Published in final edited form as:

IEEE Trans Knowl Data Eng. 2014 December ; 26(12): 2956–2968. doi:10.1109/TKDE.2013.91.

Composite Bloom Filters for Secure Record Linkage

Elizabeth Ashley Durham,

Dept. of Biomedical Informatics, Vanderbilt University, Nashville, TN 37232.

ea.durham@vanderbilt.edu

Murat Kantarcioglu [Senior Member, IEEE],

Department of Computer Science, University of Texas at Dallas, Richardson, TX, 75083.

muratk@utdallas.edu

Yuan Xue [Member, IEEE],

Dept. of Electrical Engineering & Computer Science, Vanderbilt University, Nashville, TN 37232.

yuan.xue@vanderbilt.edu

Csaba Toth,

Dept. of Biomedical Informatics, Vanderbilt University, Nashville, TN 37232.

csaba.toth@vanderbilt.edu

Mehmet Kuzu, and

Dept. of Computer Science, University of Texas at Dallas, Richardson, TX, 75083.

mehmet.kuzu@utdallas.edu

Bradley Malin [Member, IEEE]

Depts. of Biomedical Informatics and EECS, Vanderbilt University, Nashville, TN 37232.

b.malin@vanderbilt.edu

Abstract

The process of record linkage seeks to integrate instances that correspond to the same entity. Record linkage has traditionally been performed through the comparison of identifying field values (*e.g.*, *Surname*), however, when databases are maintained by disparate organizations, the disclosure of such information can breach the privacy of the corresponding individuals. Various private record linkage (PRL) methods have been developed to obscure such identifiers, but they vary widely in their ability to balance competing goals of accuracy, efficiency and security. The tokenization and hashing of field values into Bloom filters (BF) enables greater linkage accuracy and efficiency than other PRL methods, but the encodings may be compromised through frequency-based cryptanalysis. Our objective is to adapt a BF encoding technique to mitigate such attacks with minimal sacrifices in accuracy and efficiency. To accomplish these goals, we introduce a statistically-informed method to generate BF encodings that integrate bits from multiple fields, the frequencies of which are provably associated with a minimum number of fields. Our method enables a user-specified tradeoff between security and accuracy. We compare our encoding method with other techniques using a public dataset of voter registration records and demonstrate that the increases in security come with only minor losses to accuracy.

Keywords

data matching; record linkage; entity resolution; privacy; security; Bloom filter

1 Introduction

The set of records referring to a specific individual is often distributed across datasets managed by independent organizations. To ensure unbiased analytics upon their integration, it is essential to merge records in a manner that mitigates duplication, as well as fragmentation, of an individual's information. Over the years, various data engineering strategies have been developed to support the integration process, which are collectively referred to as record linkage (*e.g.*, [1]–[3]). In practice, record linkage relies upon the identifying information that typically characterizes an individual, often referred to as fields (*e.g.*, *surname* and *street address*). However, various regulations and policies prohibit the disclosure of identifiers because the sharing of such information can violate the privacy of individuals. As such, over the past decade, private record linkage (PRL) approaches have been proposed to enable data holders to integrate records without revealing the raw information [4].

PRL is a diverse notion and is characterized by several classes of approaches. One class is based upon secure multiparty computation (SMC), which leverages cryptographically strong protocols¹. These protocols enable similarity comparisons to be performed without revealing *any* information, save for the input and output of the protocol [5]. While secure in principle, the computationally expensive encryption techniques required to achieve SMC do not scale well to very large databases [6]. This is because record linkage approaches are fundamentally statistical processes that require model development and subsequent evaluations of model fitness [7].

As an alternative, a second class of PRL, which we focus on in this work, utilizes a weaker form of security based on data transformation. These transformed values, referred to as encodings, are used as inputs to PRL. It is critical to strike a balance amongst the competing priorities of accuracy, security, and efficiency when identifying an appropriate data transformation method for PRL. A recent evaluation [6] showed that a transformation based on encoding values in a Bloom filter (BF) [8] is superior to other approaches (*e.g.*, unique hashed [9] or obfuscated [10] identifiers, SMC-based equijoins [11], and SMC-based similarity comparisons [12]) in terms of efficiency and accuracy. This method, which we refer to as *field-level BF* (FBF) encoding, maps each field value in a record into a corresponding BF.

There are concerns, however, that FBFs may be vulnerable to frequency-based attacks [13]. It has been suggested that BF-based encoding techniques can be hardened by combining FBFs into a single BF for the entire record [14]. The intuition behind such a strategy is that cryptanalysis on the composite data structure will be more difficult because field values are

¹We refer to human readable data as plaintexts and encrypted or enciphered data as ciphertexts.

considered in combination rather than independently. It has yet to be determined, however, how such a technique can be refined into a principled methodology. In this paper, we show how such a strategy can be statistically informed through the discriminatory power of FBFs and can be made quantifiably robust against frequency-based attacks. We refer to our solution as a *record-level BF* (RBF) encoding.

The proposed RBF generation method uses a tunable parameter that determines the computational complexity required to map a bit from an RBF back to the FBF from which it was drawn. This approach is data-driven and can be tailored to any dataset to achieve the desired level of accuracy and security. We demonstrate this approach on a large, publicly-available dataset and show that it maintains the desirable properties of the initially proposed BF-based encoding [8] while significantly improving its robustness to frequency-based attacks.

1.1 Contributions

The contributions of this work are:

1. **Enhanced security:** Our encoding method generates RBFs from FBF encodings via a data-driven bit selection procedure. This encoding utilizes a tunable security parameter with quantifiable resistance to frequency-based cryptanalysis attacks [13].
2. **Top Rank Preserving:** The resulting RBFs provide a transformation from the plaintext space to the ciphertext space, such that the nearest neighbor to record is retained with a high likelihood. This paves the way for the application of PRL in the ciphertext space in a manner that maintains a high degree of accuracy.
3. **Empirical Evaluation:** We perform an evaluation of the RBF strategy with several competing approaches using a dataset of personal identifiers derived from a real voter list. We use statistical hypothesis testing to demonstrate that the RBF strategy provides better top rank preservation than its competitors.

1.2 Outline

The remainder of this paper is organized as follows. In Section 2, we summarize the FBF generation methodology as proposed in the literature and its known vulnerabilities. We also describe the dataset used in this research, so that it may be used in illustrative examples throughout this paper. Section 3 then describes our RBF encoding method. In Section 4, we describe the experimental framework for the evaluation presented in Section 5. In Section 6, we discuss the practical implications of these results, limitations, and future work. Section 7 discusses related research and Section 8 summarizes the work.

2 Background

Section 2.1 introduces the notation and framework used in this research. In Section 2.2, we provide an overview of the FBF encoding process, and discuss security flaws of FBFs in Section 2.3. Finally, Section 2.4 describes the dataset utilized to illustrate our methodology.

2.1 Notation & Framework

In this work, a record is comprised of f fields, and records to be linked are contributed by two database holders, referred to as Alice and Bob. A denotes the dataset of records provided by Alice, $|A|$ is the number of records, $a_x[i]$ indicates the x^{th} record within set A , and $a_x[i]$ refers to the value of the i^{th} field in record a_x , where $i \in \{1, \dots, f\}$. Similarly, B , $|B|$, $b_y[i]$, and correspond to the set of records provided by Bob.

In the PRL process, Alice and Bob locally encode their records and send the encodings to a third party, Charlie, where they are compared. The goal is to enable Charlie to correctly classify all record pairs (i.e., $A \times B$) into the classes M (*Predicted Matches*) and U (*Predicted Non-matches*) such that the record pairs in M refer to the same individual (i.e., *True Matches*) and the record pairs in U refer to different individuals (i.e., *True Non-matches*).

2.2 Field-level Bloom Filter Encodings

A recent evaluation [6] of PRL transformation strategies shows that FBF encodings [8] provide highly accurate linkage results in a reasonable running time. The specific steps in FBF encoding are:

- Each field value is treated as a string and decomposed into a set of tokens of length n , or n -grams. To account for the first and last characters, the value is padded with $n - 1$ blank characters on each end.
- Each n -gram is hashed into an FBF of length m bits using k hash functions. We use m_{FBF} (m_{RBF}) to represent the number of bits in an FBF (RBF).

The encodings can be compared using a set-based similarity measure, such as the Dice coefficient [8]:

$$\text{Dice}(\alpha, \beta) = \left(\frac{|\alpha \cap \beta|}{|\alpha| + |\beta|} \right), \quad (1)$$

where α and β are FBF encodings contributed by Alice and Bob, respectively. Figure 1 illustrates the construction of FBFs for three fields, each with two hash functions.

2.3 Security Flaw of FBFs

The FBF encodings reveal frequency distributions for field values (e.g., specific names) and bit positions (i.e., n -grams). Given that frequency-based attacks have been designed to compromise simple substitution ciphers (e.g., [15]), it is critical to ask “How much knowledge can Charlie² infer about the plaintext from the FBFs?” To investigate, [13] designed a cryptanalytic attack based on frequency analysis. Here, we summarize the cryptanalysis (with further details in the Appendix) to clarify how RBFs provide protection from such an attack.

²We use Charlie to refer to the attacker, but it could be any entity that observes plaintext values.

In short, Charlie observes the frequency with which bits are set in the FBF encodings and attempts to map bits back to the n -grams to which they correspond. It was shown that, for parameter settings recommended in the literature [8] (i.e., $n = 2$, $m_{\text{FBF}} = 500$, $k = 15$), FBF encodings are vulnerable, such that Charlie can infer some plaintext values. However, it was further shown that as n grows, the likelihood of success diminishes. This is because as n increases, so too does the number of distinct n -grams, which leads to a smaller quantity of frequency-related information to commit a successful cryptanalysis. While this observation suggests a simple strategy to improve security, it was also found that n and accuracy were inversely correlated. This correlation is natural because as the length of the n -grams hashed into the filter increases, it is more difficult to detect small discrepancies between their corresponding strings. Thus, we propose a principled mechanism, independent of n , to increase the security of BF-based encodings.

We note that recent evidence suggests the identified data available to an attacker may not be of the same frequency distribution as that observed in the FBFs [16]. However, it was shown that, even though the distributions may not be equivalent, an attacker can allow for weaker constraints in their cryptanalysis and commit a successful cracking of the system. Given that it is not always possible to determine what data sources an adversary may have access to, we account for the worst case scenario (i.e., when the hashed records and information available to the adversary are derived from the same resource), and thus ensure a system is robust against a constraint satisfaction-based cryptanalysis.

2.4 Dataset

We evaluate our methods with the publicly available North Carolina voter registration (NCVR) database [17]. We utilize the following fields: *Forename*, *Surname*, *City*, *Street address*, *Ethnicity*, and *Gender*. This dataset contains 6,190,504 individual records.

We use NCVR as dataset *A* and generate a dataset *B* by corrupting each record in *A*. To do so, we extended the “data corrupter” implemented in the Febrl [18] toolkit, which is based on [19]. Like the Febrl model, we introduced character-level errors at rates consistent with those reported in real datasets. These include optical character recognition (OCR) errors (e.g., *S* swapped 8), phonetic errors (e.g., *ph* swapped for *f*), and typographic errors, including insertions deletions, transpositions, and substitutions. For our experiments, the probability of introducing these errors was set as follows: insertions and deletions - 0.15, phonetic errors - 0.03, OCR errors - 0.01, substitutions - 0.35, and transpositions - 0.05. This corruption is performed for all of the aforementioned fields, including gender, to assess how PRL performs without any preprocessing or cleaning of records.

Our extension to the Febrl corruptor introduces errors at the value-level, such as the use of a nickname or a change in residential address. First, *Nicknames* were substituted for *Forenames* (e.g., *Alex* swapped for *Alexandra*) with probability 0.15. Nicknames were based on the Massmind Nicknames Database [20]. Second, *Surnames* were substituted for females (to simulate change of name due to marriage) with probability 0.1, for males with probability 0.01, and were hyphenated with probability 0.01. *Surnames* were based on the 2000 U.S. Census names data [21]. Third, *Street addresses* were changed with probability

0.1. These were randomly selected from the NCVR dataset. We refer the reader to [22] for further details of these datasets and examples.

The record linkage experiments were based on subsets of A and B . Datasets created in this manner eliminate the need for the extensive manual curation required to determine the true match status for independent datasets. We recognize that a “true” record linkage dataset – where the datasets are constructed independently – would be ideal [23], but believe this approach is appropriate for benchmarking and testing new ideas.

For the experiments reported in this paper, we use ten independent samples of 100,000 records from the NCVR dataset. We exclude records with missing values to obtain a clear idea of the performance of each method.

3 Record-level Bloom Filters

The proposed method builds upon the strengths of the FBF encodings, such as their high accuracy and expedient running time, while increasing resistance to frequency-based cryptanalysis. It is anticipated that the composite RBF encoding will provide stronger resistance against frequency analysis and therefore greater security. One of the contributions of our research is to demonstrate how to compose a such a composite data structure with quantifiable security guarantees. For now, we describe, at a high level, how to create this RBF encoding. We compare our proposed strategy and that in [14] using an empirical and syntactic analysis in the Experiments and Related Work sections, respectively.

The general architecture for the proposed approach is illustrated in Figure 1. In the remainder of this section, we discuss design decisions that guide RBF generation:

1. **FBF parameterization and generation.** FBF parameters should be selected to maximize security.
2. **Eligible bit identification.** Bits sampled from the FBFs should be chosen to maximize security.
3. **Field weighting.** A scheme is applied to determine *how many* bits should be sampled from each FBF.
4. **RBF parameterization & generation.** RBF parameters are established and bits are sampled from the FBFs resulting in an RBF encoding.
5. **RBF permutation.** Bits are randomly ordered using a permutation agreed upon by Alice and Bob.

Figure 1 illustrates how strings in multiple fields (*Fore-name*, *Surname*, and *City*) are mixed into a single RBF. As noted earlier, the strings are initially split into their respective n -grams, which are hashed into FBFs. These FBFs are then weighted according to their discriminatory power for record linkage. Based on this weighting, *Forename* and *City* will each contribute 25% of the bits to the RBF, while *Surname* will contribute 50% of the bits. The RBF is then composed of bits sampled from each FBF and the resulting bit locations are finally permuted.

3.1 FBF Parameterization & Generation

In this section, we address how FBF parameterization affects the security of the resultant encodings.

In cryptanalysis, Charlie attempts to map bits in the RBF to the fields from which they were drawn. When successful, Charlie can mount an attack such as the one highlighted in the Appendix, using the FBFs as inputs. Therefore, evaluating the extent to which Charlie can map bits in the RBF to the field from which they were drawn is important to characterize the security of the RBF. In particular, we consider how much information Charlie can gather by modeling the frequency with which bits are set under two FBF parameterization methods: 1) *static* and 2) *dynamic* FBF sizing.

3.1.1 Static FBF Sizing—In the static FBF sizing strategy, the k and m values are held constant across all fields, regardless of the expected number of n -grams to be encoded in the field. In this case, observation of the percentage of bits set to 1 may allow Charlie to determine some information about the length of the encoded value and the number of hash functions used in encoding.

Shorter fields, such as *Gender*, result in FBF encodings with fewer bits set to 1 than longer fields, such as *Forename*. Consider an example where $n = 2$, $k = 15$, and $m = 500$. *Gender*, encoded “M” or “F”, will contain only two n -grams for each field value. Therefore, for each gender encoding, a maximum of 30 bits can be set (15 per n -gram), which corresponds to 6% of the bits in the FBF. However, if *Forename* has an average of 6 letters per record, this corresponds to 7 n -grams and a maximum of 105 bits can be set, which corresponds to 21% of the bits in the FBF. This variability in the average number of bits set across fields in statically sized FBFs is depicted in Figure 2.

As a consequence, when static parameters are applied to each field, Charlie can observe the percentage of bits set in a FBF. With this information, Charlie can infer the length of the value encoded in the filter and the number of hash functions used in encoding.

3.1.2 Dynamic FBF Sizing—We anticipate that tailoring the encoding parameters to the specific properties of each field will limit the extent to which Charlie can determine the field from which a bit was drawn. Therefore, we propose dynamically sizing the Bloom filters (*i.e.*, setting the m_{FBF} value) such that the same percentage of bits are expected to be set to 1 in the FBFs of each field³. Additionally, we let the expected frequency at which a given bit is set in the Bloom filter encoding to be equal to 0.5. The justification for this choice is that in order to maximize entropy, and therefore maximize security, half of the bits should be set [24]. In this case, Charlie would observe that approximately half of the bits are set to 1 and would gain little information about which field values are encoded in the filter.

More formally, let the expected frequency at which a bit is set p be equal to 0.5 and let g be the number of items (*e.g.*, n -grams) stored in the FBF. The probability that a certain bit remains unset at a value of 0 is $p=1 - (\frac{1}{m})^{kg}$ [24]. Therefore, if we select a value for k , and

³Due to collisions, only the expected frequency at which a bit is set can be controlled.

hold it constant, we can calculate the number of bits needed in the FBF according by solving for m in the previous equation. This yields:

$$m = \frac{1}{1 - \frac{kq}{\sqrt{p}}} \quad (2)$$

We can then measure the information content, as indicated by average field value length, associated with each field and adjust the FBF length accordingly.

The values of m_{FBF} (i.e., number of bits in each FBF) for the NCVR dataset are shown in Table 1. Additionally, the average number of bits set across fields in dynamically sized FBFs is shown in Figure 2. It can be seen that the FBFs of *dynamic* size have a more uniform average frequency across fields than the FBFs of *static* size. This suggests that it is more difficult for an attacker to determine from which FBF a bit was drawn when the FBFs are dynamically sized. We further explore this issue, and introduce additional measures to increase security, in Section 3.2.

3.2 Eligible Bit Identification

In this section, we introduce a statistically-informed method for determining which bits should be sampled from each FBF so that the resulting RBFs are resistant to frequency analysis.⁴

When Charlie attempts to determine the plaintext values encoded in an RBF, a first step is likely to map RBF bits back to the fields from which they were drawn, based on the frequency with which a bit is set. This is particularly true if the frequency distributions of the bits within fields is distinctive across fields. For example, we expect that the FBF encodings corresponding to fields having a small domain, such as *Gender* and *Ethnicity*, will have a frequency distribution that is markedly different from fields having a large domain, such as *Surname*. Therefore, it is possible that the frequency at which a given bit in the RBF is set may betray from which FBF it was drawn. Imagine that the only field containing a bit set exactly 50% of the time corresponds to *Gender*. Then, when Charlie observes this frequency for a particular bit, he can conclude it encodes information about *Gender*.

We now look at more detailed information about the frequency with which each bit is set. For illustration, Figure 3 shows a heat map of the frequencies with which bits are set across all FBFs of both *static* and *dynamic* size. This figure suggests that the frequency distributions for categorical fields, such as *Gender*, (Figures 3(e) and 3(f)) are less uniformly distributed than for string-based fields (Figures 3(a) - 3(d)). Due to the fact that categorical variables can take on a limited number of values, many of the bits in the FBF are never set. Those bits that are set are relatively frequent because many individuals in the population have the corresponding value.

⁴The results of this process must be coordinated, so that Alice and Bob select the same FBF bit locations to compose their RBFs. If Alice and Bob have data on reasonably similar populations, then either Alice or Bob could execute this process on their dataset and communicate the result to the other party. In the event that the populations are anticipated to be significantly different, the protocol may require the assistance of an additional third party.

In general, the evidence illustrates that FBFs of *dynamic* size are more uniformly distributed than the FBFs of *static* size. This provides further evidence that an attacker would have greater difficulty using this frequency information in an attack. However, Figure 3 indicates that there is still information Charlie can exploit. Specifically, the frequency at which bits are set is not uniform. By breaking down the frequencies into ranges, Charlie may map bits in the RBF to the FBF from which they were drawn. Figure 4 illustrates how this could occur. In this figure, the x -axis depicts the frequency range, divided by hundredths. The y -axis depicts the number, as well as identity, of the FBFs containing bits in each range. Notice that bits with frequency range $[0.78, 0.79]$ in *dynamic* FBFs reveal that such bits must have been drawn from the *City* FBF (see Figure 4(b)).

To limit the extent to which Charlie can map RBF bits back to the FBFs from which they were drawn, we introduce a security constraint q . This constraint states that only bits that can be mapped back to at least q fields, based on an analysis of the frequency at which bits are set, can be included in the RBF. For example, it is clear that bits which can only be mapped back to a single field, such as bits with frequency range $[0.78, 0.79]$ in *dynamic* FBFs, should be avoided because these bits could only have been drawn from the field *City* (see Figure 4(b)).⁵

By avoiding bits set at frequencies that are unique to a single field, or a small number of fields, the resultant RBF encodings are more secure. This is because it becomes more difficult for an attacker to leverage frequency information to map RBF bits back to the FBFs from which they were drawn. The data holders can establish a security constraint q by allowing only bits that can be mapped back to q fields, by leveraging frequency information, to be included in the RBF. Therefore, a higher value of q implies additional computations required to compromise RBF encodings and thus greater security.

3.3 Field Weighting

We now describe a method for determining the percentage of bits in the RBF that should be sampled from each FBF, which we call the weight attributed to the field. Our field weighting mechanism is based on the discriminatory power of the field (i.e., the extent to which it supports linkage). We hypothesize that sampling bits according to a weighting based on the discriminatory power of each field will provide RBFs that facilitate highly accurate record linkage. For example, *Surname* is intuitively more useful in resolving identity than *Gender*, so it is natural to devise a method that samples more bits from the FBF corresponding to *Surname*.

If Alice and Bob know the discriminatory power of each field, they can weight each field accordingly. In the absence of this knowledge, the weights can be estimated using a probabilistic approach. One such method is based on the work of Fellegi and Sunter [25], [26], which we call the FS field weighting approach. This method computes an agreement weight w_a and a disagreement weight w_d for each field based on the conditional probability of field agreement.⁶

⁵Note that the value for q can be made public because it only communicates the complexity of the RBF and not how to attack the bits within the RBF.

In traditional record linkage applications [25], the agreement and disagreement weights are used to calculate a similarity “score” for each record pair. In this work, we adapt the FS weights to measure the relative discriminatory power of each field. In Equations 3 and 4, we present a mechanism for combining the agreement and disagreement weights into a single weight, $w[i]$, associated with each field i :

$$range[i] = w_a[i] - w_d[i] \quad (3)$$

$$w[i] = \frac{range[i]}{\sum_{i=1}^f range[i]} \quad (4)$$

The weight associated with each field is derived by normalizing the range of a single field by the sum of the ranges of the agreement and disagreement weights over all fields. The resulting weight is a measure of the relative discriminatory power of each field, as compared to other fields. We propose this weight be used to determine the percentage of bits drawn from each FBF for inclusion in the RBF.

The w_a , w_d , $range$, and w values for the dataset used in this work are shown in Table 2. By applying this approach, we sample a varying number of bits from each field. For instance, the *Surname* and *Forename* fields comprise 24% and 22% of the bits in the RBF, respectively.

3.4 RBF Parameterization & Generation

To determine the actual number of bits to be drawn from each FBF, we must select a value for the number of bits to be included in the RBF (i.e., m_{RBF}). To provide a baseline, we assume that all of the eligible bits that satisfy the security constraint from each field should be included in the RBF. Given this assumption and the weight associated with each field, we can calculate the lower bound for m_{RBF} , where each field is included in full and at the relative discriminatory power calculated. We take the maximum of these values to be m_{RBF} .

This computation is illustrated for NCVR in Table 3. In this example, we set $q = 1$. If the 214-bit long FBF for the field *City* is to comprise 16% of the record-level Bloom filter, then m_{RBF} must be 1,315 bits. As this is the maximal value for m_{RBF} required across all fields, this is selected as the length for the RBF. The number of bits to be drawn from each FBF is then selected in accordance with the field weights, as shown in Table 2. The bits are selected at random, with replacement, to construct the RBF. By doing so, the measure of distance between RBFs will be proportional to the contribution of each field.

3.5 RBF Permutation

The bits of the RBF are shuffled according to a random permutation agreed upon by Alice and Bob. This process is akin to the random mixing of bits associated with block ciphers and oblivious transfers in SMC protocols [29]. This permutation should not be made public and

⁶The weights can be approximated through an Expectation Maximization (EM) algorithm to estimate the conditional probabilities associated with each field [23], [26]–[28]. This algorithm could be run by one of the parties over their own dataset.

is considered a “key” used to make the RBF encodings more secure. The shuffling prevents Charlie from determining which FBF the bits in the RBF correspond to, based on their order.

4 Experimental Evaluation

This section provides details of the experimental evaluation. Specifically, Section 4.1 describes the experimental methods evaluated in this work while Section 4.2 describes the control methods. Section 4.3 presents the evaluation measures considered and Section 4.4 provides the implementation details.

4.1 Experimental Methods

In this section, we describe variations of the FBF parameterization (Section 3.1) and field weighting (Section 3.3) strategies applied in our empirical study. It is assumed that Alice and Bob agree upon the parameters for FBF and RBF construction, so that their RBFs are directly comparable. The naming convention for the variations is *FBF parameterization method / field weighting method*. The variations considered in RBF generation are:

1. *Static/Uniform* RBFs. FBFs are sized statically, such that $m_{\text{FBF}} = 500$ and we let $k = 15$ as recommended by Schnell *et al* [8]. Fields are weighted uniformly such that the same number of bits are sampled from each FBF to include in the RBF.
2. *Static/Weighted* RBFs. FBFs are sized statically such that $m_{\text{FBF}} = 500$ and again $k = 15$. Fields are weighted such that the number of bits to be sampled from each field is determined by the discriminatory power of the field.
3. *Dynamic/Uniform* RBFs. FBFs are sized dynamically based on the average field value length. The parameters are shown in Table 1. Fields are weighted uniformly such that the same number of bits are sampled from each FBF to include in the RBF.
4. *Dynamic/Weighted* RBFs. FBFs are sized dynamically based on the average field value length. Fields are weighted such that the number of bits to be sampled from each field is determined by the discriminatory power of the field.

In addition, we compare our method with the *Cryptographic Longterm Key* (CLK) in [14]. In the basic model formalized in this work, all fields are hashed into the equal-sized BFs using the same set of hash functions. The resulting filters are then integrated into a single BF by a union operation.

4.2 Controls

The following field comparison techniques are used as controls for the experimental RBF techniques:

1. *Binary Field Comparison*. Field values are checked for equality, resulting in a binary measure of equivalence (i.e., each field receives a score of 0 or 1).
2. Jaro-Winkler (JW) String Comparator [30] over Plaintext. This comparator was designed to measure the similarity of plaintext strings by looking at the number of characters that agree, the number of transpositions present, and weighting the

characters at the beginning of the string based on the observation that errors are more likely to occur at the end of a string. This comparator has been shown to work well in record linkage. [31], [32].

These controls are applied to determine field similarity. We convert field similarities to record pair similarities by multiplying by either i) a uniform value $\left(\frac{1}{f}\right)$ or ii) the FS weight w values shown in Table 2.

4.3 Evaluation Measures

To determine the extent to which proximity is preserved in the transformation to the encoded space, we developed a measure called the neighborhood test. In our experiments, we sample 1,000 records - 500 from dataset A and 500 from dataset B . Consider an example where $a_x \in A$ is the record of interest and $b_x \in B$ is its *True Match*. Let B' be a version of B which is ordered according to the similarity⁷ of all $b \in B$ to a_x . The neighbor rank of a_x equals the index of $b_x \in B'$.⁸

If the transformation preserves similarity well, the *True Matches* should be very close to one another and, thus, close neighbors. In the best case, a sample's neighbor rank is 1 (*i.e.*, the sample is closest to its *True Match*). In the worst case, the sample's neighbor rank is the number of records in the opposing dataset (*i.e.*, the sample's *True Match* is the most distant record in the dataset to which the sample is being linked). We select ten independent samples and report the average and standard deviation.

4.4 Implementation Details

All experiments were run on a Linux server with four dual-core processors at 2 GHz and 16 GB of memory. All methods were implemented in Java. The datasets were stored as flat text files.

5 Results

Section 5.1 presents the results of the neighborhood test, Section 5.3 examines the relationship between the security constraint q and accuracy, and Section 5.4 provides a summary of the results.

5.1 Impact of Corruption

We use the neighborhood test to measure the accuracy afforded by different encoding methods. Figure 5 presents the results of the neighborhood test. In this experiment, we let $q = 1$, such that all FBF bits were eligible for inclusion in the RBF. To assess the statistical significance in the results, we performed a pairwise comparison of each of the methods at a neighbor rank of 1 using Welch's t-test. It was found that every method pair was significantly different at a p-value of 0.001 except for *static/uniform* vs. *dynamic/weighted* and *static/weighted* vs. *dynamic/weighted*, which were significant at a p-value of 0.01.

⁷As measured by the Dice coefficient

⁸In contrast to a measure like precision, which corresponds to the cases where the neighbor rank is 1, the neighborhood test allows us to get a feel for the extent to which similarity is preserved, even when it is not preserved to the greatest extent possible.

Notably, all experimental RBFs outperform the controls. These findings indicate that RBFs are more robust in accurately measuring similarity in the face of the types of errors present in this dataset.

In all cases, *weighted* bit sampling from FBFs, in accordance with the discriminatory power of the field, performs better than *uniform* bit sampling. Therefore, we recommend *weighted* bit sampling in RBF creation and explore this variant further in the following experiments.

Recall that in determining m_{RBF} , we made the assumption that all eligible bits satisfying the security constraint should be included in the RBF. We now test whether this assumption is necessary. Again in this experiment, we let $q = 1$. Figure 6 presents the results of the neighborhood test on RBFs of varying lengths. These results indicate that as m_{RBF} decreases, the extent to which RBF encodings are able to accurately measure similarity decreases as well. For example, when $m_{\text{RBF}} = 1315$, the number of samples having a neighbor rank of 1 is 981. When $m_{\text{RBF}} = 50$, only 792 samples have a neighbor rank of 1. However, it does not seem to be vital that the RBF should be sufficiently long to accommodate all bits from all FBFs as RBFs of length less than 1,315 bits still provide nearly as accurate results. For example when $m_{\text{RBF}} = 1,000$ and 500, the number of samples having a neighbor rank of 1 are 974 and 978, respectively.

5.2 Neighborhood Test Results

In recognition that the data corruptor may influence the results, we ran several additional experiments to assess how the degree of corruption modifies our findings. To do so, we created two additional environments; one with a low level of corruption and one with a high level of corruption. To simulate a low and high level of corruption, we generated datasets with corruption probabilities set to 50% and 150% of their standard level (see Section 6), respectively. The results of the comparison between *dynamic/weighted* RBFs and CLKs is depicted in Figure 7. As the amount of corruption in a dataset decreases, so too does the difference between the RBF and CLK. However, it was observed that RBF outperformed CLK at all corruption levels. The differences in the neighborhood rank 1 results were statistically significant using Welch's t-test at a p-value of 0.001 for the standard and high corruption levels and at a p-value of 0.002 for the low corruption level.

5.3 The Relationship Between the Security Constraint q and Accuracy

Recall that security constraint q indicates the number of fields represented in a given frequency range and is used to limit bit selection for RBF generation. We hypothesize that this increase in security may come at the cost of a decrease in accuracy. To test this hypothesis, we first consider the number of bits eligible for RBF inclusion under various values of the security constraint q . The results of this analysis are shown in Figure 8, which demonstrates the relationship between the security constraint q and the number of bits that are eligible for RBF inclusion. For example, consider the FBFs of *dynamic* size. If $q = 6$ (*i.e.*, only bits can be included in the RBF that can be mapped back to 6 fields), there are no bits would meet that security requirement. However, if a slightly more relaxed security constraint is used such that $q = 5$, then 312 bits are eligible for inclusion in the RBF.

The relationship between the security constraint q and the number of bits eligible for inclusion in the RBF implies there may also be a relationship between q and accuracy. To examine this relationship, we perform the neighborhood test on RBFs generated under various security constraints. These results, shown in Figure 9, indicate that accuracy decreases very slightly (981 samples have a neighbor rank of 1 when $q = 1$, and 970 samples have a neighbor rank of 1 when $q = 5$) when more stringent security constraints are introduced. However, the relationship does not appear to significantly affect accuracy nor is it linear. To further explore this relationship, the effect of security constraints is examined in the context of condensed RBFs. These results are shown in Figure 10. Again, the results indicate that as q increases, accuracy tends to decrease. This is shown by the decreased number of samples having a neighbor rank of 1.

These results are further broken down in Figure 11. In this figure, the scenario where the neighbor rank is 1 is examined in greater detail to model the tradeoffs between q and accuracy. The results indicate q does not significantly affect accuracy until very small RBF sizes ($m_{\text{RBF}} = 100$) are applied.

5.4 Summary of Results

Section 5.1 showed that RBF encodings provide for more accurate record pair similarity computation than several control measures, indicating that RBFs are more robust in accurately measuring similarity in the face of the types of errors present in this dataset. Additionally, *weighted* bit sampling in RBF generation provides higher accuracy than *uniform* bit sampling across FBFs. It was also demonstrated that as the RBF is condensed (*i.e.*, the m_{RBF} value is lowered), the accuracy decreases slightly.

Section 5.3 demonstrates that enforcing a stringent security constraint q eliminates some FBF bits from inclusion in the RBF, but does not necessarily lead to a high cost in the accuracy of comparisons of the resultant encodings. RBFs created under a stringent value of the security constraint q are only slightly less accurate than those created without any constraints.

6 Discussion

This work described an encoding strategy for PRL that enhances the security properties of BF encodings as proposed in the literature [8] while preserving their efficiency and accuracy in record linkage. In the remainder of this section, we demonstrate that the RBF encoding method satisfies these properties.

6.1 Record-level Bloom Filters are Resistant to a Known Attack

In Section 2.3, we described a known attack on FBF encodings. The attack is modeled as a constraint satisfaction problem, where the task is to map the bits in the encoding to the corresponding plaintext n -grams. This is accomplished by leveraging the frequency with which bits in the encoding are set and the frequency with which n -gram values are seen in the general population.

An attacker can use publicly available demographic information, such as census records [21], to estimate these frequencies. However, when field values are grouped together in a disordered way (e.g., in a shuffled RBF), it becomes more difficult to exploit such a resource. For example, the frequency distribution of *Forename* may be available and relatively distinct. However, the frequency distribution of {Surname, Forename, City, Street address, Ethnicity, Gender} is more difficult to calculate and less distinct. Additionally, an attacker must determine which bits in the RBF correspond to each field. As discussed in Section 3.2, security constraints can be invoked to make this arbitrarily difficult.

In addition to being unable to gather the background information needed for this attack, the computational complexity of the attack grows quadratically when field values are grouped together into a single RBF. When FBF encodings are used, an attacker can handle each field individually. Therefore, in order to map the FBFs corresponding to the six fields back to their plaintext value, in the worst case, the attacker would have to perform $|Surname| + |Forename| + |City| + |Street\ address| + |Ethnicity| + |Gender|$ operations $|\bullet|$ where $|\bullet|$ is the size of the domain of field $\bullet$. However, when these fields are encoded in a RBF, an attacker must perform $|Surname| \times |Forename| \times |City| \times |Street\ address| \times |Ethnicity| \times |Gender|$ operations. This number of computations is outside the scope of standard computing frameworks. To illustrate the magnitude of difference consider that the NCVR dataset consists of approximately 274000 *Surnames*, 187000 *Forenames*, 94000 *Street names*, 790 *Cities*, and 2 *Genders*. Thus, an attack on FBFs and an RBF would require on the order of 5.5×10^5 and 7.6×10^{18} operations, respectively.

Since the attacker cannot gather the background knowledge necessary to mount this kind of attack, and the computational complexity renders it intractable on standard computing frameworks, the RBF encodings presented in this paper are more resistant to this kind of attack.

Unfortunately, the proposed RBF encoding scheme does not have any formal security proofs. Clearly, it would be ideal to prove that our RBF encoding scheme is a secure hash function that only leaks the nearest neighbor relationship (i.e., if record r_i is the nearest neighbor of record of r_j ($1 - nn(r_i) = r_j$), then such nearest neighbor relationship is preserved after RBF encoding ($1 - nn(RBF(r_i)) = RBF(r_j)$)). Instead, we empirically show that proposed construction is resilient to the known attacks and scale well for large datasets. Previous work including our own provides protocols (e.g., [12], [33]) that offer theoretical security guarantees based on cryptographic techniques (e.g., secure multi-party computation techniques). Yet, such protocols are far from being practical even for a medium-size datasets [6]. On the other hand, the proposed scheme offers a practical solution that can scale well for real-world datasets without any obvious security weakness.

6.2 Record-level Bloom Filters are Efficient

Since our goal is to preserve the efficiency of FBF encoding generation, it is critical that the proposed RBF encoding scheme has very little overhead. In our Java-based implementation, on average, it takes 1.45ms to construct an RBF from a given FBF. This implies that for a dataset of size 100,000, the proposed RBF scheme will have an overhead less than 3 minutes.

Of course, the other important part of the overall computational performance is to compare generated RBFs. We note that both RBFs and FBFs are similar sized bit vectors; therefore, comparing RBFs does not bring any significant overhead compared to comparing FBFs. In our experiments, on average, it takes 0.0067 ms to compare RBFs. This means that even the naive quadratic record linkage algorithm running on two datasets of size 100,000 can finish in less than 19 hours. On the other hand, some of the proposed SMC based protocols would take years to link two datasets of size 100,000 (assuming no blocking) [6].

6.3 Limitations and Future Work

Despite the merits of our work, there are several avenues left to future research. First, indexing, also known as blocking, seeks to reduce the computational complexity of record linkage by limiting the number of record pairs that are evaluated and classified [34]–[37]. There have been various investigations in privacy preserving blocking algorithms [33], [38], [39] and preliminary work has shown that RBF encodings are suitable data structures to serve as inputs for blocking methodology in the record linkage process [22]. Blocking will not influence the structure of RBFs and thus will not affect the security of such a data structure, but it may limit the matching accuracy of an RBF. We plan to explore this aspect of the RBF data structure in future work.

Second, we believe that the similarity of RBFs are an accurate measure of record pair similarity and can be used directly in PRL to replace an explicit record pair comparison step in which field similarities are consolidated into a single measure of record pair similarity.

Finally, we plan to investigate how to handle missing field values in the context of Bloom filter encodings.

7 Related Research

As alluded to earlier, there are several classes of approaches that have been proposed for PRL. One class utilizes secure multiparty computation (SMC) techniques, which allow the similarity comparison between two records to be performed without revealing *any* information about the input records. One example of this is an SMC approach for calculating the edit distance between strings [12]. However, such techniques are computationally intense and do not scale well in practice [6].

Another class of approaches for performing PRL proposes the encryption of each field value [9]–[11], [40], [41] and sending the encodings to a centralized third party, who then performs record linkage by comparing the encoded field values [9], [42]. A drawback to the use of traditional encryption methods is the loss of the ability to detect field similarity in the encrypted space. This stems from the fact that encryption methods are not similarity-preserving transformations. Therefore, records that exhibit high similarity in the plaintext space (e.g., “John” and “Jon”) may be quite dissimilar in the ciphertext space. As a result, encodings that are *True Matches* may be classified as *Predicted Non-matches*.

To address this drawback, several encoding methods (e.g., [8], [43]–[53]) have been developed to enable similarity detection in the ciphertext space. The ability to measure the

approximate similarity of encoded values means that the transformation from plaintext to ciphertext is not as lossy. Therefore, encodings which provide for approximate field comparison have the potential to improve record linkage accuracy. In fact, the evaluation published in [6] shows this to be the case.

The work most similar to our own is that of the CLK reported in a recent working paper [14]. While CLKs are similar to RBFs in that they both encode records into a single BF, there are several significant differences. First, the core CLK method as described in [14] uses the same set of hash functions for all fields. A benefit of this strategy is that it can assist in making CLKs robust to CSP attacks. However, a drawback of this approach is that it can lead to semantic conflicts, such that the same n -gram in two different fields will be treated as equal (e.g., “jo” may have been derived from the forename or surname). Second, the methodology behind CLKs is not yet well-developed. Specifically, formal methodology is not provided for weighting different field values according to their discriminatory power other than “use relatively more hash functions” to store more discriminatory fields. While [14] notes that changing encoding parameters, and in particular the ratio of the number of hash functions used to the length of the Bloom filter, has security implications, there are no concrete recommendations on how to set parameters and no formal proof of security. We believe we have provided a fair comparison with CLKs in the empirical analysis, which suggest that the RBF strategy yields more accurate results with more explicit security guarantees.

8 Conclusions and Future Work

In this paper, we presented a method to construct RBF encodings for use in PRL. Our method is flexible in that it provides for several design choices, which users can adapt to best suit their requirements with respect to security, the speed at which the encodings can be compared to one another, and the desired accuracy of similarity comparisons based on the encodings. Our empirical investigations have shown that using FBFs of *dynamic* size and sampling random bits from FBFs in a measure commensurate with the discriminatory power of the field are important for secure RBF encodings that facilitate accurate similarity calculations. Most importantly, we have demonstrated that the new encodings are resistant to a cryptanalysis attack that has been successful against previously proposed forms of FBF encodings. We note our approach provides part of the foundation for a private record linkage system, but highlight that additional work is necessary to ensure its application in a complete record linkage framework, which will require coordination for field weighting, blocking, and linkage.

Acknowledgements

This research was supported by grants R01-LM009989 and 2-T15LM0745006 from the NIH and CNS-0845803, CNS-0964350, and CCF-0424422 from the NSF.

Biography

Elizabeth Ashley Durham received the B.S. degree in computer science from Georgia Institute of Technology in Atlanta, GA, and the Ph.D. degree in biomedical informatics from Vanderbilt University in Nashville, TN.

She is currently a post-doctoral research fellow who splits her time between Vanderbilt University and the Centre for Health Informatics & Multiprofessional Education (CHIME) at University of College London in England. Her research interests include computational biology, data privacy, and global health informatics. She was a recipient of a graduate training fellowship from the U.S. National Library of Medicine, National Institutes of Health and, in 2010, she received the best student paper award from the American Medical Informatics Association.

Murat Kantarcioglu received the B.S. degree in computer engineering from Middle East Technical University in Ankara, Turkey, and the M.S. and Ph.D. degrees in computer science from Purdue University in West Lafayette, IN.

He is currently an Associate Professor of Computer Science and the Director of the UTD Data Security and Privacy Laboratory at the University of Texas at Dallas in Richardson, TX. His research focuses on creating technologies that can efficiently extract useful information from any data without sacrificing privacy or security. His research has been supported by grants from the National Science Foundation (NDF), the Air Force Office of Scientific Research, the Office of the Naval Research, the National Security Agency, and the National Institutes of Health.

Dr. Kantarcioglu's research has been covered by various media outlets, including the Boston Globe and NBC News. His publications have received two best paper awards. He is the recipient of an NSF CAREER Award and the Purdue CERIAS Diamond Award for academic excellence.

Yuan Xue received the B.S. degree in computer science from Harbin Institute of Technology in Harbin, China and the Ph.D. degree in computer science from University of Illinois at Urbana-Champaign in Urbana, IL.

She is currently an Assistant Professor of Computer Engineering and Computer Science at Vanderbilt University in Nashville, TN. Her research areas includes networking and distributed systems with a focus on wireless and mobile systems, web applications and services, clinical information systems, and cloud computing. Her research goal is to provide performance optimization, quality of service support, application and data integrity and privacy for next generation networking and distributed systems.

Dr. Xue is the recipient of an NSF CAREER Award, has been a visiting research at the Air Force Research Lab, and has received several best paper awards.

Mehmet Kuzu Mehmet Kuzu received the B.S. and M.S. degrees in computer engineering from Middle East Technical University in Ankara, Turkey, in 2007 and 2009, respectively.

He is currently a Ph.D. student in the computer science department at the University of Texas at Dallas. His research areas include intelligent systems, searchable encryption, and security and privacy issues related to the management of data.

Csaba Toth received the M.S. degree in computer science from Budapest University of Technology and Economics in Budapest, Hungary. He is currently a software engineer in the Health Information Privacy Laboratory in the Department of Biomedical Informatics at

Vanderbilt University in Nashville, TN. He has experience developing software in various programming platforms using C, C++, Java, .NET and other significant platforms in industrial and research projects.

Bradley Malin received the B.S. degree in biological sciences, M.S. degree in machine learning, M.Phil. degree in public policy and management, and the Ph.D. degree in computer science, all from Carnegie Mellon University, in Pittsburgh, PA.

He is currently an Associate Professor of Biomedical Informatics, in the School of Medicine, and Computer Science, in the School of Engineering, at Vanderbilt University in Nashville, TN, where he directs the Health Information Privacy Laboratory. His current research interests include data mining, biomedical informatics, and trustworthy computing. His research has been supported by the National Science Foundation and the National Institutes of Health, with which he directs a data privacy and research consultation team for the Electronic Medical Records and Genomics (eMERGE) consortium.

Dr. Malin's research has been cited in various governmental proceedings on health information privacy and security. He is a recipient of the Presidential Early Career Award for Scientists and Engineers (PECASE).

References

1. Elfeky M, Verykios V, Elmagarmid A. Tailor: a record linkage toolbox. Proc. 18th IEEE Int'l Conf. Data Eng. 2002:17–28.
2. Bhattacharya I, Getoor L. Iterative record linkage for cleaning and integration. Proc. 9th ACM SIGMOD Workshop on Research Issues in Data Mining & Knowledge Discovery. 2004:11–18.
3. Halevy A, Rajaraman A, Ordille J. Data integration: the teenage years. Proc. 32nd Int'l Conf. on Very Large Data Bases. 2006:9–16.
4. Vatsalan D, Christen P, Verykios V. A taxonomy of privacy-preserving record linkage techniques. Information Systems. 2013 in press.
5. Yao AC. Protocols for secure computations. IEEE Annual Symp. on Foundations of Computer Science. 1982:160–164.
6. Durham E, Xue Y, Kantarcioglu M, Malin B. Quantifying the correctness, computational complexity, and security of privacy-preserving string comparators for record linkage. Information Fusion. 2012; 13(4):245–259. [PubMed: 22904698]
7. Christen, P. Data matching: concepts and techniques for record linkage, entity resolution, and data deduplication. Springer; 2012.
8. Schnell R, Bachteler T, Reiher J. Privacy-preserving record linkage using bloom filters. BMC Medical Informatics & Decision Making. 2009; 9(1):41. [PubMed: 19706187]
9. Quantin C, Bouzelat H, Allaert F, et al. How to ensure data security of an epidemiological follow-up: quality assessment of an anonymous record linkage procedure. Int'l Journal of Medical Informatics. 1998; 49(1):117–122.

10. Berman JJ. Zero-check: A zero-knowledge protocol for reconciling patient identities across institutions. *Archives of Pathology & Laboratory Medicine*. 2004; 128(3):344–346. [PubMed: 14987147]
11. Agrawal R, Evfimievski A, Srikant R. Information sharing across private databases. *Proc. 2003 ACM SIGMOD Int'l Conf. on Management of Data*. 2003:86–97.
12. Atallah, M.; Kerschbaum, F.; Du, W. *Proc. ACM Workshop on Privacy in the Electronic Society*. Washington, DC: 2003. Secure and private sequence comparisons; p. 39-44.
13. Kuzu M, Kantarcioglu M, Durham E, Malin B. A constraint satisfaction cryptanalysis of bloom filters in private record linkage. *Proc. 11th Privacy Enhancing Technologies Symp*. 2011:226–245.
14. Schnell R, Bachteler T, Reiher J. A novel error-tolerant anonymous linking code. *German Record Linkage Center*. 2011 Working Paper Series No. WP-GRLC-2011-02.
15. Lucks M. A constraint satisfaction algorithm for the automated decryption of simple substitution ciphers. *Proc. Annual International Cryptology Conf*. 1990:132–44.
16. Kuzu M, Kantarcioglu M, Durham E, Toth C, Malin B. A practical approach to achieve private medical record linkage in light of public resources. *Journal of the American Medical Informatics Association*. doi:10.1136/amiajnl-2012-000917, in press.
17. North Carolina Voter Registration Database. <ftp://www.app.sboe.state.nc.us/data>
18. Freely Extensible Biomedical Record Linkage (FEBRL) software. <http://sourceforge.net/projects/febrl/>
19. Christen P, Pudjijono A. Accurate synthetic generation of realistic personal information. *Proc. 13th Pacific-Asia Conf. on Advances in Knowledge Discovery & Data Mining*. 2009:507–514.
20. Massmind Nicknames Database. <http://techref.massmind.org/techref/ecommerce/nicknames.htm>
21. Census Bureau, US. Population Division. http://www.census.gov/genealogy/names/names_files.html
22. Durham, EA. Ph.D. dissertation. Vanderbilt University; 2012. A framework for accurate, efficient private record linkage.
23. Grannis, SJ.; Overhage, JM.; Hui, S.; McDonald, CJ. Analysis of a probabilistic record linkage technique without human review. *Proc. American Medical Informatics Association Annual Symp*; Washington, DC. 2003; p. 259-263.
24. Broder A, Mitzenmacher M. Network applications of Bloom filters: A survey. *Internet Mathematics*. 2002; 1:636–646.
25. Fellegi I, Sunter A. A theory for record linkage. *Journal of the American Statistical Society*. 1969:1183–1210.
26. Winkler, WE. Using the EM algorithm for weight computation in the Fellegi-Sunter model of record linkage. U.S. Census Bureau; 2000. Research Report RR00/05
27. Jaro, MA. UNIMATCH: a record linkage system. U.S. Census Bureau; Washington, DC: 1978.
28. Torra, V.; Domingo-Ferrer, J. *Information Fusion in Data Mining*. Springer; 2003. Record linkage methods for multidatabase data mining; p. 101-132.
29. National Institute of Standards and Technology. Advanced encryption standard AES. Vol. 197. Federal Information Processing Standards Publication; p. 2001
30. Porter EH, Winkler WE. Approximate String Comparison and its Effect on an Advanced Record Linkage System. 1997
31. Cohen WW, Ravikumar P, Fienberg SE. A comparison of string distance metrics for name-matching tasks. *Proc. IJCAI- 03 Workshop on Information Integration*. 2003:73–78.
32. Grannis SJ, Overhage JM, McDonald C. Real world performance of approximate string comparators for use in patient matching. 2004; 107(Pt 1):43–47.
33. Inan A, Kantarcioglu M, Ghinita G, Bertino E. Private record matching using differential privacy. *Proc. 13th Int'l Conf. on Extending Database Technology*. 2010:123–134.
34. Whang SE, Menestrina D, Koutrika G, Theobald M, Garcia-Molina H. Entity resolution with iterative blocking. *Proc. 35th SIGMOD Int'l Conf. on Management of Data*. 2009:219–232.
35. Koudas N, Sarawagi S, Srivastava D. Record linkage: similarity measures and algorithms. *Proc. 32nd ACM SIGMOD Int'l Conf. on Management of Data*. 2006:802–803.

36. Dey D, Mookerjee V, Liu D. Efficient techniques for online record linkage. *IEEE Trans. on Knowl. and Data Eng.* 2011; 23(3):373–387.
37. Christen P. A survey of indexing techniques for scalable record linkage and deduplication. *IEEE Trans. on Knowl. and Data Eng.* 2012; 99(9):1537–1555.
38. Al-Lawati, A.; Lee, D.; McDaniel, P. Blocking-aware private record linkage. *Proc. Int'l Workshop on Information Quality in Information Systems*; Baltimore, MD. 2005; p. 59-68.
39. Karakasidis A, Verykios V. Reference table based k-anonymous private blocking. *Proc. 27th Annual ACM Symp. on Applied Computing.* 2012:859–864.
40. van Eycken E, Haustermans K, Buntinx F, et al. Evaluation of the encryption procedure and record linkage in the Belgian national cancer registry. *Archives of Public Health.* 2000; 58:281–294.
41. Agrawal R, Asonov D, Srikant R. Enabling sovereign information sharing using web services. *Proc. 2004 ACM SIGMOD Int'l Conf. on Management of Data.* 2004:873–877.
42. Clifton C, Kantarcioglu M, Doan A, Schadow G, Vaidya J, Elmagarmid A, Suci D. Privacy-preserving data integration and sharing. *Proc. 9th ACM SIGMOD Workshop on Research Issues in Data Mining & Knowledge Discovery.* 2004:19–26.
43. Churches T, Christen P. Some methods for blindfolded record linkage. *BMC Medical Informatics & Decision Making.* 2004; 4(1):9. [PubMed: 15222890]
44. Song, D.; Wagner, D.; Perrig, A. Practical techniques for searches on encrypted data. *Proc. IEEE Symposium on Security & Privacy*; Oakland, CA. 2000; p. 44
45. Kaya SV, Pedersen TB, Savas E, Saygin Y. Efficient Privacy Preserving Distributed Clustering Based on Secret Sharing. 2007:280–291.
46. Inan A, Kaya S, Saygin Y, Savas E, Hintoglu A, Levi A. Privacy preserving clustering on horizontally partitioned data. *Data and Knowledge Eng.* 2007; 63:646–666.
47. Ravikumar P, Fienberg SE. A secure protocol for computing string distance metrics. *Proc. IEEE Workshop on Privacy and Security Aspects of Data Mining.* 2004:40–46.
48. Hylton, J. Master's thesis, M.I.T. 1996. Identifying and merging related bibliographic records.
49. Pang C, Gu L, Hansen D, Maeder A. Privacy-preserving fuzzy matching using a public reference table. *Intelligent Patient Management.* 2009; 189:71–89.
50. Scannapieco M, Figotin I, Bertino E, Elmagarmid A. Privacy preserving schema and data matching. *Proc. ACM SIGMOD Int'l Conf. on Management of Data.* 2007:653–664.
51. Karakasidis, A.; Verykios, VS. Privacy preserving record linkage using phonetic codes. *Proc. 4th Balkan Conf. in Informatics*; Thessaloniki, Greece. 2009; p. 101-106.
52. Jonas J, Pattak PB, Litchko JP. Using advanced information technology to combat insider threats. *J. Organizational Excellence.* 2001; 20(4):19–27.
53. Du W, Atallah MJ. Protocols for secure remote database access with approximate matching. CERIAS, Purdue University, Tech. Rep. 2001

Fig. 1.

RBF encoding generation. 1) **FBF parameterization & generation.** FBFs are sized based on the expected field length. In this example, there are two hash functions per FBF. 2) **Eligible bit identification.** More secure bits, shown with a heavy outline, are identified for inclusion in the RBF. 3) **Field weighting.** It is determined that 25% of the bits in the RBF should be drawn from *Forename*, 50% of the bits in the RBF should be drawn from *Surname*, and 25% of the bits in the RBF should be drawn from *City*. 4) **RBF parameterization & generation.** m_{RBF} is determined and eligible bits are sampled according to the field weights. 5) **RBF permutation.** Bits in the RBF are randomly permuted. The number of bits set in the RBF shown to the left and right is 11 and 15, respectively. The number of bits in the intersection is 11, yielding a Dice coefficient of 0.84.

Fig. 2.

The effect of FBF sizing on the average (and standard deviation) percent of bits set to 1 for the static and dynamic settings. The percent of bits set is calculated over all of the encoded NCVR records, both clean and corrupt. The variability across fields for static FBFs reveals information about the length of encoded field values, but this is not the case for dynamic FBFs.

Fig. 3.

A heat map representation of the frequencies at which bits in the FBFs are set for select fields.

Fig. 4.

Histograms for the number of bits in a) *static* and b) *dynamic* FBFs set to 1 at frequency intervals of $\frac{1}{100}$.

Fig. 5.

The results of the neighborhood test where each color indicates the neighbor rank of the sample's *True Match* (black: top rank, dark gray: 2nd through 10th rank, light gray: beyond the 10th rank). The controls include binary field comparison and the JW string comparator. The experimental techniques are the RBFs based on statically or dynamically sized FBFs. The numbers on the bars correspond to how many of the 100 neighborhood tests in which the first rank was the *True Match*.

Fig. 6.

The results of the neighborhood test for *dynamic/weighted* RBFs as a function of m_{RBF} (black: top rank, dark gray: 2nd through 10th rank, light gray: beyond the 10th rank).

Fig. 7.

The results of the neighborhood test for *dynamic/weighted* RBFs and CLK for varying levels of corruption (black: top rank, dark gray: 2nd through 10th rank, light gray: beyond the 10th rank).

Fig. 8.

The number of bits eligible for RBF inclusion, given various values for the security constraint q , for dynamic and static settings.

Fig. 9.

Neighborhood test results show that the effect of the security constraint q on *Dynamic/Weighted* RBF accuracy, as determined by the Neighborhood test (black: top rank, dark gray: 2nd through 10th rank, light gray: beyond the 10th rank).

Fig. 10.

These neighborhood test results show the effect of *Dynamic/Weighted* RBF sizing and the security constraint q (black: top rank, dark gray: 2nd through 10th rank, light gray: beyond the 10th rank).

Fig. 11.

A detailed view of the effect of the security constraint (q) on neighborhood order preservation.

TABLE 1

An example of FBF parameterization when $k = 15$, $n = 2$, $p = 0.5$. The m_{FBF} value varies across fields when FBFs are dynamically sized but remains the same across fields when FBFs are statically sized.

	Surname	Forename	City	Street	Ethnicity	Gender
Average field length	6.3	5.8	8.8	8.7	1.0	1.0
Average # n -grams (g)	7.4	6.8	9.8	9.7	2.0	2.0
<i>Dynamic</i> m_{FBF}	159	149	214	212	44	44
<i>Static</i> m_{FBF}	500	500	500	500	500	500

TABLE 2

Weighted RBF creation. FS weights can assist in determining the extent to which each FBF should contribute to the RBF.

	Surname	Forename	City	Street	Ethnicity	Gender
w_a	6.9	6.0	4.3	8.2	0.6	0.8
w_d	-0.8	-0.9	-0.8	-1.0	-0.5	-0.7
<i>range</i>	7.7	6.9	5.1	9.2	1.1	1.5
w	24%	22%	16%	29%	4%	5%

TABLE 3

An example of RBF parameterization for *Dynamic/Weighted* RBFs.

	<i>Surname</i>	<i>Forename</i>	<i>City</i>	<i>Street</i>	<i>Ethnicity</i>	<i>Gender</i>	Total
m_{FBF}	159	149	214	212	44	44	–
w	24%	22%	16%	29%	4%	5%	100%
Possible m_{RBF} values	651	683	1,315	726	1,245	934	–
RBF composition	321	287	214	384	46	62	1,315