A Constraint Satisfaction Cryptanalysis of Bloom Filters in Private Record Linkage

Mehmet Kuzu[1], Murat Kantarcioglu[1], Elizabeth Durham[2], and Bradley Malin[2]

[1] Dept. of Computer Science, University of Texas at Dallas
Richardson, TX, 75080 USA
{mxk093120, muratk}@utdallas.edu
[2] Dept. of Biomedical Informatics, Vanderbilt University
Nashville, TN, 37232 USA
{ea.durham, b.malin}@vanderbilt.edu

Abstract. For over fifty years, "record linkage" procedures have been refined to integrate data in the face of typographical and semantic errors. These procedures are traditionally performed over personal identifiers (e.g., names), but in modern decentralized environments, privacy concerns have led to regulations that require the obfuscation of such attributes. Various techniques have been proposed to resolve the tension, including secure multi-party computation protocols, however, such protocols are computationally intensive and do not scale for real world linkage scenarios. More recently, procedures based on Bloom filter encoding (BFE) have gained traction in various applications, such as healthcare, where they yield highly accurate record linkage results in a reasonable amount of time. Though promising, no formal security analysis has been designed or applied to this emerging model, which is of concern considering the sensitivity of the corresponding data. In this paper, we introduce a novel attack, based on constraint satisfaction, to provide a rigorous analysis for BFE and guidelines regarding how to mitigate risk against the attack. In addition, we conduct an empirical analysis with data derived from public voter records to illustrate the feasibility of the attack. Our investigations show that the parameters of the BFE protocol can be configured to make it relatively resilient to the proposed attack without significant reduction in record linkage performance.

1 Introduction

There are many societal needs, as well as legal requirements, for organizations to share data about their constituents in support of a wide range of endeavors, ranging from homeland security to biomedical research. At the same time, increasing decentralization of our world has led to the storage of an individual's personal information across independent organizations. To ensure accurate analytics, it is critical to apply "record linkage" techniques to integrate information that corresponds to the same individual. Record linkage is a relatively mature field and a sizable number of algorithms have been refined to support the task [1]. Yet, record linkage has traditionally been applied to explicit identifiers, such as names and Social Security Numbers, and there are concerns that sharing such information beyond an organization's boundaries can endanger an individual's privacy. To mitigate risk, various private record linkage (PRL) protocols have been developed to enable data integration without revealing the identity of the corresponding individuals (e.g., [2–6]).

Most data sources contain records with typographical (e.g., "ei" vs. "ie") or semantic errors (e.g., maiden vs. married name) [7, 8], so it is critical that PRL protocols enable similarity tests between records. It has been demonstrated that such tests can be accomplished through the use of sophisticated cryptographic methods based on secure multi-party computation (SMC) [9, 10]. Unfortunately, protocols based on SMC are not practical for large dataset integration because they incur substantial computational costs and require continuous interaction between the organizations involved in the protocol [11]. More recently, a PRL protocol based on Bloom filter encoding (BFE) was proposed to measure the similarity of records in a more efficient manner, which is particularly notable because it has gained traction in the medical environment [12]. Although the accuracy and performance of the BFE approach are promising for real world applications [4], a detailed cryptanalysis has not been performed. Given the sensitivity of the data which BFEs are being proposed to protect (e.g., medical information), we believe a formal analysis is necessary and timely. In this paper, we construct a novel attack for BFEs, based on a combination of constraint satisfaction and intelligent heuristics. We use real personal identifiers, derived from publicly available resources, to empirically illustrate that the attack can compromise a significant amount of private data if the parameters of the BFE are selected as suggested in the literature. In summary, there are several notable contributions of this paper:

Frequency-Aware Constraint Satisfaction Model: We frame the attack against the BFE protocol as a constraint satisfaction problem (CSP). Though a cryptanalytic method based on constraint satisfaction was previously proposed for simple substitution ciphers [13], it does not directly apply to BFEs, which are significantly more complex due to the ingredients involved in the encoding process (see Section 2). The proposed cryptanalytic approach integrates frequency analysis into the construction of the CSP to reduce the complexity.

Statistically Reliable Constraints: The CSP attack on the BFE protocol leverages constraints that are approximately correct (i.e., accurate with a very high probability). While the constraints in a CSP should be accurate, an overspecified system can lead to high computational costs when solving the problem. By utilizing statistically reliable constraints, we enable a complex CSP to be solved efficiently by pruning the search space of the CSP solver.

Empirical Vulnerability Assessment: We show that the BFE protocol is vulnerable to attack, provided that the adversary has a certain amount of reasonable background information. At the same time, we explore the relationship between the encoding parameters (e.g., filter length) and vulnerability through extensive experiments. Our investigations show that BFE can be made relatively resilient to the proposed attack by tuning the BFE parameters appropriately.

The remainder of the paper is organized as follows. Section 2 provides background information and describes the adversarial model. Section 3 presents the proposed attack. We then report our experimental analysis in Section 4. We review related work in Section 5 and conclude in Section 6.

2 Background

We begin with an overview of the BFE techniques and our threat model. A legend for the notation used throughout the paper is provided in Table 1.

Symbol	Description	Symbol	Description
A, B	datasets of Alice and Bob	A^T, B^T	encoded versions of A, B
G	global dataset	D, D^T	$(A \cup B)$, $(A^T \cup B^T)$
g	string encoding function	f	n-gram encoding function
bf	belief function	X_i	variable of CSP
v_i	value assigned to X_i	q_i, Q_{X_i}	single n-gram, n-gram set of X_i
m	filter length	k	number of hash functions in f

Table 1: Symbol Definitions

2.1 Bloom Filter Encoding

A Bloom filter [14] is a bit array of length m that is affiliated with k hash functions. Each function maps a given element to one bit location with a uniform probability. For the purposes of this work, we define an element as a string, $S \in \Sigma^*$, over an alphabet Σ.

In the BFE model described in [12], S is represented as the set of substrings of length n, or n-grams such that $Q_S = \{q_1, \ldots, q_z\}$. Each n-gram is subject to each hash function and the corresponding bit indices are set to 1. The encoding of a string is then obtained by combining the n-gram encodings with the bitwise OR ($\vee$) operation. Formally, let $f : \Sigma^n \mapsto \{0, 1\}^m$ be the n-gram encoding function obtained by combining k hash functions and $g : \Sigma^* \mapsto \{0, 1\}^m$ be the string encoding function that converts any string into its BFE:

$$g(S) = \bigvee_{i=1}^{z} f(q_i) \tag{1}$$

Example 2.1 : In Figure 1, the bigrams of "amy" are encoded with two hash functions. Notice "␣a" and "y␣" are mapped to the same index by one of the hash functions.

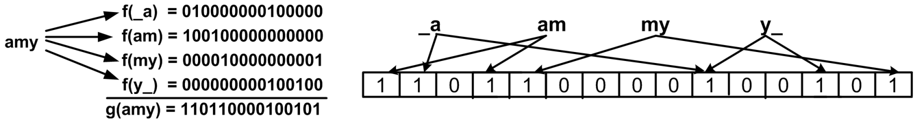


Fig. 1: Sample BFE: $Q_{amy} = \{\_a, am, my, y\_\}$

2.2 Threat Model

In this paper, the linkage of data sources A and B, owned by Alice and Bob, respectively, is facilitated through a third party, Charlie. The utilization of the third party is a common practice in real world privacy preserving data sharing environments for healthcare (e.g., [15]). First, the data owners agree on the filter length, the keyed hash functions, and the secret keys. Next, the owners convert their strings into the BFEs. Then, the owners transfer the BFE lists $(A^T \cup B^T)$ (Henceforth, we will use D and D^T to refer $(A \cup B)$ and $(A^T \cup B^T)$, respectively.) to Charlie who compares the BFEs using a set-based similarity measure to find matching pairs of records.

In this setting, we assume Charlie is the adversary and does not collude with Alice or Bob. Charlie attempts to expose the original records based on their BFEs without access to the secret keys. However, we assume that Charlie has access to the following practical background knowledge:

Assumption 1: Charlie knows the global dataset G from which A and B are drawn, such that $D \subseteq G$. This is reasonable because record linkage is typically performed with personal identifiers (e.g., names and addresses), which in many countries, can be found in public resources.

Assumption 2: Charlie knows the number of hash functions (k) that are part of the Bloom encoding function. This is reasonable because Charlie can infer the number of hash functions from the number of bits set in the Bloom filters. Due to hash collisions, Charlie can only estimate this number, but it can be restricted to a small range, such that the proposed attack can be applied for all values in the range.

3 Overview of the Attack

Here, we provide an overview of the attack Charlie can execute on the BFE system. Without loss of generality, let D be represented in relational form $D(Attr_1, \ldots, Attr_y)$ with string-valued attributes that correspond to personal identifiers (e.g., surnames). For each record $t_i \in D$, the Bloom encoding process results in an encoding $t_i^T = (g(t_i.Attr_1), \ldots, g(t_i.Attr_y))$. An attribute encoding $t_i^T.Attr_j$ is said to be compromised, if Charlie learns $g^{-1}(t_i^T.Attr_j) = t_i.Attr_j$. The components of the attack are presented in subsections 3.2 to 3.4 and we refer the reader to Appendix C for a visualization of the attack flow.

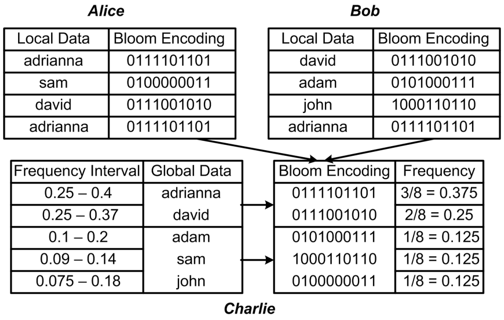
3.1 Motivating Example

Consider the scenario in Figure 2. Alice and Bob generate BFEs, which are transferred to Charlie, who initiates the attack by calculating the frequency distribution of the items in D^T. Next, Charlie performs a statistical analysis on G to form frequency intervals of the items in an arbitrary dataset of size equal to D^T. The possible encodings that could be associated with a particular string is reduced significantly via frequency analysis as illustrated in the following example.

Example 3.1.1 (Frequency Utilization): In Figure 2, Charlie estimates the frequency interval of “adrianna” as [0.25, 0.4] by performing a statistical analysis on G. Charlie then observes that the frequencies {0.375, 0.25} of BFEs {0111101101, 0111001010} are the only ones in the interval of “adrianna”. As a result, possible encodings for “adrianna” is reduced to these BFEs. □

In addition to frequency analysis, Charlie can use BFEs and the properties of the encoding for the attack as illustrated in the following example.

Example 3.1.2 (Encoding Utilization): In Figure 2, “david” contains 6 bigrams {_d, da, av, vi, id, d_} with a possible encoding set {0111101101, 0111001010}. Notice, the encoding of “david” cannot contain more than six 1’s according to the construction of the BFE. The only encoding that satisfies this condition is {0111001010}. Once “david” is mapped to {0111001010}, “adrianna” can only be mapped to {0111101101} since encoding function is one-to-one with very high probability (see Section 3.3). It should be noted that each mapping of a name to an encoding reveals additional knowledge about the behavior of the encoding function. For instance, once “sam” is mapped to {0100000011}, it is known that the encoding function can only set the second, ninth, and tenth bit locations when applied on the set of bigrams {_s, sa, am, m_}. And, this type of knowledge revision can be used to reveal further assignments. Notice that after the mappings of “david”, “adrianna” and “sam”, the possible encodings for “adam” contains {0101000111, 1000110110}. But “adam” consists of the bigrams {_a, ad, da, am, m_} which have been included in previous mappings. Thus, to be compatible with previous mappings, the first bit location of the encoding for “adam” cannot be set to 1. The only encoding that satisfies this constraint is {0101000111}, so “adam” is correctly assigned to it.□



Local Data	Bloom Encoding
adrianna	0111101101
sam	0100000011
david	0111001010
adrianna	0111101101

Local Data	Bloom Encoding
david	0111001010
adam	0101000111
john	1000110110
adrianna	0111101101

Frequency Interval	Global Data
0.25 – 0.4	adrianna
0.25 – 0.37	david
0.1 – 0.2	adam
0.09 – 0.14	sam
0.075 – 0.18	john

Bloom Encoding	Frequency
0111101101	3/8 = 0.375
0111001010	2/8 = 0.25
0101000111	1/8 = 0.125
1000110110	1/8 = 0.125
0100000011	1/8 = 0.125

Fig. 2: Records are embedded into a 10 bit Bloom filter with 1 hash function

3.2 Bloom Encoding Analysis

Some constraints can be derived from the properties of f, the n-gram encoding function. The derived constraints should be satisfied by the mapping between the original strings and the corresponding BFEs. Therefore, the problem of discovering the mappings can be modeled as a CSP. Generally, a CSP is defined by a set of variables $\{X_1, X_2, \ldots, X_p\}$, and a set of constraints $\{C_1, C_2, \ldots, C_r\}$

[16]. Each variable X_i has a nonempty domain of possible values. Each constraint C_i is related to some variables and allows only some combination of assignments between the variables and values in their domains. A state of the problem is defined by an assignment of some, or all, variables to values, such that $\{X_1 = v_1, ..., X_p = v_p\}$ and it is said to be *consistent* if all the constraints are satisfied by the assignments of the given state.

The variables of the CSP for BFE cryptanalysis are obtained from the global dataset. The domain of the variables are BFEs and the constraints are derived from the properties of f. We assemble a CSP for each attribute of the dataset independently because each attribute may be encoded with different encoding parameters (e.g., filter length). More formally, the components of the CSP in this context can be stated as follows:

Variable: Let record $t_i \in G$. Then $t_i.Attr_j$ is a candidate variable for attribute $Attr_j$. Consider the example in Figure 2, values for the forename attribute in G such as "david" and "sam" are candidate variables. The domain of the CSP variables consists of the values obtained from the encoded dataset. Since $D \subseteq G$, only some of the candidates have corresponding encoding in D^T. CSP variables are selected from the candidates according to the frequency of the item sets such that selected candidate has an encoding in D^T with very high probability. The selection procedure is described in Section 3.3.

Domain: Let record $t_i^T \in D^T$. Then $t_i^T.Attr_j$ is a candidate value for attribute $Attr_j$. In Figure 2, the encoded values for the forename attribute, such as "0100000011" and "01111101101" are candidate values for the domain of forename variables. The domains of variables are determined in two steps. In the first step, certain values are eliminated based on the number of bit locations set to 1. With respect to BFEs, each hash function that is applied to encode n-grams sets one bit location in the filter. Therefore, any value in the domain of variable X_i can contain at most $k \cdot z_i$ 1's if X_i contains z_i distinct n-grams. Values with more than $k \cdot z_i$ 1's can be eliminated from the domain of X_i. In the second step, frequency analysis is performed to refine the domains. This procedure is described in Section 3.3.

Constraints: The deterministic behavior of f and the number of hash functions are utilized for defining constraints. When f is applied on a particular n-gram, only particular bit locations are set to 1. When a BFE contains 0's in certain bit locations, Charlie can conclude f does not set those locations for the n-grams that are part of the corresponding string. This assertion is captured in the Theorem 1. Since, in our threat model we assume that k is known by the attacker, the maximum number of bit locations that can be set by f when applied on any n-gram is known to be k. In addition, the minimum number of bit locations that will be set by f can be determined probabilistically as asserted by Theorem 2.

Theorem 1. *Let variable X_i contain n-grams $Q_{X_i} = \{q_1, ..., q_z\}$, and let state $CS = \{X_1 = v_1, ..., X_p = v_p\}$ be a consistent state. If the value of bit location $\bar{l}$ is 0 in v_i, then $f(q_x)[\bar{l}] = 0$ for any $q_x \in Q_{X_i}$.*

Theorem 2. *Let $num1s : \{0,1\}^m \mapsto N$ be a function that returns the number of bit locations with value 1 for the given encoding and w be an integer in range [1,k]. For any n-gram q_i, $num1s(f(q_i)) \geq w$ with probability p such that:*

$$p = \frac{\sum\limits_{i=w}^{k} \binom{m}{i}\binom{k-1}{k-i}}{\binom{m+k-1}{k}} \tag{2}$$

Proof. We refer the reader to Appendix A for the proof of both theorems.

Another important information resource for defining constraints is available assignments in a CSP state which accommodates some knowledge about f. This knowledge is accumulated through a belief function:

Definition 3.2.1 (Belief Function): Belief function $bf : \Sigma^n \mapsto \{0,1\}^m$ is a function, that takes n-grams as input and yields a BFE as output, which simulates f on each n-gram. In its initial state, for any n-gram q_i, it is believed that $f(q_i)$ may set any bit location. The belief function reflects this knowledge by satisfying the equation $bf(q_i)[\bar{l}] = 1$ for $1 \leq \bar{l} \leq m$.

Once a new assignment $\{X_i = v_i\}$ is added to the current state of the CSP, the belief function is modified to satisfy Theorem 1. Basically, the new belief about the encoding of any q_i that is part of variable X_i is obtained by applying the bitwise AND ($\wedge$) operation to current belief $bf(q_i)$ and v_i. Let variable X_i contain n-grams $Q_{X_i} = \{q_1, ...q_z\}$. Then for each $q_i \in Q_{X_i}$:

$$bf(q_i)_{updated} = bf(q_i)_{current} \;\wedge\; v_i \tag{3}$$

Equation 3 extracts the knowledge from the assignment, such that if v_i contains 0 in $\bar{l}$, then $f(q_i)[\bar{l}] = 0$ for an updated state of the CSP.

Example 3.2.1 (Belief Update): An illustration of the belief update process is presented in Figure 3. Initially, $bf(jo) = 11111$. Now, suppose $\{joe = 10001\}$ is added to the current state of the CSP. According to Theorem 1, f does not set the second, third, or fourth bit locations when applied on the bigrams of $\{joe\}$, so we can update $bf(jo)$ to be 10001. After the assignment $\{john = 10110\}$, we learn that the second and fifth bit locations cannot be set to 1 if f is applied on the bigrams of $\{john\}$. Therefore, $bf(jo) = 10000$.

jo → 11111
oh → 11111
hn → 11111
oe → 11111
kn → 11111

(a) initial belief

jo → 10001
oh → 11111
hn → 11111
oe → 10001
kn → 11111

(b) joe = 10001

jo → 10000
oh → 10110
hn → 10110
oe → 10001
kn → 11111

(c) john = 10110

Fig. 3: Belief Update

Belief Validity Constraint (BVC): The belief validity constraint is based on Equation 1 and belief function. Let $CS = \{X_1 = v_1, \ldots, X_p = v_p\}$

be a consistent state. Then all assignments $\{X_i = v_i\}$ should be compatible with the current belief function. Compatible means Equation 1 should be satisfiable with the current belief function and the known assignments. Let X_i contain the n-gram set $Q_{X_i} = \{q_1, \ldots, q_z\}$. Then,

$$v_i = \bigvee_{i=1}^{z} bf(q_i) \quad \forall \{X_i = v_i\} \in CS \tag{4}$$

Example 3.2.2 (BVC Check): Imagine a belief state such that $bf(\_j) = bf(jo) = 10001$ and $bf(oe) = bf(e\_) = 01001$ and suppose $\{joe = 11001\} \in CS$. Now, consider the potential assignment {john = 00111}. If this assignment is performed, then $bf(\_j) = bf(jo) = 00001$. Yet, to satisfy the BVC, $(bf(\_j) \vee bf(jo) \vee bf(oe) \vee bf(e\_) = 11001)$ should hold true, which is not the case. As a result, the assignment $\{john = 00111\}$ is not permitted in this state. □

Min-Location Constraint (MLC): The minimum location constraint is based on Theorem 2 and belief function. Let w be the threshold for the minimum number of bit locations set by f, then we can enforce the following constraint:

$$num1s(bf(q_i)) \geq w \tag{5}$$

If w could be set to a large value, the search space of the CSP solver could be reduced significantly, and the CSP could be solved in a more timely manner. However, according to Theorem 2, w should be set to 1 to satisfy $num1s(bf(q_i)) \geq w$ with probability $p = 1$. By reducing p slightly, w could be increased significantly. At the same time, a reduction in p could lead to an unsatisfiable CSP, since $num1s(bf(q_i)) \geq w$ may not hold with probability $1 - p$. As a result, there is a tradeoff between solving a complex CSP in a timely manner and accepting the risk of converting a satisfiable CSP into an unsatisfiable one. In this setting, the risk is controlled by p (i.e., if p is large the risk is small). Once p is fixed, the threshold w can be calculated via Equation 2.

Example 3.2.3 (MLC Check): Consider the state prior to the potential assignment {joe = 11000}, such that $bf(jo) = 10111$. After the assignment, $bf(jo) = 10000$ and $num1s(bf(jo)) = 1$. If the threshold w is 2, then $num1s(bf(jo)) \geq w$ no longer holds true after the assignment. Therefore, the assignment $\{joe = 11000\}$ is not permitted in this state. □

3.3 Frequency Analysis

In section 3.2, the problem of discovering the mappings between original strings and their BFEs is modeled as a CSP with candidate variables and domains. To select the variables and their domains from candidates, the frequency distribution of the elements in sets G and D^T can be used. CSPs are generally hard problems to solve, however they can be solved in a timely manner by minimizing the domains of the variables and using heuristics based on domain restrictions. Such restrictions can be achieved by leveraging frequency analysis. In particular, we introduce a fair assumption to form the basis of frequency analysis.

Assumption 3.3.1 (g is a 1-1 function): It can be assumed that there is a one-to-one mapping between each distinct string and each distinct BFE. Let S_1 and S_2 be two strings, then

$$\begin{aligned} &g(S_1) = g(S_2), \text{ if } S_1 = S_2 \\ &g(S_1) \neq g(S_2), \text{ otherwise} \end{aligned} \tag{6}$$

It is guaranteed that $g(S_1) = g(S_2)$ when $S_1 = S_2$ by the construction of the encoding. It is highly unlikely that $g(S_1) = g(S_2)$ if $S_1 \neq S_2$. In this context, two strings are defined to be distinct if they have at least one distinct n-gram. Suppose $q_x \in Q_{S_1}$ and $q_x \notin Q_{S_2}$, then $g(S_1) = g(S_2)$ if and only if the Bloom filter check on $g(S_2)$ indicates membership of q_x in S_2. The probability of such false positives (p_f) depends on k, m, and the size of $Q_{S_2}(s)$ such that $p_f = (1 - e^{-ks/m})^k$ (see [17]). Notice that p_f becomes negligible with large k and m. In fact, k and m should be large in a PRL protocol; otherwise record matching quality would degrade significantly. We can derive additional information if we know that g is a 1-to-1 function, such as frequency conservation:

Corollary 3.3.1 (Frequency Conservation): Let fr_i be the frequency of X_i in set D. Then the frequency of $g(X_i)$ in set D^T is also fr_i. Given Equation (6), any string with value X_i will be mapped to the same encoding $g(X_i)$. Strings with values other than X_i will not be mapped to the $g(X_i)$. Therefore, the frequencies are preserved during transformation.

Definition 3.3.1 (Relative Frequency (fr)): Let $freq : \Sigma^* \times Z \mapsto N$ be a function that returns the number of occurrences of x in Z. Then the relative frequency of x is defined as:

$$fr_Z(x) = \left\{ \begin{array}{cc} \frac{freq(x,Z)}{|Z|} \text{ if } & x \in Z \\ 0 & \text{otherwise} \end{array} \right\} \tag{7}$$

If we know that Z is a random sample of size $|Z|$ from G, we can bound the relative frequency of any item in Z using statistics learned from G. This implies we can draw multiple samples of size $|Z|$ from G using Monte Carlo techniques [18] and, for each sample set U from G, we can compute $fr_U(x)$ to determine the frequency intervals in which the true value of $fr_Z(x)$ belongs with high confidence. Using these samples, for any X_i, we can compute α_{X_i} and β_{X_i} such that $\alpha_{X_i} \leq fr_Z(X_i) \leq \beta_{X_i}$ with high confidence. In our problem setting, D is a random sample of size $|D|$ and D^T is the encoded dataset such that $X_i \in D \rightarrow g(X_i) \in D^T$. Now, suppose $\alpha_{X_i} \leq fr_D(X_i) \leq \beta_{X_i}$ for each $X_i \in D$ with 99% confidence, then $\alpha_{X_i} \leq fr_{D^T}(g(X_i)) \leq \beta_{X_i}$ for each $g(X_i) \in D^T$ with 99% confidence by the frequency conservation principle.

In the attack model, Charlie can calculate the relative frequency of items in D^T, but can also learn α_{X_i} and β_{X_i} for any $X_i \in G$. CSP construction could then be finalized using this set of knowledge. Variables of the CSP and their domains could be selected from the candidates (see Section 3.2) as follows:

Variable Selection: Let X_i be a candidate variable. Then X_i could be selected as a CSP variable if and only if $\alpha_{X_i} > 0$. This means that $X_i \in G$ is expected to have the corresponding encoding $g(X_i) \in D^T$ if $\alpha_{X_i} > 0$.

Domain Selection: Let v_i be a candidate value for the domain of X_i, then v_i could be selected for the domain of X_i if and only if $\alpha_{X_i} \leq fr_{D^T}(v_i) \leq \beta_{X_i}$.

Example 3.3.1 (Variable & Domain Selection): Imagine the dataset D^T consists of 20,000 records. Charlie draws a sample dataset of size 20,000 from G multiple times. Given these samples, Charlie obtains an approximate sampling distribution of $fr_Z(X_i)$ for each $X_i \in G$ for the forename attribute. Now imagine $\alpha_{john} = 0.08$ and $\beta_{john} = 0.1$. Since $\alpha_{john} > 0$, 'john' could be selected as a variable. When Charlie receives D^T, he calculates $fr_{D^T}(v_i)$ for each $v_i \in D^T$ for the forename attribute. Suppose $fr_{D^T}(11001) = 0.09$ and $fr_{D^T}(01000) = 0.04$. Then '11001' is in the domain of the variable {john} while '01000' is not because $\alpha_{john} \leq fr_{D^T}(v_i) \leq \beta_{john}$ is satisfied for $v_i = 11001$, but not for $v_i = 01000$.

Clearly, the frequency analysis should also consider possible erroneous records in D because they may affect the relative frequency of items in the dataset. A simple approach to deal with errors is to update the frequency intervals of the variables by considering the possible error rate that can affect the records. Error rates could be determined by domain experts or could be extracted from the historical data [7]. Once the error rates are determined, frequency intervals could be updated accordingly. Let the amount of reduction and increment in the relative frequency of an item be at most err_r and err_i respectively, then $\alpha_{Xi_{updated}} = (1 - err_r) \times \alpha_{X_i}$ and $\beta_{Xi_{updated}} = (1 + err_i) \times \beta_{X_i}$. The effect of the erroneous records on the attack is examined in Section 4.

Notice that if the frequency of all the elements in the global dataset is similar, frequency analysis is not useful to Charlie. In such a case, domain of the variables cannot be reduced. Even worse, the variables cannot be determined since α_{X_i} will be 0 for most of the candidates. However in real life, the frequencies of items tend not to be similar. The distribution of a wide variety of natural and man made phenomena such as frequencies of family names follow power-law distribution [19]. In our problem, such frequent elements in G will have a corresponding encoding in the transformed dataset with high probability. In fact, the proposed variable selection method only allows the selection of such frequent items.

3.4 CSP Solver

Initially, the adversary models the mapping of BFEs to strings as a CSP according to the procedures described in sections 3.2 and 3.3. Once the problem is modeled, a CSP solver is applied to associate BFEs with corresponding strings.

Standard algorithms such as backtracking search [16] could be applied to solve the CSP. The performance of this search can be improved via additional heuristics. One of the most successful heuristics is dom/deg [20], which selects the variable with the smallest *domain* involved in the greatest number of constraints (i.e., maximum *degree*). For the BFEs, constraints are defined over the n-grams of the variables through the belief function. If a variable contains frequently used n-grams, then it is said to be involved in most of the constraints. In this setting, the degree of a variable is the sum of the frequencies of its n-grams.

Definition 3.4.1 (dom/deg): Let $ngramFreq : \Sigma^n \mapsto N$ be a function that returns the number of occurrences of a particular n-gram in all variables of the CSP. Let X_i be a variable in the CSP with domain size $dsize_{X_i}$ such that X_i contains the n-grams $\{q_1, ..., q_z\}$. Then the dom/deg of X_i is defined as:

$$dom/deg_{X_i} = \frac{dsize_{X_i}}{\sum\limits_{i=1}^{z} ngramFreq(q_i)}$$

In any state of the CSP, the variable with the smallest dom/deg is selected as the next variable to assign. The CSP solver applies backtracking search directed by dom/deg to assign variables according to defined constraints. We provide the summary of our CSP solver in Algorithms 1 and 2 in Appendix B.

4 Experimental Results

In this section, we present the experimental evaluation of the proposed attack. To perform our evaluation, we selected a publicly available dataset of real personal identifiers, derived from the North Carolina voter registration list (NCVR), which contains 6,190,504 records [21]. NCVR was used as dataset G from which a random sample of 20,000 records was selected to form dataset D. We investigated the success of the attack with the forename attribute, but note that the attack could be repeated for each attribute. The resulting dataset contained approximately 3,500 unique forenames. To evaluate the effect of typographical and semantic name errors, we also generated a perturbed version of D by implementing a data corrupter based on the errors typically observed in practice [7]. The corrupter introduced errors based on optical character recognition, phonemes, and typography at rates typical of real datasets.

We use precision (i.e., ratio of correctly assigned names to all assigned names) and recall (i.e., ratio of matched names to all available names) as metrics for the attack's success.

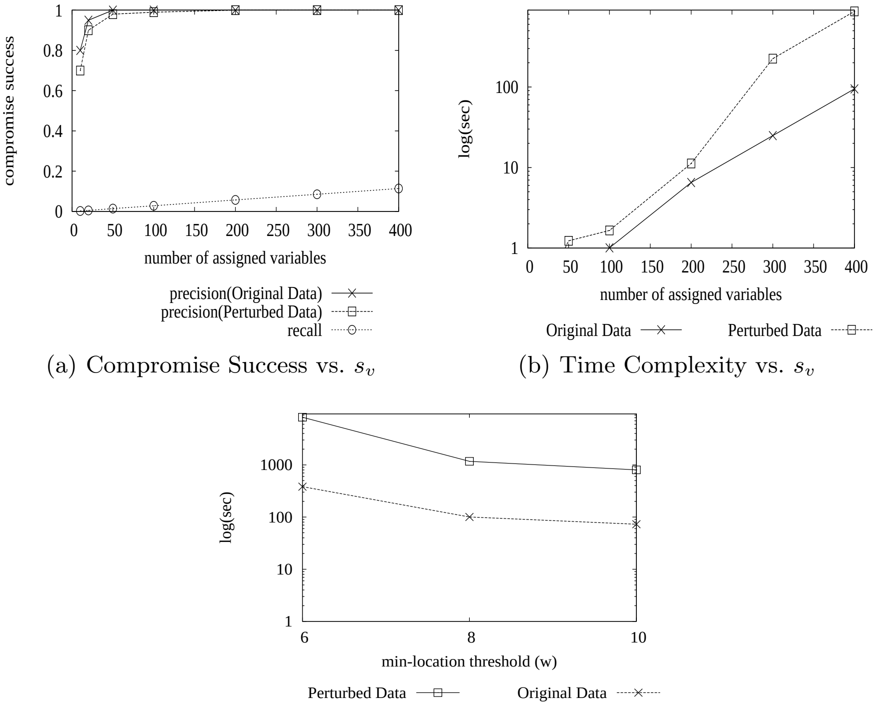
4.1 Attack on BFEs based on Parameters in the Literature

In this part, we evaluate the success of the proposed attack and the effects of the CSP parameters on the computational complexity with a fixed BFE setting. The encoding of D was obtained using the parameters: k (number of hash functions): 15, m (filter length): 500 and n (encoding unit): 2 that are reported in [12]. The effect of varying encoding parameters is then examined in subsection 4.2.

To select CSP variables and domains, frequency intervals were constructed via a Monte Carlo sampling of 10,000 datasets with 20,000 records each from G. The 400 most frequent names in G were selected as the CSP variables according to this analysis ($\alpha_{X_i} > 0$ holds for only 400 names). For the experiments with perturbed data, the frequency intervals were updated according to the data corrupter's error rates, such that $error_r = 0.5$ and $error_i = 0$ (see Section 3.3).

Number of variables (s_v): The effect of s_v is depicted in Figure 4(a) and 4(b). In Figure 4(a) we observe that the recall and precision of the assignments increases with s_v. This is because the constraints of the CSP become stronger with an increasing number of assignments (i.e., belief function updates). Once the constraints are sufficiently strong, only the correct assignment satisfies the constraints. We note that recall has an upper bound of approximately 0.11 (400/3500) due to the outcome of the frequency analysis. In fact, recall of the proposed attack reaches this upper bound along with precision 1.

In Figure 4(b), it can be seen that the complexity of the CSP increases significantly with s_v. This is because, variables with larger domains are added to the CSP as s_v grows, which makes the search space larger. For the perturbed data analysis, the attack becomes more time consuming to execute because the frequency intervals are broadened to compensate for error, which leads to variables with larger domains. For instance, although the assignment of 400 variables is fulfilled in 94 sec. in unperturbed case, it takes 870 sec. for the perturbed dataset.



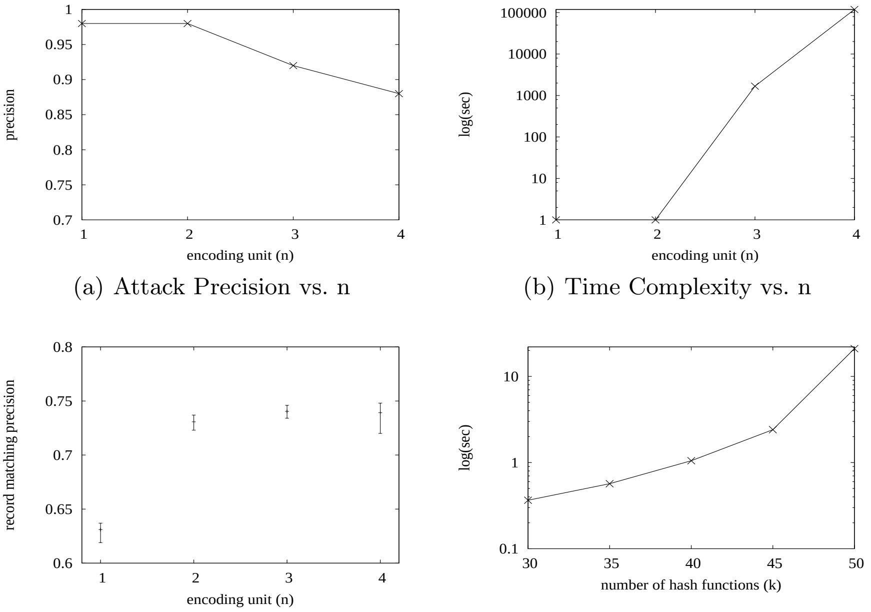
(a) Compromise Success vs. s_v

(b) Time Complexity vs. s_v

(c) Time Complexity vs. w

Fig. 4: CSP Evaluation

Min-Location Threshold (w) : Figure 4(c) illustrates that the CSP's complexity depends on the threshold w associated with the Min-Location constraint (MLC). Specifically, MLC becomes more restrictive as w increases. Therefore, the search space of the CSP solver shrinks, which permits the problem to be



(a) Attack Precision vs. n

(b) Time Complexity vs. n

(c) Record Matching Precision vs. n

(d) Time Complexity vs k: $m = 1000$

Fig. 5: $k = 25$, $m = 1000$ for a, b and c

solved more quickly. However, w can only be increased up to a certain point, after which the CSP becomes unsatisfiable. In this setting, w was initially set to 10 with 0.99 probability (based on Equation 2), and the CSP solver correctly assigned all the variables up to this threshold. The CSP solver could not propose a solution for higher w because, during the pruning of the search space, some of the true mappings were eliminated due to the wrong w constraint.

In summary, the results reported in Figure 4 indicate that proposed attack can compromise 11% of the records with precision close to one in a reasonable amount of time even under the corrupted data scenario.

4.2 Tuning BFE Parameters to Mitigate Attack

The previous experiments show that BFEs are vulnerable when the parameters are set according to recommendations in the literature. However, given that the values of the parameters can be tuned, we set out to determine if the security of BFEs can be strengthened without sacrificing record linkage accuracy. To investigate, we performed a systematic analysis with the number of variables in the CSP fixed to 50. This is a smaller set than the 400 used in the previous experiments, but Figure 4(b) shows that the assignment time grows exponentially with the number of variables, and certain experiments required several days to complete. As a result, for the following experiments, the recall of the attack is fixed to 1.43%. Since the main use of BFEs is to tolerate errors in records during PRL, all experiments were conducted over perturbed data .

***E*ncoding unit:** We note that BFEs gain resistance to the attack as n increases. Specifically, the complexity of the CSP rapidly increases and the precision of the assignments drops off as shown in Figure 5(a) and 5(b). For instance, while 50 assignments were achieved in several seconds for $n = 2$, it required almost 2 days for $n = 4$. Additionally, this was accompanied by a 12% reduction in precision. This is an expected finding because as n grows the constraints become less restrictive. Both BVC and MLC constraints depend on the belief function, such that the more accurate the belief function is, the more restrictive the constraints are. If n becomes larger, the quality of the belief function degrades significantly. This is because records share fewer n-grams as n increases, which leads to less accurate reasoning about the n-gram encoding function.

To characterize the effect of increasing n on record linkage accuracy, we attempted to match the records in the perturbed version of the dataset (P) with the ones in the unperturbed version (O). Once encoding of the datasets was formed (P^T and O^T), each item in P^T was associated with exactly one item in O^T according to similarity between encodings. Similarity was measured with Dice-coefficient [12] which is a set based similarity measure. The experiments, summarized in Figure 5-c, show that as n increases the record matching precision (i.e., ratio of correct associations to all associations) is only slightly affected. These results suggest that large n (e.g., $n = 4$) may provide sufficient record matching accuracy, while significantly reducing the recall rate of the attack and increasing its computational cost.

***B*loom filter length(m) and Number of hash functions(k):** According to our empirical observations, the security of the encoding does not depend on m or k independently, but rather on their ratio. As the m/k decreases, the number of bit locations set by individual n-gram encodings decreases. Therefore, the strength of the constraints that are dependent on the distinction between individual n-gram encodings via belief function diminishes. The complexity of the CSP increases with less restrictive constraints as shown in Figure 5(d). On the other hand, record matching quality degrades if m/k becomes smaller (see [12]). This is because less distinction between n-gram encodings leads to more false positives in the record matching process.

In summary, BFEs become more resistant to the proposed attack with increasing n and decreasing m/k.

5 Related Work

Various protocols for private record linkage (PRL) have been developed [2–5]. PRL protocols tend to use two primary mechanisms for protecting sensitive information: secure multi-party computation (SMC) and data transformation. Although SMC protocols provide strong security guarantees, they are impractical for many real data integration tasks due to their reliance on inefficient cryptography. As an alternative to heavyweight SMC, there are approaches to selectively reveal information through transformation [22–25]. Such approaches perturb, as opposed to encrypt, private information to protect individual identity. Unlike

SMC, transformation can leave data vulnerable to compromise due to the presence of partial information. In fact, several attack models have emerged against transformation techniques. Of note, the disclosure risk of pseudonymization [26] is examined in [22], and the approach is particularly applicable for situations in which the attacker has some background information (e.g, frequency distribution of anonymized items). Another popular approach for data transformation relies on distance preserving data perturbation (e.g., [25]). Disclosure risk of such approaches is examined in [27] and [28]. Their research demonstrated that an adversary can discover the original values with high confidence if mutual distances between data objects is known. While attacks and security investigations have been reported for various transformation methods, to the best of our knowledge ours is the first work to explicitly address BFEs.

In addition to attack scenarios against privacy preserving protocols, information theoretic measures have been proposed to evaluate the degree of the privacy provided by such protocols, especially in the context of anonymous routing ([29], [30], [31]). The quality of the privacy is measured according to the amount of information an attacker can gain after observing the message flow (see [29] and [30]) under various attack scenarios. The quality assessment is extended in [31] by considering the possible prior knowledge of the attackers. Available information theoretic metrics can be used as a basis to evaluate the degree of privacy provided by BFE with different BFE parameter settings. In fact, our work provides a particular attack scenario to enable such analysis for BFE.

6 Conclusions and Future Work

In this paper, we proposed an adversarial model against private record linkage protocols based on Bloom filter encoding (BFE). BFEs are part of an important emerging record linkage model for real world application domains, such as healthcare, because they enable approximate data matching with low computational resources. We modeled the problem of learning the original data from their encoded versions as a constraint satisfaction problem (CSP) using the properties of the encoding function and the frequency distribution of the identifiers in encoded and global unencoded datasets from which the encodings are derived. The unencoded records are assigned to the encoded versions iteratively, according to the constraints. We experimentally evaluated the attack with real data derived from a publicly available voter registration list. We illustrated that the attack can be highly successful if encoding is applied with the settings published in existing literature. However, we also demonstrated that the encodings can be made more resistant to the attack by adjusting encoding parameters with only a slight reduction in record matching quality.

In future work, we plan to extend the attack to determine if additional encodings, or portions of the encodings, can be compromised. In particular, the current CSP is designed to crack high frequency encodings, but items with lesser frequencies may be predictable using the knowledge learned about the encoding function.

APPENDIX

A Proof of Theorems

Proof of Theorem 1: Let us assume $v_i[\bar{l}] = 0$ and $f(q_x)$ set $\bar{l}$ in the Bloom filter to 1. According to the following equation, that is derived from Equation 1

$$v_i[\bar{l}] = \left(\bigvee_{q_i \in (Q_{X_i - q_x})} f(q_i)[\bar{l}] \right) \bigvee f(q_x)[\bar{l}] \tag{8}$$

Notice, $(f(q_x)[\bar{l}] = 1) \rightarrow (v_i[\bar{l}] = 1)$ by the definition of bitwise OR and Equation 8. Since $f(q_x)[\bar{l}] = 1$, it follows that $v_i[l_x] = 1$, which contradicts the initial fact that $(v_i[\bar{l}] = 0) \equiv true$. Therefore, by contradiction, we conclude that $f(q_x)[\bar{l}] = 0$ provided that $(v_i[\bar{l}] = 0)$ is satisfied and $q_x \in QX_i$.

Proof of Theorem 2: Each hash function $hash_i$ for $1 \leq i \leq k$, sets a random bit location in Bloom filter for each input q_j. Consider the set $L = \{1, 2..., m\}$, which contains the bit locations in the Bloom filter, and the multiset $MS = \{l_1, ..., l_k\}$, such that $l_i = hash_i(q_j)$ and $l_i \in L$. MS is a multiset since some hash functions may set same bit locations of filter. MS could be considered as a k-multicombination [32] from set L and the number of all such multisets is:

$$n_1 = \binom{m + k - 1}{k}$$

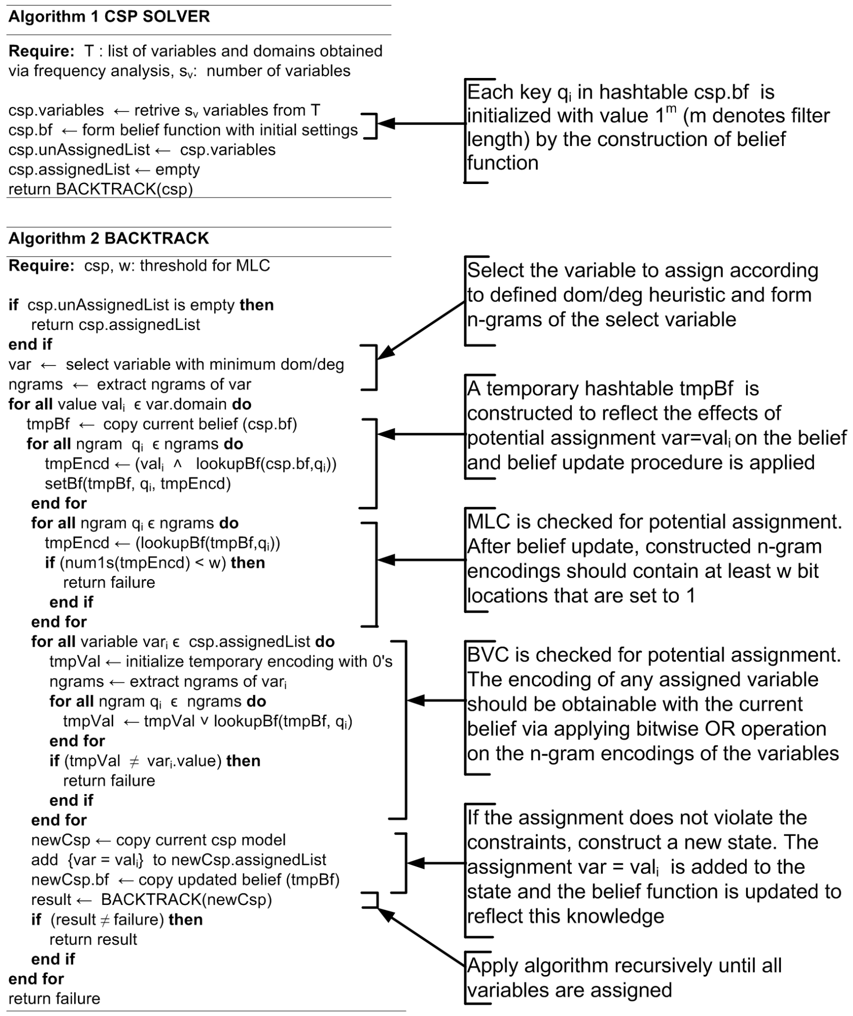
Let MS^w be a multiset of size k from L with exactly w distinct elements, MS_i^w be a set that contains w distinct elements selected from L, and MS_r^w be a multiset that contains $(k - w)$ elements such that $l_i \in MS_r^w \rightarrow l_i \in MS_i^w$. Then $MS^w = MS_i^w \cup MS_r^w$. In other words, multiset MS^w is constructed by selecting w distinct elements from set L and selecting the remaining $(k-w)$ elements from initially selected w elements. MS_i^w is also defined to be w-combination [32] from L and the number of all such sets is $C(\ell, w)$. MS_r^w is $(k - w)$ multicombination from MS_i^w and the number of all such sets is $C(k - 1, k - w)$. Then the number of multisets MS^w is $C(m, w)C(k - 1, k - w)$, since MS^w is the union of MS_i^w and MS_r^w. As a result, the number of multisets of size k from set L with at least w distinct elements is:

$$n_2 = \sum_{i=w}^{k} \binom{m}{i} \binom{k - 1}{k - i}$$

n_1 represents the number of all possible encodings of length m with k hash functions. n_2 represents the number of encodings of length m with k hash functions such that at least w of them set different locations in the Bloom filter. As a result, n_2/n_1 is the probability p such that at least w locations of Bloom filter are set by k hash functions. Since at least w distinct locations are set, there are at least w 1's exist in the corresponding encoding with probability p.

B CSP Solver Algorithm

As described in Section 3.3, the frequency analysis is applied to select the CSP variables and their domains. However, we may want to assign only a portion of these variables to reduce the complexity. In Algorithm 1, s_v items with the smallest domains are retrieved. As described in Section 3.2, the belief function is proposed to simulate the n-gram transformation function. At the implementation level, the belief function could be represented by a hashtable. Each n-gram q_i is a key in this hashtable, and belief about the n-gram encodings are the values[3].



```
Algorithm 1 CSP SOLVER
Require: T : list of variables and domains obtained
via frequency analysis, $s_v$: number of variables

csp.variables ← retrive $s_v$ variables from T
csp.bf ← form belief function with initial settings
csp.unAssignedList ← csp.variables
csp.assignedList ← empty
return BACKTRACK(csp)
```

Each key q_i in hashtable csp.bf is initialized with value 1^m (m denotes filter length) by the construction of belief function

```
Algorithm 2 BACKTRACK
Require: csp, w: threshold for MLC

if csp.unAssignedList is empty then
  return csp.assignedList
end if
var ← select variable with minimum dom/deg
ngrams ← extract ngrams of var
for all value $val_i$ ϵ var.domain do
  tmpBf ← copy current belief (csp.bf)
  for all ngram $q_i$ ϵ ngrams do
    tmpEncd ← ($val_i$ ∧ lookupBf(csp.bf,$q_i$))
    setBf(tmpBf, $q_i$, tmpEncd)
  end for
  for all ngram $q_i$ ϵ ngrams do
    tmpEncd ← (lookupBf(tmpBf,$q_i$))
    if (num1s(tmpEncd) < w) then
      return failure
    end if
  end for
  for all variable $var_i$ ϵ csp.assignedList do
    tmpVal ← initialize temporary encoding with 0's
    ngrams ← extract ngrams of $var_i$
    for all ngram $q_i$ ϵ ngrams do
      tmpVal ← tmpVal ∨ lookupBf(tmpBf, $q_i$)
    end for
    if (tmpVal ≠ $var_i$.value) then
      return failure
    end if
  end for
  newCsp ← copy current csp model
  add {var = $val_i$} to newCsp.assignedList
  newCsp.bf ← copy updated belief (tmpBf)
  result ← BACKTRACK(newCsp)
  if (result ≠ failure) then
    return result
  end if
end for
return failure
```

Select the variable to assign according to defined dom/deg heuristic and form n-grams of the select variable

A temporary hashtable tmpBf is constructed to reflect the effects of potential assignment var=val_i on the belief and belief update procedure is applied

MLC is checked for potential assignment. After belief update, constructed n-gram encodings should contain at least w bit locations that are set to 1

BVC is checked for potential assignment. The encoding of any assigned variable should be obtainable with the current belief via applying bitwise OR operation on the n-gram encodings of the variables

If the assignment does not violate the constraints, construct a new state. The assignment var = val_i is added to the state and the belief function is updated to reflect this knowledge

Apply algorithm recursively until all variables are assigned

Fig. 6: CSP Solver

[3] setBf(H,K,V) sets the value of key K as V, lookupBf(H,K) returns the value for the key K in hashtable H.

C Attack Flow

The attack flow that is executed on BFEs is depicted in Figure 7. In this setting, the adversary attempts to compromise BFEs received from Alice and Bob.

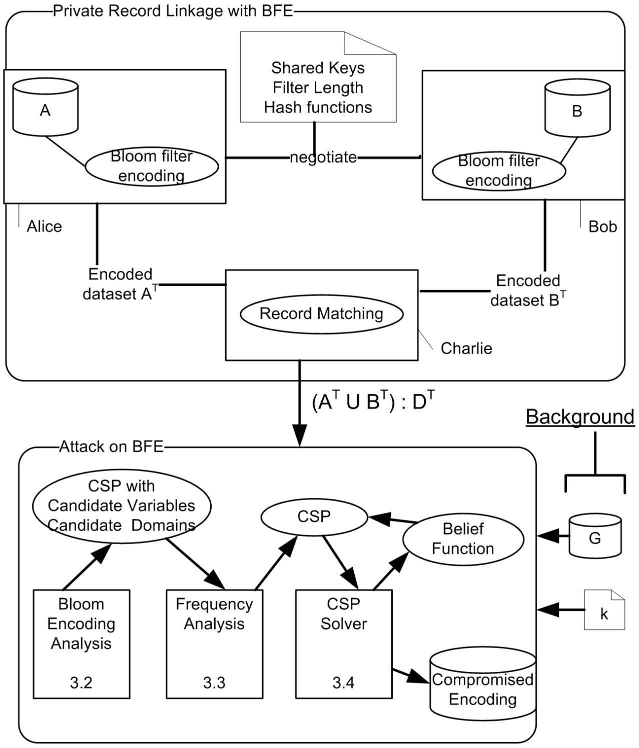


Fig. 7: A schematic of BFE data sharing and the attack issued by the third party

Acknowledgments. This work was partially supported by Air Force Office of Scientific Research MURI Grant FA9550-08-1-0265, National Institutes of Health Grants 1R01LM009989, 2-T15LM07450-06, National Science Foundation (NSF) Career Grant CNS-0845803, and NSF Grants CCF-0424422, CNS-0964350, and CNS-1016343.

Bibliography

[1] Elmagarmid, A., Ipeirotis, P., Verykios, V.: Duplicate record detection: a survey. IEEE Transactions on Knowledge and Data Engineering **19** (2007) 1–16

[2] Churches, T., Christen, P.: Blind data linkage using n-gram similarity comparisons. In: Proceedings of the 8^{th} PAKDD. (2004) 121–126

[3] Clifton, C., Kantarcioglu, M., Doan, A., Schadow, G., Vaidya, J., Elmagarmid, A., Suciu, D.: Privacy-preserving data integration and sharing. In: Proceedings of the 9^{th} ACM SIGMOD Workshop on Data Mining and Knowledge Discovery. (2004) 19–26

[4] Durham, E., Xue, Y., Kantarcioglu, M., Malin, B.: Private medical record linkage with approximate matching. In: Proceedings of the 2010 American Medical Informatics Association Annual Symposium. (2010) 182–186

[5] Inan, A., Kantarcioglu, M., Bertino, E., Scannapieco, M.: A hybrid approach to private record linkage. In: Proceedings of the 24^{th} IEEE International Conference on Data Engineering. (2008) 496–505

[6] Verykios, V., Karakasidis, A., Mitrogiannis, V.: Privacy preserving record linkage approaches. International Journal of Data Mining, Modelling and Management **1** (2009) 206–221

[7] Christen, P., Pudjijono, A.: Accurate synthetic generation of realistic personal information. In: Proceedings of the 13^{th} Pacific-Asia Conference on Knowledge Discovery and Data Mining. (2009) 507–514

[8] Hernandez, M., Stolfo, S.: Real-world data is dirty: data cleansing and the merge/purge problem. Data Mining and Knowledge Discovery **2** (1998) 9–37

[9] Atallah, M., Kerschbaum, F., Du., W.: Secure and private sequence comparisons. In: Proceedings of the 2003 ACM Workshop on Privacy in the Electronic Society. (2003) 39–44

[10] Feigenbaum, J., Ishai, Y., Nissim, K., Strauss, M., Wright, R.: Secure multiparty computation of approximations. ACM Transactions on Algorithms **2** (2006) 435–472

[11] Goldreich, O.: The Foundations of Cryptography. Volume 2. Cambridge University Press (2004)

[12] Schnell, R., Bachteler, T., Reiher, J.: Privacy-preserving record linkage using Bloom filters. BMC Medical Informatics and Decision Making **9** (2009) 41

[13] Lucks, M.: A constraint satisfaction algorithm for the automated decryption of simple substitution ciphers. In: Proceedings of CRYPTO. (1990) 132–144

[14] Bloom, B.: Space/time trade-offs in hash coding with allowable errors. Communications of the ACM **13** (1970) 422–426

[15] Quantin, C., Bouzelat, H., Allaert, F., Benhamiche, A., Faivre, J., Dusserre, L.: Automatic record hash coding and linkage for epidemiological follow-up data confidentiality. Methods of Information in Medicine **37** (1998) 271–277

[16] Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach. 2 edn. Prentice Hall (2003)

[17] Mitzenmacher, M., E.Upfal: Probability and computing: An introduction to randomized algorithms and probabilistic analysis. Cambridge University Press (2005)

[18] Mooney, C.: Monte Carlo Simulation. Sage Publications (1997)

[19] Newman, M.: Power laws, pareto distributions and zipf's law. Contemporary Physics **46** (2005) 323–351

[20] Bessire, C., Regin, J.: Mac and combined heuristics: Two reasons to forsake fc (and cbj?) on hard problems. In: Principles and Practice of Constraint Programming CP96. Volume 1118., Springer LNCS (1996) 61–75

[21] North Carolina Voter Registiration Database: `ftp://www.app.sboe.state.nc.us/data` (2011)

[22] Lakshmanan, L., Ng, R., Ramesh, G.: On disclosure risk analysis of anonymized itemsets in the presence of prior knowledge. ACM Transactions on Knowledge Discovery from Data **2** (2008) 13

[23] Agrawal, R., Srikant, R.: Privacy preserving data mining. In: Proceedings of the 2000 ACM SIGMOD Conference on Management of Data. (2000) 439–450

[24] Kargupta, H., Datta, S., Wang, Q., Sivakumar, K.: Random data perturbation techniques and privacy preserving data mining. Knowledge and Information Systems **7** (2005) 387–414

[25] Chen, K., Liu, L.: Privacy preserving data classification with rotation perturbation. In: Proceedings of the 2005 IEEE Interanational Conference on Data Mining. (2005) 589–592

[26] Pfitzmann, A.: Anonymity, unobservability, and pseudonymity - a proposal for terminology. In: Proceedings of the Privacy Enhancing Technologies Workshop. (2001) 1–9

[27] Liu, K., Giannella, C., Kargupta, H.: An attackers view of distance preserving maps for privacy preserving data mining. In: Proceedings of the 10^{th} ECML/PKDD. (2006) 297–308

[28] Turgay, E., Pedersen, T., Saygin, Y., Savas, E., Levi, A.: Disclosure risks of distance preserving data transformations. In: Proceedings of the 20th International Conference on Scientific and Statistical Data Management. Volume 5069., Springer LNCS (2008) 79–94

[29] Diaz, C., Seys, S., Claessens, J., Preneel, B.: Towards measuring anonymity. In: Proceedings of the 3^{rd} Workshop on Privacy Enhancing Technologies. Volume 2482., Springer LNCS (2002) 54–68

[30] Serjantov, A., Danezis, G.: Towards an information theoretic metric for anonymity. In: Proceedings of the 2^{nd} Privacy Enhancing Technologies Workshop. Volume 2482., Springer LNCS (2002) 41–53

[31] Deng, Y., Pang, J., Wu, P.: Measuring anonymity with relative entropy. In: Proceedings of the 4^{th} International Conference on Formal Aspects of Security and Trust. Volume 4691., Springer LNCS (2006) 65–79

[32] Koshy, T.: Discrete Mathematics with Applications. Elsevier Academic Press (2003)