plaintext and ciphertext values are identical, the sorting attack is equivalent to the multinomial attack.

Medical Condition (CCS Diagnosis Category)	Prevalence	Accuracy (±stddev)
Diabetes mellitus with complications	7.08%	100.00%(±0.00%)
Diabetes or abnormal glucose tolerance complicating pregnancy; childbirth; or the puerperium	0.91%	99.40%(±0.70%)
Other complications of pregnancy	5.56%	98.80%(±1.03%)
Chronic kidney disease	11.75%	97.50%(±1.18%)
Hypertension with complications and secondary hypertension	11.80%	97.30%(±1.42%)
Respiratory failure; insufficiency; arrest (adult)	8.09%	96.80%(±1.55%)
Mood disorders	14.59%	96.30%(±0.95%)
Congestive heart failure; nonhypertensive	11.96%	95.90%(±1.66%)
Coronary atherosclerosis and other heart disease	17.66%	95.80%(±1.81%)
Infective arthritis and osteomyelitis (except that caused by TB or STD)	0.97%	95.80%(±1.81%)
Rehabilitation care; fitting of prostheses; and adjustment of devices	1.16%	95.40%(±1.71%)
Hypertension complicating pregnancy; childbirth and the puerperium	1.25%	95.40%(±2.50%)
Chronic obstructive pulmonary disease and bronchiectasis	11.97%	95.10%(±1.73%)
Miscellaneous mental disorders	1.06%	94.20%(±2.04%)
Acute cerebrovascular disease	2.11%	93.40%(±2.22%)
Urinary tract infections	8.66%	92.20%(±2.82%)
Acute and unspecified renal failure	9.63%	91.00%(±2.49%)
Other liver diseases	4.80%	89.60%(±3.06%)
Osteoarthritis	8.29%	88.90%(±1.85%)
Delirium dementia and amnestic and other cognitive disorders	6.00%	86.50%(±4.88%)
Cardiac dysrhythmias	16.19%	86.40%(±4.12%)
Chronic ulcer of skin	3.02%	85.90%(±2.47%)
Paralysis	1.71%	83.60%(±2.67%)
Esophageal disorders	14.99%	83.50%(±3.14%)
Shock	1.71%	83.40%(±4.20%)
Diabetes mellitus without complication	15.69%	82.50%(±4.45%)
Thyroid disorders	10.80%	82.20%(±3.16%)
Diseases of white blood cells	3.85%	82.10%(±3.87%)
Substance-related disorders	5.16%	82.10%(±2.60%)
Maintenance chemotherapy; radiotherapy	0.45%	81.30%(±3.02%)
Anxiety disorders	9.19%	80.50%(±3.37%)
Crushing injury or internal injury	0.66%	80.40%(±3.53%)
Alcohol-related disorders	4.93%	78.90%(±3.48%)
Cardiac and circulatory congenital anomalies	0.94%	78.50%(±2.51%)
Menstrual disorders	0.55%	78.00%(±4.45%)
Septicemia (except in labor)	5.66%	77.70%(±3.62%)
Leukemias	0.73%	77.30%(±3.23%)
Complications of surgical procedures or medical care	4.60%	76.60%(±3.34%)
Hepatitis	2.05%	75.80%(±2.82%)
Hyperplasia of prostate	3.64%	75.00%(±2.54%)

Table 4: Prediction accuracy (for 100 positive predictions) for the redacted medical conditions, privacy schema (A). Rows corresponding to the conditions for which we achieve over 90% accuracy are shaded.

Medical Condition (CCS Diagnosis Category)	Prevalence	Accuracy (±stddev)
Other complications of pregnancy	5.56%	92.30%(±2.11%)
Respiratory failure; insufficiency; arrest (adult)	8.09%	90.70%(±2.91%)
Mood disorders	14.59%	84.40%(±2.95%)
Chronic kidney disease	11.75%	82.40%(±2.46%)
Coronary atherosclerosis and other heart disease	17.66%	81.90%(±2.60%)
Hypertension complicating pregnancy; childbirth and the puerperium	1.25%	81.40%(±2.55%)
Osteoarthritis	8.29%	81.40%(±3.03%)
Hypertension with complications and secondary hypertension	11.80%	79.10%(±4.38%)
Cardiac and circulatory congenital anomalies	0.94%	78.70%(±3.50%)
Congestive heart failure; nonhypertensive	11.96%	76.40%(±4.25%)

Table 5: Prediction accuracy (for 100 positive predictions) for the redacted medical conditions, privacy schema (B). These conditions are a subset of those for privacy schema (A). Rows corresponding to the conditions for which we achieve over 90% accuracy are shaded.

Group of Medical Conditions	Accuracy (±stddev)
Mental Health / Substance Abuse (MHSA)	99.30%(±0.48%)
Strict Mental Health / Substance Abuse	97.10%(±1.85%)
Mental Health	96.90%(±1.29%)
Diseases of the musculoskeletal system and connective tissue	94.90%(±1.52%)
Congenital anomalies	93.40%(±2.76%)
Diseases of the nervous system and sense organs	82.70%(±2.98%)
Substance and Alcohol Abuse	81.00%(±2.21%)

Table 6: Prediction accuracy (for 100 positive predictions) for groups of semantically related medical conditions, privacy schema (B). Rows corresponding to the conditions for which we achieve over 90% accuracy are shaded.

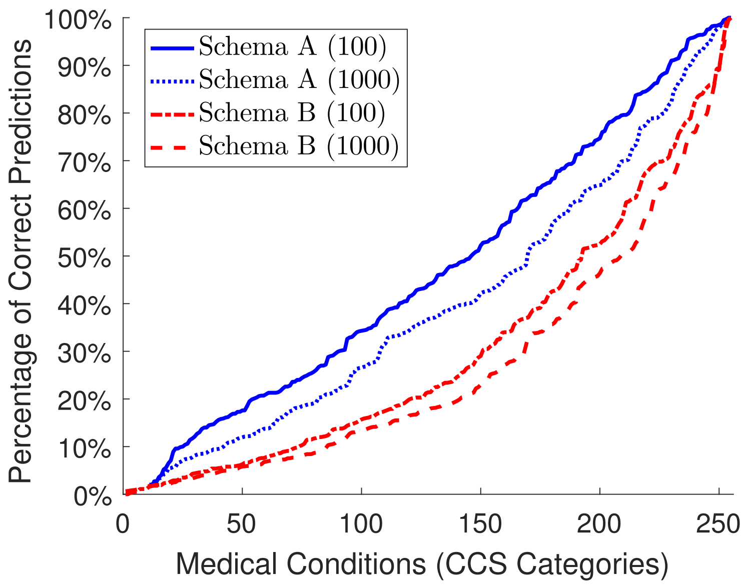

Figure 4: Percentage of correct positive predictions for all CCS categories for privacy schema (A) with 100 (blue solid line) and 1000 (blue dotted line) predictions, and privacy schema (B) with 100 (red dot-dashed line) and 1000 (red dashed line) predictions.

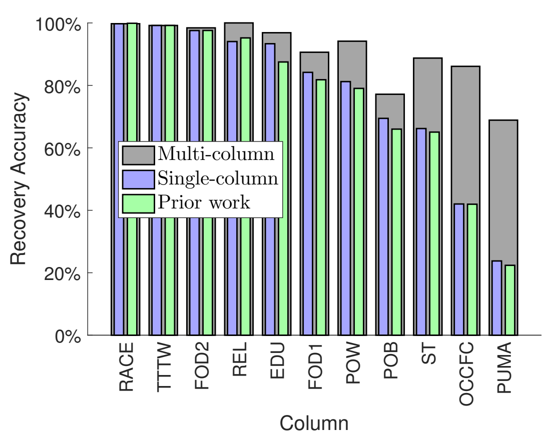

Figure 5: Recovery accuracy for PRE-encrypted ACS columns. The plot compares the multinomial attack (single-column and mutli-column) with prior work across the columns that we cannot recover with 100% accuracy in all cases.

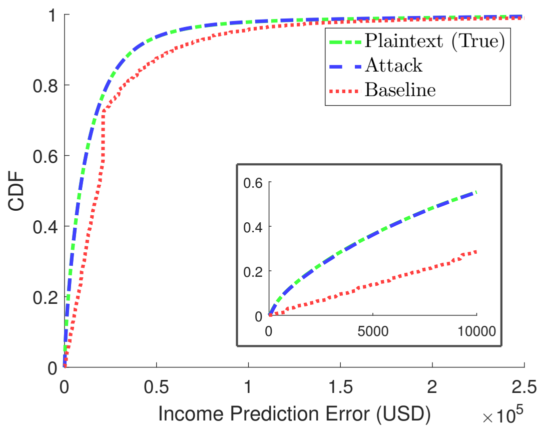

Figure 6: Empirical CDF of the prediction error for yearly income (blue dashed line). There are two points of comparison: (1) the baseline predicts median yearly income for each record (red dotted line), and (2) CDF of the prediction error for a model trained on the plaintext (true) target database (green dashed line).

tissue (prevalence 0.15%), only about 36% of the classifier's predictions are correct. Figure 4 shows the percentage of correct predictions for both privacy schemas across all CCS categories.

Sensitivity analysis based on measuring correlations between columns will fail to prevent our attack because weak correlations can enable accurate inferences. Consider Substance and Alcohol Abuse (Table 6), which has prevalence of about 8.6%. To understand the classifier's behavior, we look at the typical characteristics among the positive predictions, i.e., features that are popular among the positive records but rare in the population. Around 96% of the positive predicted records have 7 in the `DISPUNIFORM` column ("disposition of patient: against medical advice"), whereas only 1.02% of all records have 7 in this column.

The correlation between the target and the `DISPUNIFORM` column is very low (the Pearson correlation coefficient is -0.007). Therefore, even an experienced security architect may deem the `DISPUNIFORM` column as not sensitive. Nevertheless, because $\mathbb{P}\mathrm{r}\{\text{S. \& A. Abuse} = 1 \mid \texttt{DISPUNIFORM} = 7\} = 0.411$, the classifier uses this column to make accurate positive predictions. Since $\mathbb{P}\mathrm{r}\{\text{S. \& A. Abuse} = 1\} = 0.086$, predicting a positive label if $\texttt{DISPUNIFORM} = 7$ increases the probability that this prediction is correct by five times.

7.3 Scenario 2: Inferring sensitive demographic data

This case study is based on the 2013 ACS dataset described in subsection 7.1. We apply PRE to all columns except income, which is redacted. The auxiliary information is the 2012 dataset, but we also consider noisier and more obsolete auxiliary data in subsection 7.4.

Auxiliary data. Because the redacted column (`INC`) is numerical (as opposed to binary for the medical case study), we train a regression model on 1 million records from the auxiliary dataset. We do this in MATLAB with an ensemble of 40 regression trees using bagging with the default parameters. For each column, we specify whether it contains numerical or categorical values. `INC` is the prediction target, the other columns are the features. To improve the quality of the regression model, we convert categorical features to binary vectors. For example, we replace the column `SEX` which can take two values (male and female) with two new binary columns `SEX_MALE` and `SEX_FEMALE`.

Inferring PRE columns. We start with a single-column multinomial attack on each PRE column independently, as in Table 7.2. For about half of the columns, the recovery is perfect (100% accuracy). These columns are `AGE`, `SEX`, `MS`, `COGD`, `DIS`, `ENG`, `LANX`, `WORKCLASS`, `HPW`, `WLW`, and `CITZ`.

For the rest of the PRE columns, we improve the recovery using the multi-column attack from section 5. We first create a dependency graph. To this end, we compute conditional pairwise entropy between columns

Sub-population		Median Error (USD)
Characteristic (Column = Value)	Proportion	
EDU = 12 (Grade 9)	2.76%	1371.4
EDU = 13 (Grade 10)	3.34%	1816.5
REL = 2 (Biological son or daughter)	12.40%	2024.9
REL = 4 (Stepson or stepdaughter)	0.71%	1899.8
OCCFC = 3955 (PRT-LIFEGUARDS AND OTHER RECREATIONAL, AND ALL OTHER PROTECTIVE SERVICE WORKERS)	0.16%	1849.7

Table 7: Median prediction error for some sub-populations.

using the auxiliary data. As explained in section 5, we cannot simply pick the most correlated dependent column because this may create a cycle. Also, we must consider (1) the number of plaintext values in the dependent column (fewer is better due to the curse of dimensionality), and (2) the length of the longest path (longer paths may result in cascading failures if there are mistakes).

We end up with the following acyclic dependency graph: RACE depends on POB, REL on MS, OCCFC on WORKCLASS, EDU on FOD1, FOD1 on OCCFC, FOD2 on FOD1, POW on EDU, ST on POW, PUMA on ST, and POB on CITZ. With these, we discover a valid topological order and perform the attack using the edge weights computed as in section 5.

We compare three attacks: best prior heuristic [42], single-column multinomial, and multi-column multinomial. All three are performed using the appropriate graph matching algorithm for each column, i.e., the Hungarian algorithm for DE and MWNCM [42] for ORE. We compute the overall row recovery rates, averaged over five experiments, as the percentage of rows across all 22 PRE columns that are recovered perfectly (no errors). These rates are 2.10% (±0.46%) for the attack of [42], 2.75% (±0.55%) for the single-column attack, and 33.38% (±7.05%) for the multi-column attack. In other words, **the recovery rate of our multi-column multinomial attack is 16 times that of prior work**.

Figure 5 shows the attack accuracy for the columns that are not recovered perfectly. The multi-column attack greatly outperforms the single-column attack, which in turn outperforms prior work. For OCCFC and PUMA, the multi-column attack doubles accuracy of the single-column attack.

Inferring sensitive columns. To infer the value of the redacted INC column, we apply our regression model trained on the auxiliary data to the plaintexts of the columns recovered by the multinomial attack. We compute the error as the absolute difference between the predicted and true income, and plot its empirical CDF in Figure 6. The figure compares it with (1) a naive baseline predictor that always predicts the median income of the population, and (2) a regression model directly trained on the true plaintext data.

The median error of our predictions is 8388.3 USD (i.e., income can be predicted to within around 8.4k USD for 50% of the individual records). This is much better than the baseline which attains such precision for only 20% of the records. Furthermore, the predictions almost perfectly coincide with a regression model trained on the true data (Figure 6). Even though the multinomial attack has recovered noisy plaintexts for some columns, the quality of predictions has not suffered.

Table 7 shows examples of population subgroups that are especially vulnerable to the attack. These subgroups share a common characteristic (e.g., the value of a certain column) that further reduces the prediction error. For example, the median prediction error for the sub-population whose educational attainment (EDU) is "Grade 9" is less than 1400 USD.

Overall, these results suggest that segmentation and strong encryption cannot protect the confidentiality even of large-domain, sparse, numerical columns.

7.4 Quality of auxiliary data

To demonstrate that our attacks are robust even if the auxiliary data is smaller or obsolete, we repeat the experiments on the 2013 ACS dataset but replace the auxiliary data with sub-samples of 10, 50, and 100 thousand records from the outdated ACS PUMS 2009, 2010, 2011, and 2012 datasets.

Figure 7 shows the attack accuracy for each combination of year and sample size. Overall, using old auxiliary data does not significantly lower the accuracy. For example, the median recovery accuracy (across

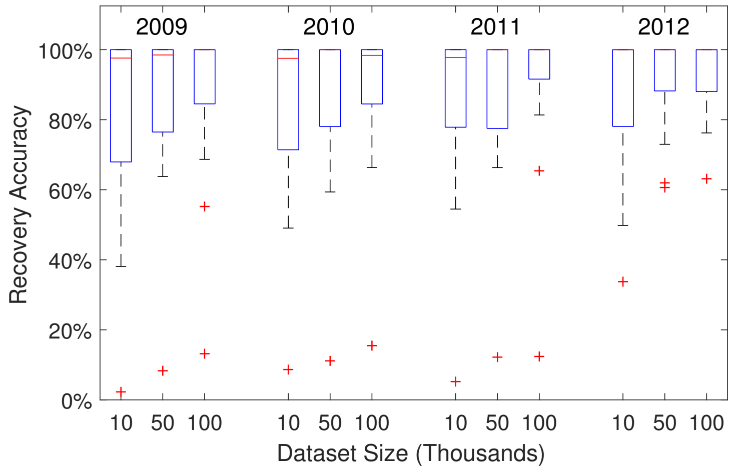

Figure 7: Recovery rates (accuracy) for degraded auxiliary data. Each box-whisker plot shows the recovery rate across all PRE columns.

		Auxiliary dataset year			
		2009	2010	2011	2012
Size	10k	9106.3	9067.8	9160.8	9428.1
	50k	8737.8	8564.9	8594.6	8587.7
	100k	8560.4	8598.2	8230.3	8501.4

Table 8: Median error for income prediction with outdated sub-samples of the auxiliary dataset.

all PRE columns) with the 2009 auxiliary data is near 100%, just like with the 2012 data. That said, for some columns (`OCCFC` and `PUMA`), the 2012 auxiliary data yields the highest accuracy by some margin. Using small sub-samples has more impact on the accuracy for some columns across all years (e.g., compare the box length for 10K and 100K), yet the overall accuracy remains high.

To evaluate the impact on income prediction (i.e., inferring the sensitive column), we train a regression model as in subsection 7.3 but on the degraded auxiliary data, then apply it to the plaintexts recovered from the PRE columns by the multinomial attack. For incomes of 50k or more, the predictions are (almost) on par with those of subsection 7.3. In all cases, the prediction error is substantially better than for the baseline (17702 USD). The median prediction error (in USD) is shown in Table 8, vs. 8388.3 USD for the attack with the 2012 auxiliary data (subsection 7.3).

Overall, this suggests that the attack will succeed even using auxiliary data that is a few years old, as long as it contains a few tens of thousands of records (e.g., it works well with 50K records, which is only 2% of the available auxiliary dataset).

In theory, the results of Section 7.3 may have "unfairly" benefitted if the auxiliary dataset contains some of the same records as the target dataset (i.e., they would not have been as good if the auxiliary and target data did not overlap). We cannot estimate the amount of overlap because ACS datasets do not include unique individual identifiers that are stable across yearly releases, but our results with degraded auxiliary data suggest that this overlap plays little or no role.

If our income prediction error had been low due to the overlap, then using smaller, outdated auxiliary data would have led to a noticeable increase in this error (since we keep the number of target records constant), but instead it is similar to subsection 7.3. Therefore, any overlap between the auxiliary and target datasets is irrelevant. In any case, our multinomial attack exploits only statistical information from the auxiliary data and does not depend on specific records.

7.5 Scenario 3: Inferring home addresses

This case study is based on the FOP dataset described in subsection 7.1. We simulate an encrypted database using the privacy schema in Figure 3 of Section 7. This privacy schema is informed by practical applications of encrypted databases. Prior work [42] motivated the use of ORE for first name, last name, and ZIP code, which are commonly sorted and queried by prefix in electronic health records and customer relationship management systems such as OpenEMR and Salesforce. State and gender are encrypted using DE to enable grouping by these attributes. Frequency analysis trivially infers both attributes because statewide FOP membership numbers are public (subsection 7.1) and because most police officers are men.

Home addresses of police officers are considered extremely sensitive due to the risk of reprisal by criminals. In Florida in 2010, a man was arrested for publishing the address of an officer online [98]. Furthermore, they have little utility as a query or filter value in applications. Therefore, we assume that addresses are encrypted using IND-CPA encryption.

Auxiliary data. As mentioned in subsection 7.1, we use the publicly available state voter registration records as the auxiliary data, as well as the ZIP code database with the list of valid ZIP codes for each state and the estimated population of each ZIP code [92]. We also use per-state first and last name distributions taken from the state's voter records.

Inferring PRE columns. We use the multi-column attacks to infer the first name, last name, and ZIP code. We infer last names and ZIP codes using the state as the dependent attribute. Because the set of possible ZIP codes is uniquely determined by the state, we partition the ZIP codes by state and run an independent attack for each state. For first names, we use both the state and gender as the dependent attributes.

The recovery rates for all attributes are high. The multi-column attack on last names performs especially well: for two states, the recovery rate is above 80%, whereas even our own single-column attack does not exceed 60%. The recovery rates for ZIP codes are surprisingly high, in contrast to prior work that recovered less than 2% of ORE-encrypted ZIP codes [42]. This is due to our ability to attack ZIP codes separately for each state, as well as the superiority of the multinomial attack over prior techniques. Recovery of ZIP codes for Florida and Colorado is much worse than for the other states due to the noise in the data: dozens of ZIP codes from other states (and even some non-existent ZIP codes) were included in the FOP database, and forcing the matching to preserve the order prevented correct matches. Increasing robustness of inference attacks to noise in the target data is left to future work.

Figure 9 directly compares our multi-column attack to our single-column attack and the best prior work. We show the overall recovery rates for the eight states for which we have the voter registration records. For first and last names, the multi-column attack yields a modest increase in accuracy (around 4% and 12%, respectively) over our single-column attack, which in turn is more accurate than the prior heuristics.

The most striking increase in accuracy is for the multi-column attack on ZIP codes. Whereas both single-column attacks (best prior and our own) perform poorly (2-5%), the multi-column attack recovers around 36% of all ZIP codes in the FOP records for these states. The whole-row recovery rate (i.e., first name, last name, gender, and ZIP code) of the multi-column attack is nearly *one quarter*, which is many times better than the single-column attacks. The whole-row recovery is highest for Connecticut: our attack recovers 46% of full rows, i.e., we **correctly infer the full name, gender, and ZIP code of almost half of FOP members in Connecticut**.

Inferring sensitive columns. Once the first name, last name, and ZIP code are recovered for each row, we link them with the state voter records, using the usaddress [17] python library to canonicalize street addresses.

We take an extremely conservative approach to measuring the accuracy of our attack. After canonicalization, we only consider an address to be recovered if it is an exact match. This penalizes us by not counting correct matches with typos due to data entry errors, nor does it count the correct recoveries of the address of a police officer's alternate residence. Another reason for the undercount is that not every officer is a registered voter. Using a more complete list of addresses from a phone directory would likely result in higher recovery rates.

For the eight states for which we had the auxiliary data, our aggregate recovery rate is about 7.7%, i.e., roughly 1 out of every 13 FOP members would have their exact home address revealed by this attack. Our attack was most successful for Michigan and Pennsylvania, recovering 13-14% for both. On a real encrypted database, this would have been equivalent to a massive privacy breach, with the home addresses of over 7,000 officers compromised in Pennsylvania alone.

	# rows	FN	LN	ZC	Addrs
	Encr. type	ORE	ORE	ORE	STR
Pennsylvania	54,428	92	69	58	7,251
Michigan	13,136	93	69	59	1,789
North Car.	14,862	92	83	44	1,683
Oklahoma	10,333	92	85	43	469
Washington	5,297	92	74	49	397
Connecticut	2,969	93	69	71	370
Florida	45,423	90	72	3	164
Colorado	11,311	90	74	0	0

Figure 8: Recovery rates for the multi-column attack on FOP attributes. "# rows" is the number of rows in the FOP database for a given state; "Encr. type" is the encryption scheme (STR is strong encryption); FN, LN, ZC, Addrs are the number of, respectively, correctly recovered first names, last names, ZIP codes, and home addresses.

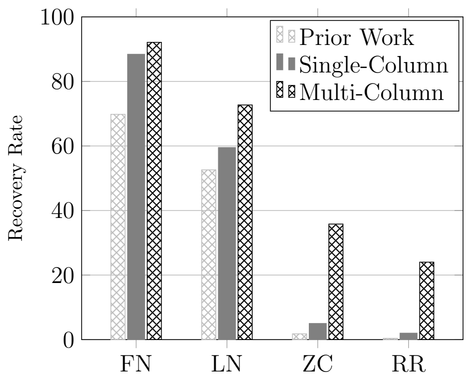

Figure 9: Recovery rates of the multi-column attack, single-column attack, and best prior heuristic on first names, last names, ZIP codes, and all three together (RR) on the eight target states.

8 Limitations and Mitigations

Our multi-column attacks depend on the correlations between columns and will fail if they are uncorrelated (rarely the case in real-world datasets). Even if correlations exist, success of our attacks depends on the privacy schema. If most or all columns are redacted or strongly encrypted, there is not enough statistical information revealed for successful inference.

This defense, however, requires that most columns be marked as "sensitive." We are not aware of any encrypted database that maintains performance and functionality in this scenario. In any case, such a privacy schema defeats the purpose of PRE. In practice, deployed encrypted databases use a utility-first approach to privacy schema and should reveal enough information for our attacks.

Another limitation is that any inference method, including ours, depends on the quality of the attacker's auxiliary data. In Section 7.4, we show that our attacks are robust even if the auxiliary data is smaller and more obsolete. Furthermore, our framework is the first to associate confidence values with the recovered mappings, helping assess the quality of inference.

Ensuring that individual PRE-encrypted columns have high entropy does not mitigate our attacks. A column can have high, even maximum entropy and still be correlated with other columns. For example, consider a database with columns `SEX` (male and female) and `AGE` (in years). Even if the proportion of males and females is the same (`SEX` has maximum entropy), the attacker can exploit the correlations with the `AGE` column, e.g., for people over 100 years old the ratio of females to males is around 4.82 [90]. Suppose that a diligent security architect bins `AGE` according to quartiles (call the new column `AGE_QRTL`). The entropy of `AGE_QRTL` is maximal because each bin contains exactly 1/4 of the samples. In the last quartile, the proportion of females will still be greater than that of males, and the attacker can still infer information.

In principle, some inference possibilities may be detected by measuring cross-column correlations, but this is not straightforward. In Section 7.2, we concretely showed that even weak correlations can, in aggregate, have high predictive power. Furthermore, in a database with m columns, any subset of the columns could be used to predict the value of another column. Thus, the number of correlation measurements required for inference detection grows exponentially in m.

9 Related Work

Encrypted databases. Companies including Ciphercloud [13], Navajo Systems [26], Skyhigh Networks [69], and Perspecsys [76] deploy proprietary PRE schemes to enable searching and sorting of ciphertexts stored within software-as-a-service (SaaS) systems. SAP's SEEED system [86], IQcrypt [51], and Microsoft's Always Encrypted [66] explicitly use privacy schema where some columns are strongly encrypted and other columns are DE- or ORE-encrypted.

CryptDB [79] is an academic proposal to use PRE in combination with "onion encryption" (also studied in [36]), which wraps PRE-encrypted data with strong encryption and relies on client proxies to expose PRE ciphertexts for querying. As far as our inference methodology is concerned, this is equivalent to a privacy schema in which the columns not used in queries are strongly encrypted.

Popa et al. claimed that CryptDB provides "provable confidentiality" [79] even if the database system is entirely controlled by the adversary. After recent attacks against PRE-encrypted databases (see below), Popa et al. revised their claims, advocating column sensitivity analysis and the use of strong encryption for columns deemed "sensitive" [81]. Monomi [89] performs a form of automated column sensitivity analysis and uses a mix of strong encryption and PRE.

Some approaches to encrypted databases achieve stronger security than the more practical systems we focus on in this paper. ZeroDB [25] interactively performs searches over a tree with strongly encrypted nodes (similar to [78]). Arx [77] is claimed to be secure against "snapshot" attacks but requires garbled circuits to perform queries [99]; also, it is not actually secure against realistic "snapshot" attacks [41]. Lewi and Wu [62], Faber et al. [27], and Roche et al. [84] present approaches supporting only range queries that leak less information to snapshot adversaries than the systems we investigate. Lewi and Wu's scheme can be adapted into a traditional ORE scheme, but at a cost of leaking information during a snapshot attack that is, in fact, greater than what is assumed by our attacks. Boneh et al. [7] and Kerschbaum [57] also present ORE schemes that do not leak plaintext frequency. These schemes are not yet used in practice due to interactivity and/or substantial re-engineering of database servers they require.

Academic systems such as Mylar [80], Mimesis Aegis [61], Shadowcrypt [47], Blindseer [75], and DupLESS [54] use PRE, and several papers demonstrated significant weaknesses in them [8,42,52,82,100]. These systems aim to protect unstructured text, as opposed to structured relational data, and we do not consider them in this paper.

Attacks on encrypted databases. Naveed et al. [68] demonstrated passive, single-column attacks against PRE-protected databases. Grubbs et al. [42] suggest a non-crossing attack that is more effective empirically against single-column ORE. Our multinomial attack improves and outperforms these attacks.

Kellaris et al. [55] and Lacharité et al. [59] give generic attacks against systems that support range queries. In the settings we consider, their attacks require a prohibitively large number of uniform queries. Our attacks do not rely on observing queries and are always faster.

None of the prior work exploits cross-column correlations except Durak et al. [22], who mount simple attacks against two-column, ORE-encrypted databases of GPS latitude-longitude coordinates. They apply, in parallel, independent single-column attacks and then use trivial geometric correlations when analyzing the resulting guesses.

Inference detection. There is a large body of work on inference detection for databases [18, 28, 50, 65] and some recent systems [101] use it to discover dependencies between sensitive and seemingly non-sensitive columns. Inference detection involves analyzing the database schema. For example, Hinke [50] suggests using a semantic graph to capture functional dependencies between attributes.

These techniques do not prevent our attacks because we rely on statistical relationships that exist regardless of the presence of certain columns in the schema. For example, if a sensitive attribute is segmented from the database, inference detection will declare the database safe even if the attribute can still be inferred from the remaining attributes.

Inference control. The problem of linking records in a protected database with the same individuals' records in other databases has been investigated since at least the 1950s [70]. In 1969, Fellegi and Sunter published a theory of record linkage, that is, "recognizing those records in two files which represent identical persons, objects or events" [29]. In 1975, Schlörer concretely demonstrated how an attacker who has prior information about an individual can identify his or her record in a database using only non-unique identifiers and statistical information [87]. A combination of attributes that are not identifying by themselves but together uniquely identify an individual is known as a quasi-identifier [16].

The goal of *inference control* [15] is to provide access to a database while preventing inference of sensitive information about individuals. Statistical disclosure control aims to answer statistical queries without revealing individual entries [1, 21, 23, 96, 97]. In our setting, the adversary has access to individual records (albeit with some attributes encrypted), i.e., *microdata*. Dozens of methods have been proposed for privacy-preserving release of microdata (see a survey in [20]), including cell swapping, suppression, microaggregation, and bucketization. All of these techniques assume that a trusted curator applies the privacy transformation to the database before it is released to the adversary. Encrypted databases aim to address a different scenario, when the system is unpredictably compromised and the adversary obtains the database "as is."

Our linkage methods draw on known techniques for linking datasets on quasi-identifiers [88] and incomplete, noisy common attributes [67]. Prior work on data segmentation argued that segmented or redacted medical codes can potentially be inferred using correlations between medical concepts [9, 43]. In particular, Chan et al. show how to extract correlations between medical concepts from the research literature and use this to construct a hypothetico-deductive model [9]. We use a different type of classifier and, most importantly, demonstrate how cross-column inference can be used on real data in privacy-protected databases. How to combine the medical-concept relationships developed by [9] with our inference methodology is an interesting topic for future research.

10 Conclusions

We developed a comprehensive methodology for inference attacks against the state-of-the-art database privacy protections and evaluated it on real data. Our attacks apply to commercial products used today to protect sensitive medical and business information, as well as the best academic constructions of efficient encrypted databases based on mixing property-revealing encryption (PRE) with strong encryption or data segmentation.

Our first conclusion is that segmentation and strong encryption are insufficient to protect the secrecy of sensitive data when used in conjunction with PRE. This contradicts recent claims [3, 56, 79] that strong encryption protects a column from inference regardless of how other columns are treated.

Our second conclusion is that column sensitivity analysis (c.f., [51, 66, 81, 89]) and inference detection (c.f., [18, 28, 50, 65]) have fundamental limitations. They analyze database schemas on a column-by-column basis, without considering all cross-column correlations, the possibility that weak correlations can enable accurate inferences, or that PRE-encrypted columns can be linked to public datasets. Segmentation in medical databases suffers from the same flaws.

To adequately protect sensitive columns in encrypted databases, the administrator must strongly encrypt or segment not only these columns, but also all other columns that may help infer the sensitive columns. This seems difficult, if not impossible, in practice. Straightforward extensions of column sensitivity analysis, such as measuring cross-column correlations, do not adequately measure the risk of cross-column inference—see our inference of the "substance and alcohol abuse" condition in Section 7.2. The administrator must also be aware of the correlated information that appears in external databases which are not under his control or must be public by law (e.g., voter registration lists). It remains an open question whether there exist column sensitivity analysis techniques that provide meaningful confidentiality guarantees.

Alternatively, one can attempt to reduce leakage by using stronger encryption. For example, several recent academic proposals for encrypted databases [74, 77, 81] claim to be fully secure against "snapshot" attacks. All of these schemes are either inefficient, or insecure in realistic scenarios [41]. Whether it is possible to build practical encrypted databases that avoid leakage on realistic datasets, even against snapshot adversaries, remains an open question.

Until this problem is solved, our framework can help practitioners and researchers assess the security of the database privacy protections that are predominant in practice today.

References

[1] Nabil R. Adam and John C. Wortmann. Security-control methods for statistical databases: A comparative study. *ACM Comput. Surv.*, 21(4), 1989.

[2] Anthem. Anthem data breach. `https://www.anthemfacts.com/`, 2016.

[3] Arvind Arasu, Spyros Blanas, Ken Eguro, Raghav Kaushik, Donald Kossmann, Ravishankar Ramamurthy, and Ramarathnam Venkatesan. Orthogonal security with Cipherbase. In *CIDR*, 2013.

[4] Mihir Bellare, Thomas Ristenpart, Phillip Rogaway, and Till Stegers. Format-preserving encryption. In *SAC*, 2009.

[5] John Black and Phillip Rogaway. Ciphers with arbitrary finite domains. In *CT-RSA*. 2002.

[6] Alexandra Boldyreva, Nathan Chenette, Younho Lee, and Adam O'Neill. Order-preserving symmetric encryption. In *EUROCRYPT*. 2009.

[7] Dan Boneh, Kevin Lewi, Mariana Raykova, Amit Sahai, Mark Zhandry, and Joe Zimmerman. Semantically secure order-revealing encryption: Multi-input functional encryption without obfuscation. In *Eurocrypt*. 2015.

[8] David Cash, Paul Grubbs, Jason Perry, and Thomas Ristenpart. Leakage-abuse attacks against searchable encryption. In *CCS*, 2015.

[9] Ellick Chan, Peifung Lam, and John C. Mitchell. Understanding the challenges with medical data segmentation for privacy. In *USENIX HealthTech*, 2013.

[10] Chandra R Chegireddy and Horst W Hamacher. Algorithms for finding k-best perfect matchings. *Discrete Applied Mathematics*, 1987.

[11] Nathan Chenette, Kevin Lewi, Stephen A Weis, and David J Wu. Practical order-revealing encryption with limited leakage. In *FSE*, 2016.

[12] Siddhartha Chib and Edward Greenberg. Understanding the Metropolis-Hastings algorithm. *The American Statistician*, 1995.

[13] Ciphercloud. Ciphercloud. `http://www.ciphercloud.com/`, 2016.

[14] CNN. Yahoo data breach. `http://money.cnn.com/2016/09/22/technology/yahoo-data-breach/`, 2016.

[15] Tore Dalenius. Towards a methodology for statistical disclosure control. *Statistik Tidskrift*, 15, 1977.

[16] Tore Dalenius. Finding a needle in a haystack or Identifying anonymous Census records. *J. Offic. Stat.*, 2(3), 1986.

[17] datamade. USAddress library. `https://github.com/datamade/usaddress`, 2016.

[18] Harry S. Delugach and Thomas H. Hinke. Wizard: a database inference analysis and detection system. *IEEE Transactions on Knowledge and Data Engineering*, 1996.

[19] Pennsylvania State Department. PA voters. `http://www.dos.pa.gov/votingelections/pages/default.aspx`, 2016.

[20] Josep Domingo-Ferrer. A survey of inference control methods for privacy-preserving data mining. `http://vneumann.etse.urv.es/webCrises/publications/bcpi/domingo-ferrer_survey_SDC_corrected.pdf`, 2008.

[21] Josep Domingo-Ferrer. Inference control in statistical databases. In *Encyclopedia of Database Systems*, 2009.

[22] F. Betül Durak, Thomas M. DuBuisson, and David Cash. What else is revealed by order-revealing encryption? In *CCS*, 2016.

[23] Cynthia Dwork. Differential privacy. In *ICALP*, 2006.

[24] Kevin P. Dyer, Scott E. Coull, Thomas Ristenpart, and Thomas Shrimpton. Protocol misidentification made easy with format-transforming encryption. In *CCS*, 2013.

[25] Michael Egorov and MacLane Wilkison. ZeroDB white paper. *CoRR*, abs/1602.07168, 2016.

[26] eWeek. Navajo Systems. `http://tinyurl.com/y85obds6`, 2009.

[27] Sky Faber, Stanislaw Jarecki, Hugo Krawczyk, Quan Nguyen, Marcel-Catalin Rosu, and Michael Steiner. Rich queries on encrypted data: beyond exact matches. In *ESORICS*, 2015.

[28] Csilla Farkas and Sushil Jajodia. The inference problem: a survey. *ACM SIGKDD Explorations Newsletter*, 2002.

[29] Ivan Fellegi and Alan Sunter. A theory for record linkage. *J. Am. Stat. Assoc.*, 64(328), 1969.

[30] FOP. About Michigan FOP. `http://www.mifop.com/?zone=/unionactive/view_page.cfm&page=About20MIFOP`, 2017.

[31] FOP. Colorado state local FOP lodges. `http://www.coloradofop.org/?zone=/unionactive/private_view_page.cfm&page=Local20Lodge20Directory`, 2017.

[32] FOP. Florida FOP lodge. `http://www.floridastatefop.org/howjoin.asp`, 2017.

[33] FOP. North Carolina FOP lodge 69. `http://www.ncfop69.org/about-us/`, 2017.

[34] FOP. Oklahoma FOP lodge. `http://www.okfop.org/index.cfm?zone=/unionactive/view_page.cfm&page=About20us`, 2017.

[35] FOP. Pennsylvania FOP. `http://www.pafop.org/`, 2017.

[36] Benny Fuhry, Walter Tighzert, and Florian Kerschbaum. Encrypting analytical web applications. In *CCSW*, 2016.

[37] Komei Fukuda and Tomomi Matsui. Finding all the perfect matchings in bipartite graphs. *Applied Mathematics Letters*, 1994.

[38] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. In *FOCS*, 1984.

[39] Melissa Goldstein and Alison Rein. Data segmentation in electronic health information exchange: policy considerations and analysis. The George Washington University Medical Center, 2010.

[40] Paul Grubbs, Richard McPherson, Muhammad Naveed, Thomas Ristenpart, and Vitaly Shmatikov. Breaking web applications built on top of encrypted data. In *CCS*, 2016.

[41] Paul Grubbs, Thomas Ristenpart, and Vitaly Shmatikov. Why your encrypted database is not secure. In *HotOS*, 2017.

[42] Paul Grubbs, Kevin Sekniqi, Vincent Bindschaedler, Muhammad Naveed, and Thomas Ristenpart. Leakage-abuse attacks against order-revealing encryption. In *S&P*, 2017.

[43] Carl Gunter, Mike Berry, and Martin French. Decision support for data segmentation (DS2): application to pull architectures for HIE. In *USENIX HealthTech*, 2014.

[44] Leonid Gurvits. On the complexity of mixed discriminants and related problems. In *International Symposium on Mathematical Foundations of Computer Science*, 2005.

[45] Shai Halevi and Phillip Rogaway. A tweakable enciphering mode. In *CRYPTO*. 2003.

[46] Shai Halevi and Phillip Rogaway. A parallelizable enciphering mode. In *CT-RSA*. 2004.

[47] Warren He, Devdatta Akhawe, Sumeet Jain, Elaine Shi, and Dawn Song. ShadowCrypt: encrypted web applications for everyone. In *CCS*, 2014.

[48] Healthcare Cost and Utilization Project (HCUP). HCUP nationwide inpatient sample (NIS). `https://www.hcup-us.ahrq.gov/nisoverview.jsp`, 2008.

[49] Healthcare Cost and Utilization Project (HCUP). HCUP clinical classifications software (CCS) for ICD-9-CM. `https://www.hcup-us.ahrq.gov/toolssoftware/ccs/ccs.jsp`, 2016.

[50] Thomas H Hinke. Inference aggregation detection in database management systems. In *S&P*, 1988.

[51] IQrypt. IQrypt: Encrypt and query your database. `http://www.iqrypt.com/`, 2016.

[52] MS Islam, Mehmet Kuzu, and Murat Kantarcioglu. Access pattern disclosure on searchable encryption: ramification, attack and mitigation. In *NDSS*, 2012.

[53] Mark Jerrum, Alistair Sinclair, and Eric Vigoda. A polynomial-time approximation algorithm for the permanent of a matrix with nonnegative entries. *JACM*, 2004.

[54] Sriram Keelveedhi, Mihir Bellare, and Thomas Ristenpart. DupLESS: Server-aided encryption for deduplicated storage. In *USENIX Security*, 2013.

[55] Georgios Kellaris, George Kollios, Kobbi Nissim, and Adam O'Neill. Generic attacks on secure outsourced databases. In *CCS*, 2016.

[56] Jeremy Kepner, Vijay Gadepally, Pete Michaleas, Nabil Schear, Mayank Varia, Arkady Yerukhimovich, and Robert K Cunningham. Computing on masked data: a high performance method for improving big data veracity. In *HPEC*, 2014.

[57] Florian Kerschbaum. Frequency-hiding order-preserving encryption. In *CCS*, 2015.

[58] Florian Kerschbaum and Axel Schröpfer. Optimal average-complexity ideal-security order-preserving encryption. In *CCS*, 2014.

[59] Marie-Sarah Lacharité, Brice Minaud, and Kenneth G. Paterson. Improved reconstruction attacks on encrypted data using range query leakage. IACR ePrint, 2017. `http://eprint.iacr.org/2017/701`.

[60] Marie-Sarah Lacharité and Kenneth G Paterson. A note on the optimality of frequency analysis vs. ℓ_p-optimization. `http://eprint.iacr.org/2015/1158.pdf`, 2015.

[61] Billy Lau, Simon Chung, Chengyu Song, Yeongjin Jang, Wenke Lee, and Alexandra Boldyreva. Mimesis Aegis: a mimicry privacy shield. In *USENIX Security*, 2014.

[62] Kevin Lewi and David J Wu. Order-revealing encryption: new constructions, applications, and lower bounds. In *CCS*, 2016.

[63] Daniel Luchaup, Kevin P. Dyer, Somesh Jha, Thomas Ristenpart, and Thomas Shrimpton. LibFTE: A user-friendly toolkit for constructing practical format-abiding encryption schemes. In *USENIX Security*, 2014.

[64] Daniel Luchaup, Thomas Shrimpton, Thomas Ristenpart, and Somesh Jha. Formatted encryption beyond regular languages. In *CCS*, 2014.

[65] Teresa F Lunt. Aggregation and inference: facts and fallacies. In *S&P*, 1989.

[66] Microsoft. Always Encrypted (Database Engine). `https://msdn.microsoft.com/en-us/library/mt163865.aspx`, 2016.

[67] Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large sparse datasets. In *S&P*, 2008.

[68] Muhammad Naveed, Seny Kamara, and Charles V Wright. Inference attacks on property-preserving encrypted databases. In *CCS*, 2015.

[69] Skyhigh Networks. Skyhigh Networks. `https://www.skyhighnetworks.com/`, 2017.

[70] H. Newcombe, J. Kennedy, S. Axford, and A. James. Automatic linkage of vital records. *Science*, 130(3381), 1959.

[71] State of Ohio. Ohio voters. `http://ohiovoters.info/`, 2016.

[72] Office of the Chief Privacy Officer. The data segmentation for privacy initiative. `https://www.healthit.gov/providers-professionals/ds4p-initiative`, 2016.

[73] Office of the National Coordinator. HL7 implementation guide for DS4P initiative. `https://www.hl7.org/implement/standards/product_brief.cfm?product_id=354`, 2016.

[74] Antonis Papadimitriou, Ranjita Bhagwan, Nishanth Chandran, Ramachandran Ramjee, Andreas Haeberlen, Harmeet Singh, Abhishek Modi, and Saikrishna Badrinarayanan. Big data analytics over encrypted datasets with seabed. In *Proceedings of the 12th USENIX conference on Operating Systems Design and Implementation*, pages 587–602. USENIX Association, 2016.

[75] Vasilis Pappas, Fernando Krell, Binh Vo, Vladimir Kolesnikov, Tal Malkin, Seung Geol Choi, Wesley George, Angelos D. Keromytis, and Steve Bellovin. Blind Seer: A scalable private DBMS. In *S&P*, 2014.

[76] Perspecys. Perspecsys: A Blue Coat Company. `http://perspecsys.com/`, 2016.

[77] Rishabh Poddar, Tobias Boelter, and Raluca Popa. Arx: A strongly encrypted database system. `https://eprint.iacr.org/2016/591`, 2016.

[78] Raluca Popa, Frank H Li, and Nickolai Zeldovich. An ideal-security protocol for order-preserving encoding. In *S&P*, 2013.

[79] Raluca Popa, Catherine Redfield, Nickolai Zeldovich, and Hari Balakrishnan. CryptDB: Protecting confidentiality with encrypted query processing. In *SOSP*, 2011.

[80] Raluca Popa, Emily Stark, Steven Valdez, Jonas Helfer, Nickolai Zeldovich, and Hari Balakrishnan. Building web applications on top of encrypted data using Mylar. In *NSDI*, 2014.

[81] Raluca Popa, Nickolai Zeldovich, and Hari Balakrishnan. Guidelines for using the CryptDB system securely. `https://eprint.iacr.org/2015/979`, 2015.

[82] David Pouliot and Charles V Wright. The Shadow Nemesis: inference attacks on efficiently deployable, efficiently searchable encryption. In *CCS*, 2016.

[83] Timothy RC Read and Noel AC Cressie. *Goodness-of-fit statistics for discrete multivariate data.* Springer Science & Business Media, 2012.

[84] Daniel S. Roche, Daniel Apon, Seung Geol Choi, and Arkady Yerukhimovich. POPE: partial order preserving encoding. In *CCS*, 2016.

[85] Phillip Rogaway and Thomas Shrimpton. The SIV mode of operation for deterministic authenticated-encryption (key wrap) and misuse-resistant nonce-based authenticated-encryption. `http://web.cs.ucdavis.edu/~rogaway/papers/siv.pdf`, 2007.

[86] Andreas Schaad. SAP SEEED Project. `https://www.sics.se/sites/default/files/pub/andreasschaad.pdf`, 2016.

[87] J. Schlörer. Identification and retrieval of personal records from a statistical data bank. *Methods Inf. Med.*, 14(1), 1975.

[88] Latanya Sweeney. Weaving technology and policy together to maintain confidentiality. *J. of Law, Medicine and Ethics*, 25, 1997.

[89] Stephen Tu, M Frans Kaashoek, Samuel Madden, and Nickolai Zeldovich. Processing analytical queries over encrypted data. In *VLDB*, 2013.

[90] US Census Bureau. Age and sex composition. http://www.census.gov/prod/cen2010/briefs/c2010br-03.pdf, 2010.

[91] US Census Bureau. American community survey (ACS). http://www.census.gov/programs-surveys/acs/, 2016.

[92] USPS. ZIP code database. http://www.unitedstateszipcodes.org/zip-code-database/, 2016.

[93] Gregory Valiant and Paul Valiant. Instance optimal learning of discrete distributions. In *STOC*, 2016.

[94] Wikipedia. Sony pictures entertainment hack. https://en.wikipedia.org/wiki/Sony_Pictures_Entertainment_hack, 2017.

[95] Wikipedia. Target customer privacy. https://en.wikipedia.org/wiki/Target_Corporation#Customer_privacy, 2017.

[96] Leon Willenborg and Ton de Waal. *Statistical Disclosure Control in Practice*. Springer-Verlag, 1996.

[97] Leon Willenborg and Ton de Waal. *Elements of Statistical Disclosure Control*. Springer-Verlag, 2001.

[98] Wired. RateMyCop user ensnared in dumbest case ever. https://www.wired.com/2010/06/dumbest-case-ever/, 2010.

[99] Andrew C Yao. Protocols for secure computations. In *FOCS*, 1982.

[100] Yupeng Zhang, Jonathan Katz, and Babis Papamanthou. All your queries are belong to us: the power of file injection attacks. In *USENIX Security*, 2016.

[101] Wenting Zheng, Ankur Dave, Jethro G Beekman, Raluca Popa, Joseph E Gonzalez, and Ion Stoica. Opaque: An oblivious and encrypted distributed analytics platform. In *NSDI*, 2017.

A Simultaneous Multiple Attributes via MCMC

The multinomial attack also provides a way to perform an attack on $k > 1$ columns simultaneously by considering a single (unknown) encryption function f over all k columns. In this case, the inference formulation is the same as in Equation 1, with the following differences: (i) f is a function taking k inputs and producing k outputs, (ii) $\vec{c}$ is a vector of k-tuples, and (iii) ρ is the joint distribution of k columns.

The likelihood term $\Pr\{\vec{c} \mid \mathbf{f} = f \ ; \rho\}$ can be modeled as a multinomial distribution over k-tuples such that $\Pr\{\vec{c} \mid \mathbf{f} = f \ ; \rho\} = K_c \prod_i \rho_i^{c_{f(i)}}$. This looks strikingly like Equation 3 (the multinomial likelihood for the single-column case) but here the product is over all possible tuples of plaintexts. In fact, the mapping f maps all plaintext k-tuples (auxiliary) to all ciphertexts k-tuples (observed). However, not all mappings are valid, some violate the constraint that (over a column) a given plaintext value always maps to the same ciphertext.

If we can enumerate all valid mappings, then $f_{\max}$ and the full posterior distribution can be computed through Equation 1. If not, we can use Markov-chain Monte Carlo (MCMC) techniques, e.g., Metropolis-Hastings [12]. We sketch the idea here and leave its full treatment and evaluation to future work.

The Metropolis-Hastings algorithm works by creating a Markov chain with a stationary distribution equal to a desired probability distribution, which for us is the posterior of f (Equation 1). To use the algorithm, we need a concept of neighboring mappings which is formally captured by a proposal distribution. We can let two mappings be neighbors if one mapping can be obtained from the other by changing an edge in the projected bipartite graph for a particular column (for ORE) or switching two edges (for DE).

Informally, the algorithm keeps a state (current mapping) f and iteratively explores the space by repeatedly (1) selecting a candidate neighbor mapping f_c using the proposal distribution, and (2) updating the current state to the candidate mapping (i.e., $f \leftarrow f_c$) according to a probabilistic rule.

The probabilistic rule requires computing the acceptance ratio $\alpha = \frac{p(f_c)}{p(f)}$. If $\alpha \geq 1$, then we unconditionally accept f_c as our new mapping, else we accept f_c with probability α. Here α is exactly the posterior ratio.

To approximate $f_{\max}$ in this case, we only need to keep track of the best mapping f_{best} found so far. We proceed using the acceptance ratio α as follows. If $\alpha \leq 1$, there is nothing to do because f_c is no more probable than the current mapping, therefore it must be less probable than f_{best}. If $\alpha > 1$, then f_{best} is updated to f_c whenever $p(f_c)/p(f_{\text{best}}) > 1$.

If we want to approximate the posterior, we can do it via sampling. After some burn-in period, the Markov chain will converge towards its stationary distribution. From then on, we take the current mapping as a sample every t steps, for some integer t. (The gap t helps prevent taking correlated samples because we need i.i.d. samples.)

The algorithm must start with some initial state (i.e., mapping f). We propose to use a single-column attack on each of the k columns independently and then use the resulting collection of single-column mappings as initial state. Alternatively, it is possible to use the dependency DAG approach as an initial mapping and then look for an overall better solution using Metropolis-Hastings.

B A Note on the L1 Attack

The mapping returned by the L1 minimum-cost assignment attack of Naveed et al. [68] is not always the same as that of frequency analysis (or the multinomial attack). This observation has also been made in other contexts such as [93], which noted that using frequency analysis (though the authors did not call it that) for labeling samples from a discrete distribution does not minimize the L1 distance between the labelled empirical distribution and the true distribution. One way to understand the difference is by comparing mappings that are equally good in terms of the L1 attack but have different likelihoods.[3]

Consider the following example of a column with 3 possible plaintexts. Let $\rho = (3/4, 3/20, 1/10)$ and $\vec{c} = (13, 4, 3)$. If we re-scale ρ to correspond to a total of $n = 20$ observations, we get $n\rho = (15, 3, 2)$. Clearly, the mapping $f_I = f(i) = i$ has the lowest sum of L1 frequency differences, i.e., $\sum_{i=1}^{3} |c_i - n\rho_i| = 2+1+1 = 4$. Note that this is also the mapping returned by frequency analysis (or the multinomial attack). On the other hand, the mapping $f_{1,3,2} = f$ such that $f(1) = 1$ but $f(2) = 3$ and $f(3) = 2$ also has cost 4. However, from the point of view of frequency analysis, the mapping f_I is more likely than $f_{1,3,2}$. Furthermore, the weight of mapping f_I using the multinomial attack is $w_{f_I} = 13 \log 3/4 + 4 \log 3/20 + 2 \log 1/10 > 13 \log 3/4 + 4 \log 1/10 + 3 \log 3/20 = w_{f_{1,3,2}}$.

[3]Another explicit counterexample was given in [93].