Compromising Privacy with Trail Re-Identification: The REIDIT Algorithms

Bradley Malin

December 2002
CMU-CALD-02-108

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213-3890

Abstract

Re-identification is the process of relating unique and specific entities to seemingly anonymous data, and as such, is an attack on the privacy of a data collection. This work introduces a new re-identification attack, termed the trail problem, for data distributed over multiple locations. Through the use of data trails an adversary can independently reconstruct the trails of locations that identified entities and their un-identified data visited, which can then employed for re-identification via trail matching. The attack strategy is based on the premise that data collecting institutions partition and release a dataset as multiple subsets, such that one release contains identifying attributes (e.g. name, social security number, phone number) and a second is devoid of these attributes (e.g. DNA sequences). The trail attack is dependent on whether the identified data is always collected with the un-identified data, termed complete, or whether one of the attributes is under-collected, termed incomplete. Both the complete and incomplete trail problems are formalized and several novel algorithms for re-identification are introduced. Examples are drawn from the areas of clickstream, DNA sequence, health, and video data.

Keywords: Re-identification, privacy, data mining, data trails, distributed databases

1 Introduction

In recent times, our society has witnessed a dramatic increase in the ability to collect, record, and store entity-specific information as an individual entity proceeds through their daily routine. This proliferation can be attributed to several factors; most notably the propagation of low-cost storage computing and the development of technologies for relative ease in the computer-based collection and dissemination of various types of data. As a result, the collection of entity-specific data has become ubiquitous in a diverse range of environments, including various levels of government, the medical community, and private corporations [4]. Along with entity-specific data collections comes a vast array of research possibilities at a micro-managed level that up until recently resided beyond the auspices of many communities. In many situations, a collection is useful to researchers and institutions outside of the original collector. Thus, while the application and analysis of such collections may be performed by the collecting institution, where the identities associated with the information is known to the researchers, collections are often sold for profit, shared, or released for public review. When releases of entity-specific collections are made available, the identities of the entities which make up the collection must be protected to prevent the misuse and abuse of such granular and sensitive information corresponding to an entity [6].

Until recently, it was the belief that if data simply looked anonymous, then it actually was anonymous. This assumption is not true and the fallacy has been exposed with respect to de-identified data that, while devoid of explicit identifiers, such as name, address, and phone number, includes additional attributes useful for establishing identity. For example, publicly available hospital discharge databases are devoid of explicit identifiers, yet still incorporate date of birth, gender, and zip code, which in combination is uniquely distinguishing for approximately 87% of the individuals in the United States [30]. The identities of the transactions in such a collection could be determined, with relative ease, by linking the data to an external dataset containing these attributes, such as a voter registration list [29]. Formally, the process by which seemingly anonymous data is related to unique and specific entities, that are the subjects of the data, is referred to as re-identification and is a direct way of compromising the privacy within a data collection. To prevent unwanted re-identifications, computational and statistical privacy protection algorithms and protocols, also known as disclosure control or privacy preserving algorithms, have been developed to prevent unwanted disclosure of various types of data or identities in the data.

However, this work introduces a new re-identification attack strategy not addressed in prior privacy analysis that compromises the protection of identity via the releases of collections from multiple institutions. The basic outline of the attack proceeds as follows. Institutions collecting data record identified transactions. Each institution releases their collection as two subcollections, such that one subcollection contains identifying attributes and the second contains the de-identified set of attributes. For the releases of any single institution, the records of the latter collection can not be uniquely matched to the records of the previous collection. However, an attacker can collect releases from multiple institutions and reconstruct the locations that a particular entity visited, and likewise for unidentified data with a static or traceable component. We refer to such

reconstructions as data trails, and it is through the trails that a relationship is established between identities and their unidentified data via the uniqueness of trails. Such an attack we term the trail privacy problem. This problem is a bit more complex when dealing with institutions that may record/release one type of data (identifiable vs. unidentified) and not the other, which we also address in this paper.

To make the problem more concrete and establish real-world significance, consider an introductory example of a hospital that maintains a database corresponding to the medical records of visiting patients. In such a database, the hospital records contain personally identifying information, such as name, address, and phone number of its visiting patients. Specific medical information about the patients, such as diagnoses provided by attending physicians, is also recorded. For diagnostic or research purposes, DNA can be sequenced and the data is additionally documented in the electronic medical records. The DNA sequence data, devoid of the additional recorded information, might be thought of as anonymous data simply because there are no explicit identifiers. While an individual's DNA sequence may be unique, if it is not accompanied by any explicit demographics, how could the person who is the subject of the DNA be determined? No centralized registry exists that matches up sequence data with the names of individuals. Therefore, if the hospital releases a dataset consisting only of DNA data, the DNA should remain unidentifiable. If the inherent relationships between certain genetic disorders and DNA are not considered (or are vague) this belief would hold when the release of a single hospital is considered. Yet, the individuals who leave behind their sequence at this particular hospital can visit multiple hospitals, and at each hospital, they can leave behind data (via a new sequencing or passing of the medical record), thus creating a trail of data. Each hospital releases 1) a DNA database consisting of sequence data only and 2) a database of diagnosis information, which has the actual identities of individuals, possibly for discharge evaluation or quality assurance analysis. Since the DNA data for an individual is static, one can determine the set of hospitals, or the hospital trail, that each DNA sequence was collected at. Furthermore, it is also possible to determine the hospital trail that an individual visited, based on their directly identifying information. With the constructed trails from the two types of seemingly disparate databases, DNA can be matched to an identity based on uniqueness of the trails. The concept of the trail problem permeates many areas of data privacy, including DNA, clickstream, and video surveillance data.

The remainder of the paper is organized as follows. Section 2 provides background in disclosure control methodologies, which have been developed to protect data when released by collecting institutions and discusses how the trail re-identification problem relates to such schemas. Section 3 defines relational database concepts, definitions, and functions for the trail problem. The trail-based re-identification algorithms, REIDIT-Complete, REIDIT-Incomplete, and REIDIT-Multiple are formalized in sections 4 and 5. An analysis of REIDIT on real world data is provided in section 6, while in section 7 a novel implementation of the algorithm is proposed. Related work in re-identification methods is offered in Section 8. A discussion of the limitations, further development of the algorithms, and its implications is discussed in Section 9.

2 Disclosure Control and Privacy Preservation

Privacy protection models have been addressed in several related fields, including access control, statistical disclosure control, computational disclosure control, and data mining. While each of the communities offers viable models and methods for protecting data, no explicit discussion of the privacy compromising abilities of the trail problem have been addressed. Here fields related to data protection are discussed with respect to the extent that the trail problem does not fall under the protection models.

Traditional access control, also referred to as query restriction, attempts to manage the data that can be provided (released) given a query (request) to a multi-level relational database system (MDB) [5, 7, 12, 13, 19, 22, 24, 26]. Denning and Lunt [7] describe a MDB as a relational database with data classified into a hierarchy of security access levels. For any given query and a security level, the goal is to return a dataset, such that information at the given security level is viewable, but information at a more restricted level of security is obscured and can not be inferred. The main method for protection in a MDB is suppression, where sensitive information, as well as information that can be used to infer sensitive information, is withheld from the release. In the trail problem, at any particular institution, the relationship between the unidentifiable dataset and identifiable dataset is such that there does not exist a direct inference to map the records from one set to the other. There is no protocol to suppress values in the records of a released unidentified registry of data. The suppression of an entire attribute would prevent any release of the dataset when there is only one attribute.

The field of statistical disclosure control (SDC) attempts to protect data utilizing a variety of protection techniques, which are based on the following dogma. The receiver of the released data should be able to reconstruct accurate aggregate distributions, while identities of the record of the entities can not be inferred [8, 10, 14, 17, 34]. Many of the established methods in this community are based on suppression, addition of noise, or perturbing the records in a released collection. The released dataset contains individual records, and while the aggregate statistical distributions can be preserved, the accuracy of the relationships within a particular record can be eroded. Consider that the set of unidentified tables are registries of collected unidentified information, such as IP addresses that visit a particular website. At a particular website, the perturbation of an IP addresses may falsify the information. Perturbing an IP address ip into ip' can protect the identity of ip, but will falsely denote ip' as a visitor of the website. Furthermore, if ip' is listed in a released collection from another website, then a false trail may be established representing multiple addresses.

Computational disclosure control (CDC) techniques attempt to prevent the direct linking of the records from one unidentified collection to the records of an identifiable collection. The methodology of this field protects data by releasing a dataset in a manner such that each record is the same as k-1 other records on a specified set of fields. Records are made to look the same through the generalization and suppression of a predefined set of attributes [27, 28, 29, 33]. Thus, the released table adheres to a level of k-anonymity. The problem of re-identification by linking on

combinations of values from common attributes is exemplified by the work of Sweeney [30]. In this work, publicly available hospital discharge records were purchased from the Group Insurance Commission of Massachusetts. The fields of information included zip code, birth date, and gender which, in combination, often served as unique identifiers for individuals. When this list was linked to identifiable voter registration data on the common fields used for the unique identifier in the discharge data, the identities of the records were uniquely re-identified.

Several proposals for protection have been emerged from the data mining community, usually referred to as privacy preserving algorithms. Certain methods have been proposed that employ perturbation techniques, similar to those of SDC [2, 3, 9, 25]. Other techniques have been offered under the guise of multi-party computation for data distributed over multiple institutions [15, 17, 32]. The work most related to the trail problem is that of the learning algorithms for horizontal- (same attributes - different transactions) [15], and vertical-partitioned (same transactions – different attributes) [32]. In this latter category, no data is released; rather cryptographic approaches are used to learn aggregate association rules. While privacy may be preserved in the latter techniques, neglecting leakage via collusion and other features, no specific information is ever released from an institution, which does occur in the trail problem.

3 Definitions and Identifying Relations

In this section, terminology formal definitions for the trail problem are introduced. The basic definitions used are a derivation of those in relational database theory. The term data refers to entity-specific information, which is organized as a table of rows (records) and columns (fields). Each row of the table is referred to as a tuple and each column is referred to as an attribute. Each attribute can be thought of as a semantic category of information with a set of values. Since this work is concerned with the relationships between tables, let us define a table as $\tau_c(A^c_1, A^c_2,\ldots,A^c_m)$, where the set of attributes for table τ_c is $A^c_\tau = \{A^c_1, A^c_2,\ldots,A^c_m\}$. A tuple t of the table τ_c is defined as $t[a^c_i,\ldots,a^c_j]$ and represents the sequence of values, $v^c_i \in A^c_i,\ldots,v^c_j \in A^c_j$. The size of the table is simply the number of tuples and is represented $|\tau|$.

3.1 Identifying attributes

Attributes may exhibit several different types of identifying properties. An *explicit-identifying* attribute consists of information that reveals the identity of an individual. Examples of such attributes include name, address, and phone number. Alternatively, an attribute may be *quasi-identifying*, such that alone it may not be unique and linkable to external information that contains explicit-identifying attributes, however, when utilized in combination with additional attributes it can be used for such linkage purposes. An example of such an attribute is the date of birth of an individual, which could be used in conjunction with the zip code and gender attributes of a table to uniquely define an individual. The set of attributes, for a particular table, that when used in combination permit linking to external identifying information has been termed a quasi-identifier

[27]. A dataset containing explicit-identifying attributes, or linkable to a dataset that contains such attributes, will be referred to as identified data. Attributes that are neither explicit- or quasi-identifying are referred to as *non-identifying* attributes.

3.2 Linking attributes

To establish a link between two tables τ_i and τ_j, it need not be the case where a quasi-identifier of table τ_i is equal to a quasi-identifier of table τ_j. Rather, it is sufficient to define the quasi-identifier for table τ_i that is useful for linking to table τ_j through the existence of a relationship between the attributes of the two tables. Thus, a set of attribute pairs from tables A and B can be defined, such that each pair consists of one attribute from each table. The following defines the *attribute linkage set*:

Definition 3.1 (Attribute Linkage Set) *Let Q_i and Q_j be the attributes of the quasi-identifiers for tables τ_i and τ_j. The attribute linkage set (ALS_{ij}) is defined as the set of pairs $<A_k,A_l>$ such that $A_k \in Q_i$, $A_l \in Q_j$, and there exists a relation ${}_{Ai}R_{Aj}$ that is non-null.*

Example 3.1 Given two tables, τ_i(*name, date of birth, gender, zip code*) and τ_j(*year of birth, gender, IP address*). Under the assumption that the IP address has not been spoofed, a relationship between the IP address of a computer and the geographic zip code can be established. As such, the linkage attribute set S_{ij} is defined as:

S_{ij}={<*date of birth, year of birth*>, <*gender, gender*>, <*zip code, IP address*>}

3.3 Partitioned tables

Now that the concepts of tables, attribute identifiability, and inter-table relationships have been defined for linkage purposes, the definitions for features specific to the trail privacy problem are presented. For a particular collecting institution, we are concerned with the release of a table as two subtables, one containing identified data and the other devoid of identified data. The subtables are a partitioning of the attributes of the table maintained by the collecting institution. The properties of the partitioned release are formalized in definition 3.2.

Definition 3.2 *Given a table τ maintained by a collecting institution, let τ^- and τ^+ be referred to as the negative and positive subtables of τ, such that*

(i) $A^- \cap A^+ = \varnothing$

(ii) *A^- is devoid of a quasi-identifier that is linkable to an explicit identifier*

(iii) *A^+ includes either*

(a) *explicit identifying attributes*

or

(b) *a quasi-identifier that is linkable to an explicit identifier*

The third concept of definition 3.2 is for re-identification completeness. It states that the positive table is identifiable if it contains any explicit identifying attributes or could have an explicit identifying attribute appended via linkage to external information containing explicit identifying attributes.

3.4 Models of data collection

Re-identification through trails is dependent on the manner in which data is collected. Here several models for data collection are presented. First, several assumptions crucial to understanding the environment upon which the presented version of the REIDIT algorithms function are made evident.

Assumption 3.1 (Per Institution Release) *Each institution c releases data that was collected at c and from no external source.*

Assumption 3.2 (Single Entity Per Tuple) *Each tuple in a table, either original or released, represents one entity only.*

Definition 3.3 (Complete-Collecting Model) *Let C be the set of collecting institutions, let* A^{c+} *be the set of identifying attributes, and* A^{c-} *be the set of non-identifying attributes for institution c. Under the complete-collecting model, every instance of the collection of identifiable data* (+) *is collected with non-identifiable data* (-)*, such that an arbitrary collected tuple t*$[A^{c+}, A^{c-}]$ has non-null values for both A^{c+}, A^{c-}.

Lemma 3.1 (Complete Subtable) *If the collected quasi-identifier is unique for each entity at an institution, then* $|\tau^-| = |\tau^+| = |\tau|$. *The released positive and negative subtables are called complete.*

Let C represent the set of collecting institutions and let $\mathbf{T}^-$ signify the set of all released negative tables provided by the collecting institutions. Since all tables in the set of negative tables are of the same structure, such that $A^{i-}=A^{j-}$ is the same for all $\tau_i^-, \tau_j^- \in \mathbf{T}^-$, there must exist a quasi-identifier that exists in each negative table. This quasi-identifier is referred to as Q^-. Under the per-institution release assumption, the institution itself can be utilized as an attribute in the positive and negative tables. Therefore, location-specific attribute can be appended onto each table, such that the set of attributes for a released subtable is $\{A^{i-} \cup loc_i\}$, where loc_i is what we term the *location-specific* attribute for location *i*. It is not explicitly represented in the released table, however it is implicit due to the fact that the provider of the releasing institution is known. For our proposed attack, the location-specific attribute is permitted to consist of binary values, representing the presence (1) or absence (0) of information collected on an entity at a particular institution. The set of tables with the appended location-specific attributes is termed $\mathbf{T}'^-$. It is obvious that every tuple in the released

table must correspond to information that was collected by the releasing institution, and as such each tuple in the table consists of $t[v^{i-}_1,\ldots,v^{i-}_n,1]$. Similar construction is performed for the generation of $\mathbf{T}'^{+}$.

With the establishment of the fact that Q^- exists, a single table to represent the information from each of the tables in the set $\mathbf{T}'^{-}$ can be constructed. Let us define a new negative table **N** as the union of all tables in $\mathbf{T}'^{-}$ on the attributes in Q^-. The resulting table has the attribute set

$$A^N = \bigcup_{i \in C} A^{i-}$$

Furthermore, |**N**| is equal to the number of distinct quasi-identifier values that exist in the union of all tables in $\mathbf{T}'^{-}$. Each tuple in **N** represents all released information that has been collected about one of the entities with the quasi-identifier in the set of collecting institutions. Similarly, a table **P** for the positive table set $\mathbf{T}'^{+}$ can be constructed.

Within the attribute set A^N is included the set of location-specific attributes and, by the above description, the value of each location-specific is binary. It is from the joined set of location-specific attributes that the data trail is derived from.

Definition 3.4 (Data Trail) *A data trail, henceforth referred to as a trail, corresponds to the values of the attributes for the location specific attributes of a tuple,* $t[a_i,\ldots,a_j]$, $v_i \in loc_i,\ldots,v_j \in loc_j$.

There are several types of trails that may exist. Based on the complete-collecting model, we introduce the first type, which is referred to as the complete trail.

Definition 3.5 (Complete Trail) *A complete data trail consists of values, such that each value is correctly provided without ambiguity; 0 signifies not present and 1 signifies present.*

Lemma 3.2 *If the collected quasi-identifier is unique for each entity at an institution, then for every tuple* $n \in \mathbf{N}$, $\exists\ p \in \mathbf{P}$, such that *trail*(n) = *trail*(p). The function *trail* returns the trail of the provided tuple.

Furthermore, in some cases the partitioned tables may stipulate a non-null attribute linkage set, ALS_{+-}, which might allow a small number of re-identifications without the use of trails. For example, consider the online consumer scenario, where ALS_{+-} = {<zip code, IP address>}. Let X_1 be the set of tuples in $\boldsymbol{\tau}_\mathbf{i}^{+}$ with $t_+[zip]$, *zip* equal to 15213, and let Y_1 be the set of tuples in $\boldsymbol{\tau}_\mathbf{i}^{-}$ that have $t_-[IP \in$ zip 15213]), which returns IP addresses that exist within the zip code 15213. If it is the case that $|X_1| = |Y_1| = 1$, then it must be true that the tuple in set Y_1 can be re-identified with information from the lone tuple within X_1.

There is a second type of collection model, under which there is no guarantee that an entity's data is collected a visited institution. For example, if one type of data, such as IP address, was collected about an online shopper, it is possible that the identifiable information was not recorded due to lack of purchase. It could be equally true, that identified information could be collected more often than

the unidentified information. Consider n example scenario from the healthcare community. A patient can be treated at many hospitals, and at each hospital visited the patient's identifiable data is recorded for purposes of processing insurance reports and simple reference to the patient for physicians, nurses, and other care providers with a legitimate necessity for access. When the patient is thought to harbor a genetic disorder, DNA may be sequenced for diagnostic or research purposes. For a particular patient, the DNA, which is the unidentified data, may not be recorded in every visited hospital's database. The trails that are constructed from such data , where the positive and negative tables may be of unequal size, are what we term *incomplete* trails.

Before a formal definition for incomplete trails is provided, it is imperative to understand the concept of an incomplete table. In the above scenarios, either of the positive or the negative tables could consist of less tuples than the other. For re-identification purposes, it is inconsequential which of the tables are smaller, but for the proposed re-identification algorithm to correctly link tuples across tables it must be the case that all tables of a certain type are not larger than their counterparts.

Definition 3.5 (Incomplete-Collecting Model) *Let C be the set of collecting institutions, let* A^{c+} *be the set of identifying attributes, and* A^{c-} *be the set of non-identifying attributes for institution c. Under the incomplete-collecting model,* values collected for *one of either* A^{c+} *or* A^{c-} *are permitted to be null, such that for all institutions in C, the same attribute is permitted to be null.*

Lemma 3.3 (Incomplete Subtable) *If the collected quasi-identifier is unique for each entity at an institution, then it must be true that for an arbitrary institution c, either* $\forall c \in C$: $|\tau_c^-| \geq |\tau_c^+|$ *or* $\forall c \in C$: $|\tau_c^-| \leq |\tau_c^+|$. *The smaller of the released subtables is termed called incomplete.*

The existence of incomplete information allows for an extension to the above definitions for negative and positive complete tables to incomplete tables.

Definition 3.6 (Incomplete Table) *Let* **N** *and* **P** *be the union of the negative and positive released tables, respectively. The constructed table* **N** *is negative incomplete if* $\forall c \in C$, $|\tau_c^+| \geq |\tau_c^-|$. *The constructed table* **P** *is positive incomplete if* $\forall c \in C$, $|\tau_c^+| \leq |\tau_c^-|$.

Based on the above definitions, an incomplete table must provide less information about the true set of institutions that all entities visited than its complete counterpart. The lack of information can be characterized by the difference in the trails that reside in the tables. While it is not necessarily the case that every trail in the incomplete table must be devoid of information that exists in its corresponding complete trail, it must be the case that there exists a minimum of one trail that contains less information. Such trails that are information lacking are termed *incomplete* trails.

Definition 3.7 (Incomplete Data Trail) *A trail is incomplete if for every attribute in the trail, the following holds true:*

(i) *A value of 1 must be unambiguous; it is definitely a value of 1*

(ii) *A value of 0 is ambiguous; it may represent a value of 0 or 1*

Though an incomplete trail is relatively easy to define, there may be no direct way to match up an incomplete trail with its complete counterpart. There may be multiple complete trails that an incomplete trail can be mapped onto. The ambiguity leads us to the definition of a *subtrail* in Definition 3.8. An incomplete trail is basically a skeleton of a complete trail. The subtrail provides us with a simple relation between incomplete and complete trails.

Definition 3.8 (Subtrail / Supertrail) *An incomplete trail x, is a subtrail of a complete trail y* ($x \leq y$)*, if for every value of 1 in the incomplete trail, the complete trail has a value of 1. Similarly, the complete trail y of this relation is the supertrail of x.*

Example 3.1 Let N_t = {[0,1,0]; [1,1,0]; [0,1,1]} be the set of trail from incomplete table **N** and P_t={[1,1,0]; [1,1,1]; [0,1,1]} be the set of trails from complete table **P**. For a complete trail, trail(p)=[1,1,0], the set of subtrails is {[0,1,0]; [1,1,0]}. For incomplete trail, trail(n) = [0,1,1], the set of supertrails is {[1,1,1]; [0,1,1]}.

4 REIDIT Algorithms

The trail re-identification problem can be defined as follows. Consider a set of subjects S and a set of data collectors C. Each of the collectors $c_i \in C$ releases a partition of their collection $\{d^i_1, d^i_2\}$. An adversary collecting data from multiple institutions appends loc_i to the attribute set of each release. The constructed table from the multiple releases of partitioned tables is equal to:

$$D_1 = \bigcup_{i \in C} d^i_1 \qquad\qquad D_2 = \bigcup_{i \in C} d^i_2$$

From these unioned tables, trails corresponding to the presence or absence of each unique subject are found, as depicted in Figure 1.

$trail_1$	00000000000000000000001000
$trail_2$	00010000001000000000000000
$trail_3$	00000000000000001000000000
$trail_4$	00000000000011000000000000

Figure 1. Sample of constructed household trail data for online household purchasers**.**

Let there exist a relation ${}_{D2}R_{D1}$. Since not all subjects visit all data collectors, a re-identification for a subject $s \in S$ can be made when there exists a unique occurrence of an $s \in {}_{D2}R_{D1}$.

Given the set of partitioned tables, we can employ the uniqueness of the trails to match the seemingly anonymous data to the identified data. The proposed algorithms are variations on what we refer to as Re-identification of Data in Trails (REIDIT). First an informal description of the

algorithms is provided. Following such descriptions, the algorithms are then logically formalized. Notation for the following algorithms is provided in Table 1.

QI^+, QI^-	set of quasi-identifying attributes for $\mathbf{T}^+$, $\mathbf{T}^-$
R	the set of re-identified tuples $t[A_N \cup A_P]$
M_n	set of tuples in table **P** that *trail*(n) is a subtrail of

Table 1. Notation legend.

4.1 REIDIT-Complete

For every tuple n in **N**, we determine if there exists one and only one tuple p in **P**, such that the data trails of the two tuples are equal. Equality is defined by the following feature, when an unidentified trail has a negative value for institution i, the identified trail has a negative value for institution i. The same must hold true for a positive value. Note, that if the attribute linkage set for tables within $\mathbf{T}^-$ and $\mathbf{T}^+$ is nonnull, then these relations must be accounted for as well. When there is an exact matching of tuples, and this matching is unique, then n is re-identified with the explicit identifying information in p. The uniqueness constraint derives from the fact that re-identification can only occur when an unambiguous linkage is possible. If a $trail(n) \in \mathbf{N}$ is equal to a $trail(p) \in \mathbf{P}$, but there is an additional $trail(p') \in \mathbf{P}$ that equals $trail(n)$ as well, then there is an ambiguity and no re-identification can occur for the individual under REIDIT-C.

The formalization of the REIDIT-C algorithm is provided in Figure 2.

Algorithm: REIDIT-C(N, P)

Input: Negative and positive complete tables **N** and **P**
Output: the set of re-identified tuples R

Steps:

let $R = \varnothing$
for each tuple $n \in \mathbf{N}$
  for each tuple $p \in \mathbf{P}$
    $M_n = \varnothing$
    if trail(n) ≡ trail(p)
      $M_n = M_n \cup p$
  if $|M_n| \equiv 1$
    $R = R \cup t[n[a^N - a_L] \cup (p \in M_n)[a^P - a_L]]$
return R

Figure 2. Pseudocode for REIDIT-C.

Complexity. The first step in the algorithm simply appends a single value onto each tuples in the set of tables. This step is linear in the size of the tables, or $O(|\tau_1^-|+|\tau_2^-|+\ldots+|\tau_C^-|)$. Since the number of tuples in each table is maximized when each table contains every known quasi-identifying value, these steps are on the order of $O(|QI|^2)$, where $|QI|$ is the number of unique quasi-identifying values. Similarly, construction of the tables **N** and **P** is also linear in the size of the tables and thus complexity is still $O(|QI|^2)$. In the remaining section of the algorithm, there are two loops to consider. First, the outer loop iterates over all of the tuples in **N**, which is |**N**| iterations. Second, for each iteration in **N**, the algorithm can iterate a maximum of |**P**| times. This maximum is reached

when no re-identifications are made. Furthermore, if each entity has a distinct quasi-identifying value, the second assumption confirms that $|\mathbf{N}|=|\mathbf{P}|=|QI|$. Thus, the maximum number of iterations is $|QI|^2$, and the order of complexity for this section is $O(|QI|^2)$. Since all sections of the algorithm are approximately quadratic in the number of distinct quasi-identifiers, the entire algorithm must be quadratic $O(|QI|^2)$.

Theorem 4.1 *Returned tuples from REIDIT-C are correctly re-identified to one identity.*

PROOF: First, recall the underlying assumption of the complete-collecting model: tuples of both tables **N** and **P** consist only of complete trails. Therefore, at an institution *i*, a visit from an entity must be recorded in both $\mathbf{T}_i^-$ and $\mathbf{T}_i^+$. Since this holds true for every institution, for each $trail(n) \in \mathbf{N}$, there must exist at minimum one equivalent $trail(p) \in \mathbf{P}$. Now, if there exists greater than one equivalent trail in **P** for $trail(n)$, then tuple *n* could be assigned to multiple trails, and subsequently multiple identities from **P**. However, the entity that generated $trail(n)$ could only have generated one trail in **P**. Thus, there is an ambiguity in identity and we can not represent one entity as multiple entities. Yet, if there exists only one equivalent trail for *n* in **P**, then the identity of $trail(p)$ must belong to *n*. ■

4.2 REIDIT-Incomplete

The REIDIT-C algorithm is limited in its application, due to the fact that it is derived from the assumption that the data trails constructed from each set of subcollections are complete. In other words, if an entity left a unidentified type of data at an institution, such as IP address, then the identified type of data, such as name, must also have been collected. As such, a re-identification of unidentified data could only occur if each part of the associated data trail is the same as a data trail for an identified data. Yet, such an assumption of the existence of complete data trails in both sets of released subcollections is not always valid. For example, consider an online consumer who visits several online retail websites before making a purchase. At each website the consumer's IP address is logged, however, the identifying information is only recorded at the website where the purchase is made. The same set of websites may be visited by a different online consumer, but their purchase is made at one of the sites that the first consumer visited without making a purchase. When the data trails are constructed from the released data, there now exists a trail for each data type that is not equal, despite the fact that they correspond to the same entity. How would one make a re-identification if the trails are not the same? If the receiver of the data had omniscience, then one option would be to simply drop an entity's data from a released table if one type of data was collected and on the entity and not the other. This would be ideal, however, in lieu of omniscience, there is no way of determining which entities are missing from an institutions data release.

To circumvent this problem, we introduce REIDIT-Incomplete (REIDIT-I). The main premise of this algorithm is similar to REIDIT-C in that it constructs data trails from a set of released datasets from multiple institutions. However, the re-identification step is contingent on the belief

that when an institution collects one type of data, it may not collect the second type of data. Thus, data trails generated from one set of released tables are always complete, while data trails generated by the second set of released tables can be incomplete, or underreported. When an incomplete trail can be matched to a single complete trail, a re-identification occurs. Given a table of negative trails, **N**, and positive trails, **P,** we wish to determine the identity of the data for which the trails within **N** correspond. In the REIDIT algorithm, the requirement for re-identification was rooted in the assumption that if one trail and only one trail in **P** was equal to the considered trail in **N**, then the considered trail was re-identified by the trail from **P**. However, in the case where trails from one of the databases are incomplete, the equality of trail requirement is revoked. Instead, the requirement of a subtrail is substituted. While the subtrail requirement is less strict a requirement than equality, through an iterative re-identification process, certain ambiguities can be resolved and incomplete trails can be matched to their complete counterparts. For each trail in the incomplete table, the set of trails from the complete table for which the trail is a subset of is determined. If there is only one trail in this set, then the entity of the trail from the incomplete table is re-identified with the identifiable data associated with the trail from the complete table. Also, the re-identified tuples from **N** and the re-identifying tuples from **P** are removed. When, no more re-identifications can be made, the re-identification process is re-iterated. This iterative process continues until either one of two conditions is satisfied; 1) if |**N**| or |**P**| is equal to 0 or 2) there are no re-identifications made in the current iteration.

The formalization of REIDIT-I is provided in Figure 3.

Algorithm: REIDIT-I (N, P)

Input: Negative and positive tables **N** and **P**.
Assumes: **N** is a table of incomplete trails and **P** is a table of complete trails, though the converse could just as easily be considered
Output: the set of re-identified tuples R

Steps

let $R = \varnothing$
for each tuple $n \in \mathbf{N}$
  let $M_n = \varnothing$
  for each tuple $p \in \mathbf{P}$
    if $trail(n) \leq trail(p)$
      $M_n = M_n \cup p$
  if $|M_n| \equiv 1$
    $R = R \cup t[n[a^N - a_L] \cup (p \in M_n)[a^P - a_L]]$
    $\mathbf{N} = \mathbf{N} - n$
    $\mathbf{P} = \mathbf{P} - M_n$
    $R = R \cup$ READIT-I(N,P)
return R

Figure 3. Pseudocode for REIDIT-I.

Complexity. The complexity of the algorithm is best understood by studying an alternative representation of the one in the formal steps provided above. Let Z be a matrix of size $|\mathbf{N}| \times |\mathbf{P}|$, and S be a $|\mathbf{N}| \times 1$ column vector. Let us populate Z with the values 0 and 1 via a simple indicator

function, such that if $trail(n_i) \leq trail(p_j)$ set cell $Z(i,j)$ to 1, otherwise set it to 0. Let $S(i)$ represent the rowsum of the i^{th} row of Z. Completion of this process is approximately $O(|\mathbf{N}||\mathbf{P}|)$. To determine if a re-identification has occurred for the i^{th} entity, $S(i)$ is checked to see if it equals 1. If there is a value of 1, then n_i is re-identified by linking with p_j and remove the i^{th} row and j^{th} column. Let r_x be the number of entities re-identified in the x^{th} iteration. To prepare for the next iteration, all cells of S are decremented by r_{xj}. The maximum number of iterations of the algorithm occurs when $r_1 = r_2 = \ldots r_{|\mathbf{N}|} = 1$. Under such conditions, the total number of cell checks is $|\mathbf{N}|+|\mathbf{P}|$ in the first iteration, $(|\mathbf{N}|-1) + (|\mathbf{P}|-1)$ in the second iteration, and $(|\mathbf{N}|-a) + (|\mathbf{P}|-a)$ required by the a^{th} iteration. So, for all iterations, the number of cell checks (operations) is:

$$\#\ operations = \sum_{i=0}^{|\mathbf{N}|-1} \left(|\mathbf{N}| + |\mathbf{P}| - 2i\right) = |\mathbf{N}||\mathbf{P}| + |\mathbf{N}|$$

The re-identification process is dependent on both $|\mathbf{N}|$ and $|\mathbf{P}|$. Complexity is $O(|\mathbf{N}||\mathbf{P}|)$. Overall, the total algorithm complexity of the REIDIT-I is O(matrix construction) + O(re-identification), which is $O(|\mathbf{N}||\mathbf{P}|) + O(|\mathbf{N}||\mathbf{P}|) = O(|\mathbf{N}||\mathbf{P}|)$.

Theorem 4.2 *Returned tuples from REIDIT-I are correctly re-identified to one identity.*

PROOF: For convenience, let us assume that **N** is an incomplete table and **P** is a complete table. Under definition 3.7, there are no false 1's in an incomplete trail, so it must be true that for an arbitrary tuple $n \in \mathbf{N}$, there must exist a non-null set of supertrails M_n ($|M_n| \geq 1$) for $trail(n)$. If $|M_n|$ is equal to 1, then there exists only one complete trail that could be reconstructed from $trail(n)$ through the replacement of 0's with 1's. Therefore n is re-identified by the tuple in M_n. In the event when $|M_n| > 1$, then the algorithm can still converge to a correct re-identification. This claim is quite simple and straightforward to prove. Let $|M_n|$ equal k. When a re-identification is made for a tuple other than n, then $|M_n|$ decreases by 1. And since it is already known that $|M_n|$ has a minimum of 1, if $|M_n|$-1 re-identifications are made for tuples of N, excluding n, each with a tuple from M_n, then the remaining tuple from M_n must re-identify n. ■

4.3. REIDIT-Multiple

An entity may leave different values behind for identified or unidentified data. In the event that the collection model is ICM, the REIDIT-I model can be augmented to re-identify multiple data sources to a unique entity. The main assumption of the REIDIT-I is that each complete trail can have a maximum of one subtrail. Yet, when an individual can leave behind multiple data vales for the same attribute is not necessarily true. For example, a computer can be used by multiple individuals in a shared setting, such as a household, or there may be multiple DNA sequences that belong to the same individual. The REIDIT-Multiple (REIDIT-M) algorithm relaxes the assumption that there must be a maximum of one subtrail per complete trail. Thus, if an incomplete trail is a subtrail of only one supertrail, then a re-identification occurs via a linkage between these two trails. Furthermore, multiple subtrails can map to the same supertrail and permit a re-identification. The output of the following algorithm is in the same format as for the previous READIT variations.

The formalization of the REIDIT-M algorithm is provided in Figure 4.

Algorithm: REIDIT-M (N, P)

Input: Negative and positive tables **N** and **P**
Assumes: 1) N is a table of incomplete trails and **P** is a table of complete trails, though the converse is equally feasible
2) Multiple subtrails can be derivative of the same supertrail
Output: the set of re-identified tuples R

Steps

let $R = \varnothing$
for each tuple $n \in \mathbf{N}$
  let $M_n = \varnothing$
  for each tuple $p \in \mathbf{P}$
    if $trails(n) \leq trails(p)$
      $M_n = M_n \cup p$
  if $|M_n| \equiv 1$
    $R = R \cup t[n[a^N - a_L] \cup (p \in M_n)[a^P - a_L]]$
return R

Figure 4. Pseudocode for REIDIT-M.

5 Theoretical vs. Actual Re-identification

In theory, an exact relationship between the number of entities that will be re-identified given the number of collecting institutions can not be established *a priori*, due to the reality that different entities access various institutions according to their own specific needs and constraints. Nonetheless, the theoretical maximum number of entities that can be re-identified given the number of institutions can be determined. For both READIT-C and READIT-I, the maximum number of re-identifications is dependent on the number of permutations of a binary string. Therefore, given the a set of subjects S and a set of collecting institutions C, if $|S| \leq |C|$, then the maximum number of re-identifications is bounded by the number of subjects $|S|$, which implicates that all trails may be re-identified. When $|S| > |C|$, the maximum number of re-identifications is bounded by the number of institutions in an exponential manner as $2^{|C|}$-1. Thus, when $|S| > 2^{|C|}$, it will be impossible to re-identify all trails. In contrast, for READIT-M, the number of re-identifications is independent of the number of institutions, since it is possible for multiple identified trails to be mapped to a single unidentified trail. As such, the maximum number of re-identifications is $|S|$.

While the exact number of re-identifications can only be determined through application of the appropriate READIT algorithm, there is evidence that a probabilistic model, such as a multinomial function over each institution, can be used to estimate the likelihood of a particular trail be re-identified [19]. The intuition behind the model is based on the fact that, while an exponential number of trails may be constructed, only a fraction of the trails are ever observed. Under the proposed model, the probability of observing an audit trail is dependent on the number of individuals at each institution. This theory is supported by empirical evidence of the uniqueness

and re-identifiability of DNA sequence data trails constructed from information collected from patients with particular genetic disorders [19, 20].

6 Re-identification of IP Addresses

To evaluate the REIDIT algorithms, we determine the re-identifiability of IP addresses from real-world online transaction data. The dataset chosen for this analysis was compiled by the Homenet project group at Carnegie Mellon University[1] [15], who provide families in the Pittsburgh area with internet service via Carnegie Mellon in exchange for the monitoring and recoding of the families' online services and transactions. Our analysis is conducted on URL access data collected over a two month period that includes 86 households. Since we are interested in re-identifying IP addresses to the entities using the computers, we chose to reconstruct purchase data and weblogs for websites accessed by this population. During this time, 5116 distinct websites and 66,862 distinct pages were accessed. The URL data was manually labeled as "purchase made" or "purchase not made" for each accessed page. For example, a purchase confirmation URL at Greyhound.com was labeled as a purchase, while the frontpage of the website was labeled as not being a purchase. It was determined that purchases were made at 28 distinct websites, including Amazon.com, Ticketmaster.com, and Hotwire.com**.** We make the assumption that websites collect two types of data: 1) identifying information, such as name or address of the purchaser and 2) the IP address of computers visiting their site. The websites release information partitioned into these two types of data, and as such τ_c^- (**N**) is the IP address log list and τ_c^+ (**P**) is the list of purchasing identities.

6.1 Experiment 1: Complete Collection Re-identification

In the first experiment we assume the following data release model. Each website provides consumer lists of the individuals and their mailing address who made a purchase at the website, a table **P**={*website, name, address*}, for market analysis and direct marketing. In addition, each site separately provides a list of the IP address of customers who purchased over the web; a table **N**={*website, purchaser IP address*}. Under such a data collection model, we employ the use of REIDIT-C for re-identifying IP addresses to households. There were 26 households that made purchases at a total of 28 websites. Of these trails, 16 IP addresses (~62%) were re-identified to mailing address.

In this case, the reason for providing the list was for market analysts to learn what stores had visits from the same customer. However, this experiment demonstrates that the IP address can be re-identification in some cases, thereby compromising the geographic privacy of the IP address.

[1] For additional information about the Homenet project, we refer the reader to http://homenet.andrew.cmu.edu.

6.2 Experiment 2: Incomplete Collection Re-identification

The scenario for the second experiment, considers different data release model. Under this model, each website releases the list of customers who made purchases over the web, **P**={*website, name, address*}. Websites also separately provide lists of IP addresses of all visitors to their site, both purchasers and non-purchasers; **N**={*website, IP address*}. As such, **N** provides complete trails and **P** may provide incomplete trails list. The strategy in this experiment is to find a visit trail that can account for a distinct purchase trail, and for which no other trail can account for. By account for, it is meant that 1) if no visit, no purchase in the trail, and 2) if a visit, a purchase may or may not be present. The re-identification of IP addresses to households to is attempted through REIDIT-I. For this experiment, there were 26 households that made purchases, however, IP address data was released for all 86 households. Through REIDIT-I, 9 IP addresses (~35%) were re-identified to mailing address.

6.3 Experiment 3: Multiple Identity Re-identification

For certain households there exist multiple users of a particular computer. In such a setting, it is possible for an IP address to be associated with multiple identified entities. Here, we consider a model in which each website releases a list of customers who made a purchase at the website, where the list includes the email address, but not the mailing address; **P**={*website, email address*}. Websites also separately release lists of all visiting IP addresses; **N**={*website, IP address*}. The strategy of this experiment is to find a visit trail that can account for one or more distinct purchase trails, and thereby associate multiple email addresses to the same IP address. So, in this experiment, a person-specific list of purchasers is mapped to a household-based list of web visitors. In particular, the IP address listing for this experiment is the same as the one from the incomplete collection re-identification experiment.

There were 23 households consisting of a single purchasing individual and 3 households consisting of 2 distinct purchasing individuals, for a total of 29 purchasing individuals. In total, there were 144 online individual visit trails. Re-identification was achieved for all three households. It is interesting to note, that the households re-identified were unresolved via REIDIT-I. This result is due to the fact that in the Homenet dataset, the family members visited common sites, which under REIDIT-I remain ambiguous at the individual level, yet, at the household level became distinct.

7 REIDIT as a Fraud Detection Tool

The READIT algorithms demonstrate that the identities of complex data types, such as IP address and DNA sequence data, are not sufficiently protected by a simple partitioning of explicit identifying features from complex data. While the presented methods of trail construction and re-identification exemplify how privacy can be compromised, there exist additional non-malicious

applications of the concepts and algorithms. Here, the READIT algorithms can be utilized for applications devoid of malicious intent, such as fraud detection. The goal of fraud detection is to determine when an anomalous event is occurring for a particular entity. Since the goal of the READIT algorithms is to determine outlying features relating to an individual and their behavior (or dropping of their data), the algorithms can provide assistance in automating the process of certain types of anomaly detection. Consider the problem of a malicious individual who engages in identity theft, where one individual assumes the identity of another individual. Similarly, our detection model can determine if an individual has assumed multiple identities. Let the set of collecting institutions be retail stores who keep video surveillance on their consumers. Let τ_c^- be the set of distinct faces distilled from the video surveillance data from store c and let τ_c^+ be the set of credit card purchases at the same store. Under this collecting model, the incomplete data trails are the set of faces and complete trails will are constructed from the credit card purchases. In other words, an individual must be at a store to purchase an item, yet while in the store the individual does not necessarily make a purchase. When the READIT-M algorithm is used to re-identify faces to cards, it is possible that one may find multiple credit cards corresponding to the same face. Such an event would be expected if related individuals use the same card, such as individuals from the same family, but when a relation can not be established, then a possible occurrence of identity theft or fraud may be taking place.

8 Related Re-identification Methods

In addition to trail re-identification, there have been several methods applied to the problem of re-identification. Mainly, the concept has been developed with respect to three main genres: record linkage, data linkage, and aggregation operations. The techniques of record linkage were initially introduced by Newcombe [23], Felligi, and Sunter [11] and have been ushered into the modern statistical era by the work of Winkler [35]. The problem that record linkage attempts to solve is how to automate the updating of two lists or the deduplicating of a single list. To do so, several assumptions are made about the data. First, it is assumed that there are two files with common variables and that there is typographical error in the files. The process of record linkage corresponds to building a statistical model to group pairs for records into definite matches, definite non-matches, and pairs that need clerical review. Currently, record linkage methods employ expectation-maximization algorithms for converging to classifications of record pairs. The process is not intended for compromising privacy, but rather to relate records of an individual for which minor corruption in one or both of the records has occurred. While the technique does relate the records of a particular subject, for the most part, record linkage has not been associated with associating unidentified data to identified data.

Data linkage differs from record linkage in several fundamental ways, most notably the fact that data linkage has been specifically designed for re-identification purposes. It is the intension of data linkage to make re-identifications for data devoid of an identity. In addition, the attributes of the two files are not required to be the same, but instead it is concerned with exploiting

inferential relations between attributes of the two files. A combination of the values in the attributes of a table is utilized to estimate the uniqueness of an entity's identity in a known population, beyond that of the considered files [30]. The addition of related attributes allows for an increased probability for the uniqueness of records, provided the added attributes can be related to features of the identified population. Yet still, such linkage is established through known attributes. When the uniqueness of a record can not be established based on related attributes, the re-identification process ceases for the considered record. Trail re-identification is most related to data linkage, however, it extends such a procedure into a simultaneous evaluation of a large number of tables.

The third method of re-identification, ordered weighted aggregation (OWA) operators [31], is rooted in the data mining community. While record and data linkage require that there are direct inferential relationships between attributes of two tables, this problem attempts to re-identify when there are no common attributes. However, the problem makes several large assumptions. First, there is an assumption that there are a large number of common individuals in the two datasets. Second, there is an implicit similar structure to information in the two tables. Third, the datasets consist of numerical data. The procedure takes a table of records and attempts to do dimensionality reduction by converting the data vector of a record into a weighted scalar that captures a relatively large quantity of the information in the original vector. Currently, the technique creates several scalars, resulting from different parameterizations of the aggregation operator. Re-identification is then achieved by ordering the table and matching records that have similar resulting scalars. The technique has been demonstrated to work well with the re-identification of attributes, where the data vectors are the values of an attribute for all records. However, while the claim has been made that this technique can re-identify individual records in a table, corresponding to subjects and not attributes, no current research disputing or proving this claim exists.

9 Concluding Remarks and Future Research

The READIT algorithms provide deterministic methods for discovering how re-identifications can occur given independently released datasets from a set of collecting institutions. The methodology is based on the construction of the set of institutions where an individual left behind their data, or the data trails. At this time, the algorithms are designed to re-identify binary data trails from data where, at maximum, one of the collected data types is incomplete or undercollected. However, when the possibility exists for incomplete data trails to be in both the unidentified data trails and the identified data trails, or for there to be error in the databases (i.e. the patient name was falsely recorded) the READIT algorithms converge to incorrect re-identifications. Thus, one possible extension of this research is in the design and evaluation of models that allow for the probabilistic qualification of data trail bits of the data trail. This qualification would permit an interesting optimization problem for re-identification, where a bit value of 1 or 0 corresponding to one collecting institution may provide more information about a trail than another institution. Future research will need to address such issues of error and multiple types incomplete data.

In addition to compromising privacy, the REIDIT algorithms provide a technique for evaluating the strength of privacy protection schemas. Currently, there is no work documenting how particular protection schemas, such as *k*-anonymity or perturbation, affect the various trail re-identification methods presented above. If it is found that the current protection schemas do protect against one technique, for instance the complete trail problem, further attention must be devoted to considering protection against incomplete or multiple trail problems? Thus, since trail re-identification is a novel strategy for re-identification, it provides a new and important direction for future research in data privacy.

Acknowledgements

The author would like to thank Latanya Sweeney for all of her guidance and advising in this project, as well as the rest of the researchers at the Laboratory for International Data Privacy, including Yiheng Li and Elaine Newton. Additional thanks are extended to Alan Montgomery, Robert Kraut, and the Homenet project group at Carnegie Mellon University for the use of their data.

References

[1] N.R. Adams and J.C. Wortman. Security-control methods for statistical databases: A comparative study. *ACM Computing Surveys*, 21:515-556, 1989.

[2] D. Agrawal and C.C. Aggarwal. On the design and quantification of privacy preserving data mining algorithms. *In ACM SIGACT-SIGMOD-AIGART Symposium on Principles of Database Systems*, pages 247-255, Santa Barbara, California, May 21-23 2001.

[3] R. Agrawal and S. Ramakrishnan. Privacy-preserving data mining. *In ACM SIGMOD*, Dallas, TX. 439-450, May 14-19, 2000.

[4] M. Behr. All eyes are on you: Who's tracking the digital DNA that you scatter? *Popular Science*. 261(1): 48-55, 2002.

[5] S. Castano, M.G. Fugini, G. Martella, and P. Samarati. *Database Security*. Addison Wesley. 1996.

[6] C. Clifton and D. Marks. Security and privacy implications of data mining. In *ACM SIGMOD Workshop on Data Mining and Knowledge Discovery*, Montreal, Canada, 1996.

[7] D. Denning and T. Lunt. A multilevel relational data model. *In Proc of the IEEE Symposium on Research in Security and Privacy*, Oakland, California, 220-234, 1987.

[8] G.T. Duncan, T. Jabine, and V. de Wolf. Private lives and public policies: *Confidentiality and accessibility of government statistics.* Washington, DC: National Academy Press, 1993.

[9] A. Evfimievski, R. Srikant, R. Agrawal, and J. Gehrke. Privacy preserving mining of association rules. *In ACM SIGKDD*. Edmonton, Alberta, Canada, June 2002.

[10] G.T. Duncan and S. Fienberg. A Markov perturbation method for tabular data, turning

administrative systems into information systems, 1996 - 1997, *IRS Methodology Report Series 5*, (J. Dalton and B. Kilss, eds.), 223-231, 1997

[11] I.P. Fellegi and A.B. Sunter. A theory for record linkage. *Journal of the American Statistical Association.* 64: 1183-1210, 1969.

[12] T. Garvey, T. Lunt, and M. Stickel. Abductive and approximate reasoning models for characterizing inference channels. *IEEE Computer Security Foundations Workshop*. 4, 1991.

[13] T. Hinke. Inference aggregation detection in database management systems. *In Proc of IEEE Symp. on Research in Security and Privacy*. Oakland, California, 96-107, 1988.

[14] A. Hunderpool and L. Willenborg. μ- and τ-Argus: Software for statistical disclosure control. *In Third International Seminar on Statistical Confidentiality*, Bled, 1996.

[15] M. Kantarcioglu and C. Clifton. Privacy-preserving distributed mining of association rules on horizontally partitioned data. *In ACM SIGMOD: Workshop on Research Issues in Data Mining and Knowledge Discovery (DMKD)*. Madison, Wisconsin, June 2, 2002.

[16] R. Kraut, S. Kiesler, B. Boneva, J. Cummings, V. Helgeson, and A. Crawford. Internet paradox revisited. *Journal of Social Issues.* 58: 49-74, 2002.

[17] R. Kumar. Ensuring data security in interrelated tabular data. *In Proc. of the IEEE Symposium on Security and Privacy*, 96-105, Oakland, CA, May 1994.

[18] Y. Lindell and B. Pinkas. Privacy preserving data mining. *In CRYPTO*, pages 36-54. Springer-Verlag, August 20-24 2000.

[19] T. Lunt. Aggregation and inference: Facts and fallacies. *In Proc of IEEE Symp on Security and Privacy.* Oakland, CA, 102-109, May 1989.

[20] B. Malin and L. Sweeney. *Re-identification in population-based DNA database collections*. Tech. Report CMU-CS-02-189. School of Computer Science, Carnegie Mellon University, Pittsburgh, PA. December 2002.

[21] B. Malin and L. Sweeney. Re-identification of DNA through an automated process. *In Proc AMIA Symp*. 423-427, November 2001.

[22] M. Morganstern. Security and inference in multilevel database and knowledge based systems. *In Proc of ACM SIGMOD*, 357-373, 1987.

[23] H.B. Newcombe, J.M Kennedy, S.J. Axford, and A.P. James. Automatic linkage of vital records. *Science*. 130: 954-959, 1959.

[24] X. Qian, M. Stickel, P. Karp, T. Lunt, and T. Garvey. Detection and elimination of inference channels in multilevel relational database systems. *In Proc of IEEE Symp on Security and Privacy.* Oakland, CA,196-205, 1993.

[25] S.J. Rizvi and J.R. Haritsa. Maintaining data privacy in association rule mining. *In VLDB*, Hong Kong, China, July 23-26, 2002.

[26] T. Su and G. Ozsoyoglu. Controlling FD and MVD inference in multilevel relational database systems. *IEEE Trans On Knowledge and Data Engineering*. 3: 474-485, 1991.

[27] L. Sweeney. Guaranteeing anonymity when sharing medical data, the Datafly system. *In Proc Journal of the American Medical Informatics Association,* Washington, DC: Hanley

& Belfus, Inc., 1997.

[28] L. Sweeney. Achieving k-anonymity privacy protection using generalization and suppression. *To appear in the International Journal on Uncertainty, Fuzziness, and Knowledge-based Systems*. 10 (7), 2002.

[29] L. Sweeney. K-anonymity: A model for protecting privacy. *To appear in the International Journal on Uncertainty, Fuzziness, and Knowledge-based Systems*. 10(7), 2002.

[30] L. Sweeney. Uniqueness of simple demographics in the U.S. population. LIDAP-WP4. Laboratory for International Data Privacy, Carnegie Mellon University.

[31] V. Torra. Re-identifying individuals using OWA operators. *In Proceedings of the 6th International Conference on Soft Computing*. Iizuka, Fukuoka, Japan. 2000.

[32] J. Vaidya and C. Clifton. Privacy preserving association rule mining in vertically partitioned data. *In ACM SIGKDD*. Edmonton, Alberta, Canada, July 23 - 26, 2002.

[33] S.A. Vinterbo, L. Ohno-Machado, and S. Dreiseitl. Hiding information by cell suppression. In *Proceedings of the American Medical Informatics Association Annual Symposium*. 726-30, 2001.

[34] L. Willenborg and T. de Waal. *Statistical disclosure control in practice.* New York. Springer-Verlag. 1996.

[35] W.E. Winkler. Matching and record linkage. *In Business Survey Methods*, New York, J. Wiley. 355-384.