PRESERVING DATA PRIVACY AND INFORMATION USEFULNESS FOR RFID DATA PUBLISHING

KHALIL AL-HUSSAENI

A THESIS

IN

THE CONCORDIA INSTITUTE FOR INFORMATION SYSTEMS ENGINEERING

PRESENTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF MASTER OF APPLIED SCIENCE IN INFORMATION SYSTEMS SECURITY

CONCORDIA UNIVERSITY

MONTRÉAL, QUÉBEC, CANADA

SEPTEMBER 2009

Your file Votre référence
ISBN: 978-0-494-63082-2
Our file Notre référence
ISBN: 978-0-494-63082-2

Abstract

Preserving Data Privacy and Information Usefulness for RFID Data Publishing

Khalil Al-Hussaeni

Radio-Frequency IDentification (RFID) is an emerging technology that employs radio waves to identify, locate, and track objects. RFID technology has wide applications in many areas including manufacturing, healthcare, and transportation. However, the manipulation of uniquely identifiable objects gives rise to privacy concerns for the individuals carrying these objects. Most previous works on privacy-preserving RFID technology, such as EPC re-encryption and killing tags, have focused on the threats caused by the physical RFID tags in the data collection phase, but these techniques cannot address privacy threats in the data publishing phase, when a large volume of RFID data is released to a third party. We explore the privacy threats in RFID data publishing. We illustrate that even though explicit identifying information, such as phone numbers and SSNs, is removed from the published RFID data, an attacker may still be able to perform privacy attacks by utilizing background knowledge about a target victim's visited locations and timestamps. Privacy attacks include identifying a target victim's record and/or inferring their sensitive information. High-dimensionality is an inherent characteristic in RFID data; therefore, applying traditional anonymity models, such as K-anonymity, to RFID data would significantly reduce data utility. We propose a new privacy model, devise an anonymization algorithm to address the special challenges

of RFID data, and experimentally evaluate the performance of our method. Experiments suggest that applying our model significantly improves the data utility when compared to applying the traditional K-anonymity model.

Acknowledgments

I would like to take this opportunity to express my deepest gratitude to my supervisor, Dr. Benjamin C. M. Fung, for his much-appreciated guidance and continuous support at every step throughout my work on this research. His enthusiasm inspired me, and without his valuable suggestions and critical remarks, the completion of this work would not have been possible.

I would like also to say that no matter what words I use in expressing my profound appreciation, I cannot thank my dearest beloved parents and brothers enough for their compassionate unconditional love and support at every stage during my study at Concordia University. In return, I can only offer them the chance to be proud of what I have achieved, and of course, my unconditional love.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Radio Frequency IDentification (RFID) is a prevailing technology for automatically identifying objects in an efficient fashion. RFID technology is composed of two entities; tags and readers. A tag is a small device that can be attached to a manufactured object or to an item carried by someone for the purpose of uniquely identifying that object or that person. A reader is an electronic device that retrieves the information stored in an RFID tag by broadcasting a radio signal. Figure 1 illustrates a typical RFID information system, which is composed of a large collection of tags and readers, and can manage huge amounts of RFID data. The reader scans a tag by emitting a radio signal to which the tag responds by transmitting the stored information, along with its unique Electronic Product Code (EPC) back to the reader [40]. This process causes streams of RFID data records to be dumped into an RFID database, which then grows to a gigantic size. Records take the format of $\langle EPC, loc, t \rangle$, where EPC is a unique identifier of the tagged object, loc is the location where the reader is positioned, and t is the time when the reader detected the tag. A data recipient/analysis module can submit queries to the query engine to request information on

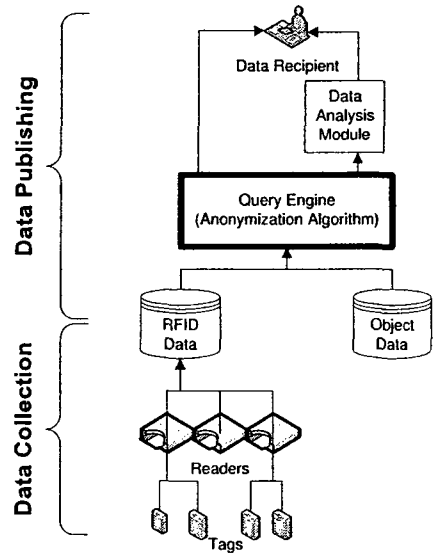


Figure 1: Data Flow in an RFID System

particular tagged objects or general workflow patterns [19]. Afterwards, the query engine processes the requests and responds back by binding the RFID data to some related object-specific data stored in a separate database.

RFID technology is useful; however, the uniquely identifiable objects raise privacy concerns for the individuals that carry these objects. For example, an adversary might be able to learn about a person's movements, and thereby gain an advantage. Several techniques [54] [6] [23] [50] including privacy-preserving RFID technologies [40], such as EPC re-encryption and killing tags [24], have been proposed to address the privacy issues in the *data collection* phase where communication between tags and readers takes place. We explore the privacy threats that occur in the data publishing phase, in which a large volume of RFID data is released to a third party, and propose a practical solution, a privacy model, that suits the special nature of RFID data. We present an anonymization approach [7] [8] which entails hiding some data before its release. We

develop anonymization algorithm for the query engine (see Figure 1) to convert the underlying object-specific RFID data from its original raw state to a transformed version that is invulnerable to attacks against individuals' privacy. We assume that a data recipient is a potential adversary (or attacker), who seeks to identify a target victim's record and/or learn about the target victim's sensitive information associated with her record in the published data. This applies to any data recipient whom we often describe as a third party. In RFID data, some particular attributes are considered to be sensitive. This is because these attributes contain sensitive values about their corresponding records' owners. The sensitive value is not be associated with its related individual upon publishing of the data. When we speak of RFID data publishing (or releasing), we refer to situations where the RFID data is shared with specific recipients, e.g., data analysis institutions, or released for public access. The term "data holder(s)" refers to the individual/organization that possesses the data and wishes to publish it. In this thesis we assume that in any given RFID data, each record belongs to only one individual, whom we refer to as the record owner.

1.1 Motivation

We begin by providing a real-life example of sharing person-specific RFID data. This shall demonstrate the potential risk of compromising individuals' privacy when publishing data.

The Oyster Travelcard in Transport for London (TfL), is a successful application of RFID technology in a transit system. Passengers register their personal information when they first purchase their RFID-tagged smart cards. Then, the appropriate fare amount is deducted from their cards every time they use the transport services. Passengers refill their smart card anytime as needed.

Table 1: Raw passenger-specific path table T

EPC	Path	Employment Status
1	$\langle a1 \rightarrow d2 \rightarrow b3 \rightarrow e4 \rightarrow f6 \rightarrow c7 \rangle$	On-welfare
2	$\langle b3 \rightarrow e4 \rightarrow f6 \rightarrow e8 \rangle$	Full-time
3	$\langle b3 \rightarrow c7 \rightarrow e8 \rangle$	Full-time
4	$\langle d2 \rightarrow f6 \rightarrow c7 \rightarrow e8 \rangle$	Retired
5	$\langle d2 \rightarrow c5 \rightarrow f6 \rightarrow c7 \rangle$	On-welfare
6	$\langle c5 \rightarrow f6 \rightarrow e9 \rangle$	Retired
7	$\langle d2 \rightarrow c5 \rightarrow c7 \rightarrow e9 \rangle$	Part-time
8	$\langle f6 \rightarrow c7 \rightarrow e9 \rangle$	Part-time

Table 2: Anonymous table T' for L=2, K=2, C=50%

EPC	Path	Employment Status
1	$\langle b3 \rightarrow f6 \rightarrow c7 \rangle$	On-welfare
2	$\langle b3 \rightarrow f6 \rightarrow e8 \rangle$	Full-time
3	$\langle b3 \rightarrow c7 \rightarrow e8 \rangle$	Full-time
4	$\langle f6 \rightarrow c7 \rightarrow e8 \rangle$	Retired
5	$\langle c5 \rightarrow f6 \rightarrow c7 \rangle$	On-welfare
6	$\langle c5 \rightarrow f6 \rightarrow e9 \rangle$	Retired
7	$\langle c5 \rightarrow c7 \rightarrow e9 \rangle$	Part-time
8	$\langle f6 \rightarrow c7 \rightarrow e9 \rangle$	Part-time

The public transit companies utilize the personal journey data (the RFID data) to improve their services. Analyzing RFID data is a non-trivial task; transit companies often do not have the expertise to perform the analysis themselves but outsource this process and therefore, require granting a third party access to the RFID data and passenger data (object data in Figure 1). The passenger data may contain person-specific (sensitive) information, such as age, disability status, and (un)employment status. TfL does say that it does not associate journey data with named passengers, although they provide such data to government agencies on request [46]. Our goal, in this case, is to answer the question: How can an RFID data holder (e.g., the transit company) safeguard data privacy while keeping the released RFID data useful for analysis? We exemplify this concept with the following scenario.

Example 1.1.1 A transit company wants to share Table 1, the passenger-specific path table, with a third party for the purpose of performing data analysis. The EPCs, passengers' names, and other explicit identifiers are removed from the table to be released. EPCs are included in Table 1 for the sake of clarity. Each record contains a *path* and some passenger-specific information, where *path* consists of a sequence of *pairs* $(loc_i t_i)$ indicating the passenger's visited location loc_i at timestamp t_i. For instance, the path associated with $EPC\#3$ is $\langle b3 \rightarrow c7 \rightarrow e8 \rangle$, meaning that the passenger has been to locations b, c, and e at timestamps 3, 7, and 8, respectively. We assume that the passenger-specific path table, Table 1, contains only one sensitive attribute, namely Employment Status. In other words, one domain value from the Employment Status attribute is associated with each record in the table. Let *On-welfare* be the only sensitive value. The goal now is to prepare a version of Table 1 that is immunized against privacy attacks and still useful for data analysis. We will demonstrate how Table 1 is susceptible to privacy attacks and, therefore, compromises the individuals' privacy. ■

In Table 1 of Example 1.1.1, some values in the *Employment Status* attribute are considered sensitive; an adversary is not supposed to associate these sensitive values with their related passengers. However, some values are usually considered more sensitive than others, e.g., a passenger might not object to people knowing that her employment status is *Full-time*, in contrast to the status of *On-welfare*. Thus, the data holder specifies a set of sensitive values which is a subset of the domain values of a sensitive attribute in the passenger-specific information. Privacy attacks manifest when an adversarial data recipient attempts to identify the target victim's record and/or sensitive value from the released data. We illuminate the following types of privacy attacks and project their impact on Table 1:

1. *Record linkage*: record linkage is a privacy attack in which the adversary exploits the uniqueness of a passenger's path in the released data table. If a passenger's path is so specific that it only matches to a small number of other passengers, linking the victim's record from the released RFID data table along with her employment status may become possible. The assumption is that the adversary possesses some knowledge about the locations and timestamps (*pairs*) existing in a victim's path. For example, assume the adversary knows that Alice has been to locations e and c at timestamps 4 and 7, respectively. Since Alice's record is the *only* record with a path containing $e4$ and $c7$, Alice's record and her sensitive value *On-welfare* can be uniquely identified.

2. *Attribute linkage*: attribute linkage is another privacy attack that occurs when a group of records that share some combination of pairs contains a frequently appearing sensitive value. Even though a target victim's record might not be identified, inferring the victim's sensitive value from such a group becomes possible. For example, suppose the adversary is aware of the existence of pairs $d2$ and $f6$ in a target victim's path. A group of three records ($EPC\#1$, 4, 5), which contain $d2$ and $f6$, has two records with the sensitive value *On-welfare*. Two out of three records having the same sensitive value allow the adversary to infer that the target victim is on welfare with 67% confidence.

Several privacy models have been proposed to combat record and attribute linkage attacks. These privacy models, however, target a different type of data, i.e., relational data. K-anonymity [5]

[12] [14] [15] [16] [30] [25] [31] [38] [47] [52] [33] [34], ℓ-diversity [27], confidence bounding [48], and t-closeness [26] are examples of such privacy models. These models assume a predetermined set of particular attributes in the released data table, called *quasi-identifier* (QID) [8] [32], that an adversary can use to identify a target individual. We define QID attributes and background knowledge below.

Quasi-identifier (QID) A QID in a table T is a set of attributes which can combine to uniquely identify single or multiple records' owners in T with the help of background knowledge.

Adversary's background knowledge Background knowledge is the information an adversary externally obtains about a particular (or multiple) record owner in T, and can be jointly used with the set of QID in T to identify a target individual.

Despite the fact that the abovementioned privacy models have been proven effective for anonymizing relational data, they become inapplicable for anonymizing RFID data due to the curse of high dimensionality [2].

High-dimensionality is an intrinsic characteristic of RFID data due to the huge possible combinations of locations and timestamps. Consider a subway system having 50 stations that operate 20 hours a day. The total number of dimensions of the RFID data table would be $50 \times 20 = 1000$ dimensions. Each dimension (pair) could be a potential piece of knowledge used by an adversary to perform record or attribute linkages; therefore every dimension is considered a potential quasi-identifying (QID) attribute. If we apply a traditional privacy model, such as K-anonymity, all dimensions would then be included in a single QID and every path would have to be indistinguishable from at least $K - 1$ other paths. In order to achieve K-anonymity, the high dimensional [2] nature of RFID data would likely cause most of the data to be suppressed. Consequently, the utility

of the resultant anonymous data becomes insufficient for further data analysis. As an example, applying K-anonymity to Table 1 for a small value of $K = 2$, i.e., 2-anonymity, results in suppressing all pairs of $a1, d2, b3, e4, c7, e9$ from the table.

As mentioned earlier, in the context of RFID data, all attributes (dimensions) could potentially be used by an adversary as background knowledge to launch privacy attacks, i.e., record or attribute linkages. Thus, in traditional K-anonymity and its extended privacy models, the assumption would be that all attributes in an RFID data table are included in a single QID. However, assuming an adversary's knowledge about the *entire* locations and timestamps in a target victim's path is unrealistic as it requires non-trivial effort to collect such knowledge from a large number of locations at different times. Hence, a real-life scenario would suggest limiting the adversary's background knowledge to a maximum of L pairs of locations and timestamps about a target victim. This is an essential improvement in our privacy model.

We define a novel anonymization privacy model, LKC-*privacy*, which adapts to the challenge of high dimensionality in RFID data. LKC-*privacy* provides a practical solution to accounting for an adversary's background knowledge. This is achieved by bounding the amount of information an adversary possesses to a maximum threshold of L pairs. The intuition is that, given an RFID data table T, we need to make sure that for any path in T, every proper subsequence q with maximum length (number of pairs) L appears in at least K records in T. Moreover, the inference confidence of any sensitive value $s \in S$ from a group of records that contain q is not greater than C, where S is a set of sensitive values chosen by the data holder from the domains of the sensitive attributes in T. For the three threshold parameters in LKC-*privacy*, L and K are positive integers and $0 \leq C \leq 1$ is a real number. When applying LKC-privacy, an adversary can successfully perform record

linkage attacks with a maximum probability of $1/K$ and successfully perform attribute linkage attacks with a maximum probability of C. These probabilities are based on the assumption that the adversary's background knowledge does not exceed L pairs of locations and timestamps.

Continuing from Example 1.1.1, applying our privacy model to Table 1 to satisfy $(2, 2, 50\%)$-privacy will transform it into an anonymous version T', Table 2, that adheres to the privacy requirement. As shown in Table 2, pairs $a1$, $d2$, and $e4$ from Table 1 have been suppressed in order to achieve $(2, 2, 50\%)$-privacy. In Table 2, there are at least 2 records that share any possible subsequence q (from any path) with a maximum length of 2. In addition, inferring the sensitive value *On-welfare* from the records that contain q is achieved with a maximum confidence of 50%. Anonymization using $(2, 2, 50\%)$-privacy preserves a considerable amount of data when compared to traditional 2-anonymity, which requires further suppressing pairs $b3$, $c7$, and $e9$ from Table 2.

1.2 Outline of the thesis

This thesis is organized as follows. In Chapter 2, we go through some of the prominent privacy models in the context of three common types of data: relational, transaction, and trajectory data. We first discuss an essential privacy model for thwarting record linkages, namely the K-anonymity paradigm. Then, we discuss methods for thwarting attribute linkages, such as confidence bounding and the ℓ-diversity principal. We provide illustrative examples for each approach. Furthermore, we perform comparisons between each type of data and RFID data, and state the reason(s) why anonymization methods for different types of data become ineffective for anonymizing RFID data. Chapter 3 formally defines the problem statement. A formal definition is given for the privacy

threats along with a new privacy model, called LKC-privacy, for anonymizing high-dimensional RFID data. Chapter 4 presents an efficient anonymization algorithm for achieving LKC-privacy. We also present the Critical Violation Tree structure, which boosts the efficiency in our anonymization algorithm. Chapter 5 evaluates the performance of our proposed model in terms of data quality, efficiency, and scalability by employing two different data sets. Data quality is a measure we use to describe the distortion inflicted on the original data set due to anonymization. We present a comparison of data quality between our new LKC-privacy and traditional K-anonymity. Finally, in Chapter 6 we conclude the work presented in this thesis and point out some possible future research directions.

Chapter 2

Literature Review

In this chapter, we study various approaches that have been proposed to address the problem of privacy preservation in different types of data. Publishing data is important and has a versatile usage, e.g., academic research, statistical studies, information mining, etc. However, due to the potential damage that could arrive if privacy is penetrated, e.g., record and attribute linkages, preserving data privacy has become an important topic in the society. We discuss three commonly used types of data; relational, transaction, and trajectory data, each of which, when published, has been shown to be susceptible to revealing information that the recipient is not supposed to acquire. We describe each type along with its respective approaches, explain how these approaches become inapplicable when RFID data is in hand, and end this chapter with a summary of these models and methods.

The goal of privacy-preserving data publishing is to release or share with a third party a person-specific data set (a table in which each record refers to a unique individual) while maintaining the record owners' privacy. Even when removing identifying pieces of information, e.g., SSNs, from

Table 3: Alumni: external table

Name	DoB	Sex	Major
Christina	Jan 1970	Female	Physics
Daisy	Mar 1970	Female	Psychology
Julia	Aug 1970	Female	Biology
Jordan	May 1965	Male	Finance
Douglas	Jan 1965	Male	Marketing
Alex	Apr 1960	Male	Marketing
Robert	Dec 1960	Male	Finance
Peter	Dec 1960	Male	Electrical Eng.
Eric	Apr 1960	Male	Industrial Eng.

Table 4: Welfare: private table

DoB	Sex	Major	Class
Apr 1960	Male	Marketing	A
Aug 1960	Male	Finance	B
Apr 1960	Male	Finance	C
Feb 1970	Female	Industrial Eng.	C
Nov 1970	Female	Electrical Eng.	B
Mar 1970	Female	Psychology	A
Dec 1970	Female	Psychology	A

the published data set, some attributes called *quasi-identifier (QID)* [8] [32] can uniquely combine, hence compromising some record owners' privacy. When successfully identified by using a QID, the victim's record in the released data set can be *linked* to externally available information pertaining to the same person, thereby revealing information the recipient is not supposed to know. This scenario, called *record linkage attack*, is illustrated in Example 2.0.1.

Example 2.0.1 Suppose that a welfare agency in a town is sharing Table 4 with an institute for statistical analysis. Table 4 is a list of all the people receiving financial assistance. The amount of financial assistance an individual recieves is described in a category of 3 classes: class A, class B, and class C. People who belong to class A are the most needy thus they receive a larger amount of financial assistance. Class C includes those who need the least assistance. The class that each individual belongs to is not to be disclosed; hence, attribute Class is considered to be sensitive

(attribute Class is not removed because it is required for the analysis task). Therefore, the agency removes identifying pieces of information, e.g., SSNs and phone numbers, from the table to be released. Meanwhile in the same town, a university, UniversityX, is posting an up-to-date table (Table 3) on its website listing information about the university's alumni, including DoB, Sex, and Major. Each record represents the information of a single alumnus. The combinatorial attributes of DoB, Sex, and Major (forming the set of QID attributes) in Table 4 can be jointly used with Table 3 to potentially reveal individuals' identities. For example, Table 4 shows that the first record refers to a man receiving financial assistance of class A. In this case, $qid = \langle Apr1960, Male, Marketing \rangle$. With a background knowledge indicating that some people who graduated from UniversityX are currently on welfare, an attacker can use the $qid = \langle Apr1960, Male, Marketing \rangle$ to narrow down the Alumni list to uniquely point out Alex. ■

Based on the QID, Example 2.0.1 shows how information can be linked between released data sets to reveal the identities of record owners even though identifying information was removed. Example 2.0.1 exemplifies a record linkage attack. In Subsections 2.1.1 and 2.1.2, we discuss some of the proposed techniques that attempt to combat record linkage (namely, K-anonymity) and attribute linkage attacks, respectively, in relational data.

2.1 Relational Data

2.1.1 Record Linkage

A decent amount of work has been done toward anonymizing relational data. One notable proposition, which emerged in the early stages of research in privacy preservation, is K-anonymity [37]

[39] [43] [25]. Initially proposed by Samarati and Sweeny [39], K-anonymity stipulates that in a relation R, each record must be indistinguishable from at least $K-1$ other records. Relation R is the privately held table depicting the data set to be "safely" released, and a record is a row assumed to be the only row representing a unique individual. That is, in any record within a table, all attributes' values have to match to a minimum of $K-1$ other values (following the same order) in different records. A data set that adheres to K-anonymity (satisfies the K-anonymity requirement) is said to be K-anonymous. A K-anonymous data set guarantees that linking any of its records to externally available information can never exceed the probability of $1/k$, since there are $K-1$ duplicates of each record. This is equivalent to saying that the probability of a successful record linkage attack is at most $1/k$.

In order to achieve K-anonymity, Samarati and Sweeny employed *generalization* and *suppression* [39]. Both techniques can be used jointly or independently. In generalization, the QID attributes' values are replaced with more general ones based on a data holder's pre-defined taxonomy tree. The purpose of generalization is to eliminate the potential uniqueness resulting from a combination of QID attributes' values in a record. In a generalized table, more records (a minimum of K) will be grouped under the same qid. Although the data becomes less precise, consistency is maintained. We will use the taxonomy trees presented in Figure 2 to generalize Table 4 on $QID = \{DoB, Sex, Major\}$ so that it adheres to K-anonymity for $K = 2$. This process can also be described as anonymizing Table 4 so that it satisfies 2-anonymity, or simply 2-anonymizing Table 4.

As seen in Example 2.0.1, $qid = \langle Apr1960, Male, Marketing\rangle$ in Table 4 uniquely identifies the individual it relates to when linked to external information. Table 5 shows that this uniqueness

Table 5: 2-anonymous Welfare table

DoB	Sex	Major	Class
1960	Male	Business	A
1960	Male	Business	B
1960	Male	Business	C
1970	Female	Engineering	C
1970	Female	Engineering	B
1970	Female	Arts & Science	A
1970	Female	Arts & Science	A

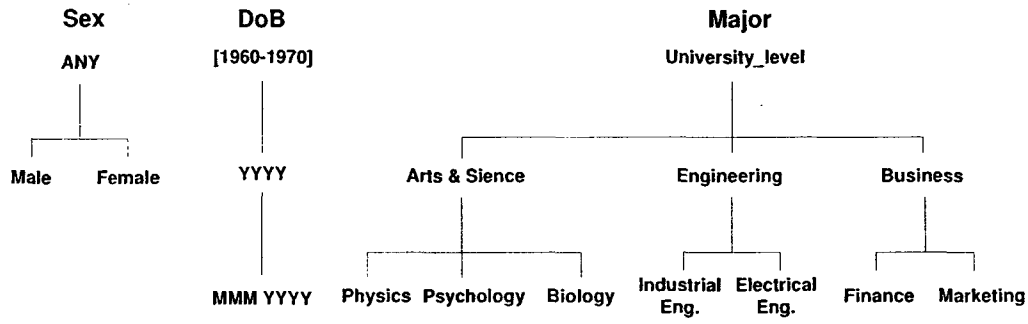


Figure 2: Taxonomy trees

is eliminated as the anonymous table consists of three distinct groups: $qid = \langle 1960, Male, Business \rangle$, $qid = \langle 1970, Female, Engineering \rangle$, and $qid = \langle 1970, Female, Arts\ \&\ Science \rangle$. Each qid group contains at least two records, making any record indistinguishable from at least one other record in the same group.

Suppression is complementary to generalization [39]. Rather than replacing data, suppression entails removing some data from the table to be released. Samarati and Sweeny suggest enforcing suppression on particular records. That is, when marked as an outlier [39] [5], the entire record in the table is removed. Outlier records are those which still fall short of fulfilling the enforced K-anonymity requirement even after the table has been generalized.

For example, suppose that Table 4 is to be 4-anonymized to table T'. Generalization is enforced, resulting in Table 5. However, Table 5 is not 4-anonymous; therefore a further sequence

of generalization steps is required. A naïve approach is to generalize all the QID attributes' values to the most abstract form (Figure 2). This approach will eventually produce a 4-anonymous table T' wherein all of its records have $qid = \langle [1960 - 1970], ANY, University\_level \rangle$. In fact, T' is 4-anonymous because any of its records matches at least $4 - 1$ other records on QID = {DoB, Sex, Major}. Table T' is indeed then more privacy-preserving than its original version, but it is much less informative as well. Each generalization step results in more data abstraction. To avoid unneeded generalization on Table 5, record suppression suggests removing the first three records. By doing so, we can perform just one more generalization step on attribute Major. The resulting table T' is 4-anonymous with all records having $qid = \langle 1970, Female, University\_level \rangle$.

Generalization and suppression are best enforced together. That is, a maximum suppression threshold can be defined by the data holder, and within that threshold suppression is more preferable than generalization.

Although suppression minimizes the data abstraction caused by generalization, it affects data integrity by removing specific records from the table to be released. Generalization, on the other hand, preserves consistency in all attributes, but affects all of the records (all values in an attribute domain are replaced by others according to a taxonomy tree). Generalization increases data privacy, but lowers data utility.

Recalling the purpose of anonymization; we want to "safely" release a privately held data set and, at the same time, ensure the data utility. In other words, it is desirable to keep the amount of information held by the anonymous data as large as possible. *Optimal anonymization* [25] describes the condition at which an anonymous table T' is most informative in comparison to other

anonymous versions of the original table T, with T' being the least anonymous (one generalization/suppression step from violating a given privacy requirement). Moreover, *minimal anonymization* [21] [42] [22] [49] [13] [15] [16] is achieved when T' is least anonymous with respect to a given privacy requirement without necessarily being most informative. Despite the fact that achieving optimal K-anonymization has been proved to be NP-hard [29], a few approaches toward optimality have been proposed [25] [37] [5]. LeFevre et al. [25] proved that achieving optimality is feasible by proposing an algorithm (and a set of its variations) called *Incognito* that guarantees optimality using *full-domain generalization.* Full-domain generalization [37] [43] [25] is a method for achieving K-anonymity with respect to the QID attributes. With this method, if one value from a QID attribute is generalized to some level in the taxonomy tree, then all the other values in that attribute domain are generalized to the same level. For example in Figure 2, if Marketing is generalized to Business, all the leaves of that tree are generalized to their pertinent parent nodes: Business, Engineering, and Arts & Science. Although Incognito maintains optimal K-anonymity, its complexity grows exponentially with the number of QID attributes.

The K-anonymity privacy model can be used to thwart record linkage attacks by making each record indistinguishable from at least $K - 1$ other records in an anonymous table. Nevertheless, it fails to prevent attribute linkage attacks when a sensitive attribute is present in an anonymous table. Next, we come across some works that counter attribute linkage attacks.

2.1.2 Attribute Linkage

For some QID attributes in a K-anonymous table, any record is indistinguishable from at least $K - 1$ other records within a qid group, thereby, preventing an attacker from uniquely identifying

a target victim with the help of external data. However, if all or most of the records of a qid group share the same sensitive value, deducing the target victim's sensitive information will be highly probable. This type of attack is the manifestation of attribute linkage. An illustration follows.

Example 2.1.1 Suppose that Daisy's friend, Bob, is looking at anonymous Table 5. Bob knows that his friend Daisy (2nd record in Table 3) has studied Psychology at UniversityX and was born in March 1970. Bob observes that Daisy's record in the Alumni list, based on QID = {DoB, Sex, Major}, links to 2 records from the 2-anonymous Welfare table. However, both of the latter records indicate that their related individuals belong to class A. In this particular case, Bob is now certain that his friend Daisy receives class A financial assistance - the very piece of information Bob is not supposed to acquire about Daisy. ■

To thwart attribute linkage attacks, Wang et al. [48] proposed a new privacy model, *confidence bounding*, which can work complementarily with K-anonymity. In a K-anonymous table T', confidence bounding prevents the inference of a sensitive value in any qid group from exceeding a certain probability threshold. In other words, the confidence of inferring a sensitive value s is upper bounded by h, a pre-defined confidence percentage determined at the data holder's discretion.

To apply confidence bounding, T' has to satisfy a set of privacy templates of the form $\langle QID \rightarrow s, h \rangle$, where QID is a quasi-identifier, s is a sensitive value, and h is a confidence threshold. For simplicity, let us assume that T has to satisfy a single privacy template. $qid \rightarrow s$ indicates the percentage of inferring s from qid. That is, $qid \rightarrow s$ represents the number of records containing the sensitive value s divided by the total number of records grouped by the same qid. This is denoted by $conf(qid \rightarrow s)$. The maximum $conf(qid \rightarrow s)$ in any qid group is denoted by

$Conf(QID \rightarrow s)$. If $Conf(QID \rightarrow s) \leq h$, then table T satisfies the privacy template $\langle QID \rightarrow s, h \rangle$.

For example, for $QID = \{DoB, Sex, Major\}$, enforcing the privacy template $\langle QID \rightarrow A, 50\% \rangle$ on Table 5 requires that at most 50% of the records in any qid group have sensitive value A. Table 5 has 3 qid groups: $\langle 1960, Male, Business \rangle$, $\langle 1970, Female, Engineering \rangle$, and $\langle 1970, Female, Arts \ \& \ Science \rangle$. For each group, the percentages of records with sensitive value A are: 33%, 0%, and 100%, respectively. Hence, the aforementioned privacy template is violated because the last group violates the confidence threshold.

Flexibility is an inherent characteristic in privacy templates. The data holder can always lay out different templates for different sensitive values depending on how sensitive some values are than others. For example, in Table 5, one template for inferring A could be $\langle QID \rightarrow A, 30\% \rangle$ while another template for inferring C could be $\langle QID \rightarrow C, 80\% \rangle$.

Another approach that has been proposed to thwart attribute linkage attacks is the notion of the ℓ-diversity principal [27] [35] [9]. Machanavajjhala et al. [27] suggest that for a table to be ℓ-diverse, every qid group is required to have at least ℓ "well-represented" records with respect to their relative sensitive values. The reason for describing ℓ-diversity as a principal is the possibility of interpreting the privacy requirement "well-represented" via several instantiations. One instance of the ℓ-diversity principal is to ensure that in every qid group there are at least $\ell \geq 2$ different sensitive values. In this case, records will be "well-represented" since their associated sensitive values are ℓ-diverse. In other words, at least ℓ distinct values of the sensitive attribute are in each qid group. This notion is captured in Example 2.1.2.

Table 6: 3-diverse Welfare table

DoB	Sex	Major	Class
1960	Male	University_level	A
1960	Male	University_level	B
1960	Male	University_level	C
1970	Female	University_level	C
1970	Female	University_level	B
1970	Female	University_level	A
1970	Female	University_level	A

Example 2.1.2 In Example 2.1.1, we illustrated how an attacker can reveal a target victim's sensitive information. With the group of $qid = \langle 1970, Female, Arts\ \&\ Science \rangle$ from Table 5 indicating that all its related individuals belong to class A, Bob correctly concluded that his friend Daisy receives class A financial assistance. In order to fix this "flaw", we can enforce 3-diversity over Table 5 to ensure that at least 3 distinct values of the sensitive attribute are present in any qid group. This is demonstrated in Table 6. ■

The reason for enforcing diversity over some sensitive values in qid groups is to distribute the frequencies of occurrences of these sensitive values for the sake of eliminating inference with high confidence. The best-case scenario is to have all sensitive values approximately evenly distributed within every qid group. Typically, this is not what we find in practice, mainly because some sensitive values are inherently more (or less) frequent than others. In patients' data, for example, where disease is considered a sensitive attribute, it is natural to have $Allergy$ more frequently than $Cancer$; generally speaking.

To measure how evenly distributed sensitive values are in every qid group in a table, Machanavajjhala et al. [27] propose another instantiation of the ℓ-diversity principal: Entropy ℓ-diversity. For a table to satisfy entropy ℓ-diversity, $log(\ell)$ should not exceed the sum of entropies of every sensitive

value s in each qid group. More formally,

$$-\sum_{s \in S} P(qid, s) log(P(qid, s)) \geq log(\ell) \quad (1)$$

where S is the domain of a sensitive attribute and $P(qid, s)$ is the fraction of records containing the sensitive value s in the qid group. Since entropy measures the uncertainty of inferring sensitive values, higher entropy implies a more even distribution of sensitive values in qid groups, and therefore less inference confidences.

For example, in Table 6, the entropy of the first group of $qid = \langle 1960, Male,$ $University\_level \rangle$, is $3 \times (-\frac{1}{3} log(\frac{1}{3})) = log(3)$, and the entropy of the second group of $qid = \langle 1970, Female, University\_level \rangle$, is $-\frac{1}{4} log(\frac{1}{4}) - \frac{1}{4} log(\frac{1}{4}) - \frac{2}{4} log(\frac{2}{4}) = log(2.82)$. Therefore, Table 6 is entropy ℓ-diverse for $\ell \leq 2.82$.

Unlike confidence bounding which states a probabilistic inference threshold for some sensitive values, entropy ℓ-diversity does not provide the data holder with a clear intuitive translation of the diversity measure. For example, as mentioned above, Table 6 satisfies entropy ℓ-diversity if $\ell \leq 2.82$. However, the data holder will not be able to conclude that 2 out of 4 records in a qid group are associated with class A sensitive value, i.e., a successful attribute linkage attack on class A would be 50%.

If the goal is to ensure low inference confidences of sensitive values through high diversity, clearly ℓ is required to be higher as more records add up to a qid group. However, this also implies a larger qid group size by which further generalization (if it was used) would be needed.

2.1.3 K-anonymity and RFID Data

As mentioned earlier, the K-anonymity privacy model addresses the privacy issue related to revealing identities through record linkage attacks in relational data sets. However, K-anonymity becomes inapplicable when anonymizing RFID data. When we describe the K-anonymity model as "inapplicable", we are referring to its ineffectiveness in achieving useful anonymous data. The following two reasons explain why. First, applying K-anonymity to high dimensional data will result in poor data quality. RFID data tend to be sparse and high dimensional [2], and the set of quasi-identifing attributes can contain hundreds of attributes, as will be seen later in Chapter 5. Each *item* in an RFID data set record is a potential *quasi-identifier*. Therefore, enforcing the K-anonymity privacy requirement would result in anonymizing most of the records, rendering the data utility weak indeed. In Chapter 5 we shall see how relatively huge the amount of distortion is when enforcing K-anonymity. In our proposed model, we overcome this obstacle by assuming that the adversary knows up to L *pairs* of locations and timestamps that a target victim has previously visited. The second reason is that the K-anonymity model does not consider sensitive attributes' disclosure. This adds an extra burden since now the data holder needs to choose an additional appropriate privacy model to mitigate attribute linkage attacks that target sensitive attributes. Example 2.1.1 demonstrates the second reason. Accounting for attribute linkage attacks is an essential privacy requirement in our model, enforced by limiting the inference confidence of a sensitive value to a pre-determined threshold.

Up to this point, we have discussed K-anonymity and some other methods to defend against record and attribute linkages, respectively, in relational data. In the next section, we show some anonymization approaches for releasing high-dimensional transaction data.

2.2 Transaction Data

In transaction data sets, data is organized as a set of *transactions*. Each record encompasses an arbitrary number of *items* forming a *transaction* relating to a distinct individual. A transaction is a subset of a much larger population U of items. Click streams, visited web sites logs, online/offline shopping, and query logs are examples of transaction data where each item indicates an event: clicking a picture, browsing a web site, buying an item (not to be confused with the transactions constituent elements *items*), and performing a query, respectively. A transaction can describe one's behavior, making the data set an excellent source for mining [20]. However, such rich data raises privacy concerns about its transactions' owners, as illustrated in Examples 2.2.1 and 2.2.2.

Unlike relational data which in practice tend to have a small number of QID attributes (5 or 6 [29]), transaction data is considered to be high dimensional. Each item in a transaction data set is a dimension by itself because it is a potential piece of knowledge/information previously acquired by an adversary and it can be used with a combination of other items to reveal a target victim's identity. Regardless of a transaction's variable length (number of items), all items from the large population U are considered quasi-identifiers. The following example illustrates an attack scenario on high-dimensional transaction data.

Example 2.2.1 Suppose that an online video sharing web site, VidShare, has released an "anonymous" data set about its users' logs, for research purposes. Each record describes recently watched videos by a distinct user. Brian and Linda are classmates in business school who always watch their favorite movie reviews together on VidShare. In the published data set, Brian noticed a record containing the same movie reviews that he and Linda had watched lately. Moreover, the same record

contains videos relating to the same subjects they study in their class. Brian is now certain that the record belongs to his Linda's account as it is unique in the entire data set. Brian also noticed that the (identified) record contains further videos pertaining to uterine cancer. Considering the sensitivity of the matter, Linda may not want to disclose such information about herself. In spite of removing identifying pieces if information from the released data set, e.g., name/nickname, Linda's record has been identified and has leaked sensitive private information. ■

We further provide a real-life example of the AOL [1] incident that reflects actions data holders might take if users' privacy is compromised.

Example 2.2.2 In a data of search queries released by AOL for academic research purposes, amongst 650,000 AOL users, record No. 4417749 was successfully traced back to its legitimate owner revealing her identity. Ms. Thelma Arnold, a 62-year-old widow, is an AOL subscriber who has conducted many search queries which mirrored her daily activities leading to uniquely identifying her persona. This incident resulted in AOL's retrieval of the data and, more importantly, rendered data holders unenthusiastic about publicly releasing their data even when specific identifying pieces of information are removed from the published data [4]. ■

Examples 2.2.1 and 2.2.2 capture two major characteristics about transaction data: (1) transaction data is demanded for research purposes especially in the field of data mining and (2) transaction data poses privacy concerns, since removing personal information proved a useless precaution, hence the need for further "processing" before data release. Next, we briefly review some of the proposed anonymization approaches in the context of transaction data.

[1] www.aol.com

Some recent studies attempt to tackle the challenges presented when anonymizing high dimensional sparse transaction data [45] [17] [52] [53]. Terrovitis et al. [45] extended the K-anonymity paradigm to K^m-anonymity. In relational data, K-anonymity assumes that an attacker may be able to collect background knowledge with respect to the *entire* QID attributes, but in transaction data this assumption becomes unrealistic for the following reason. As we saw earlier, any item in the population U, which may contain tens of thousands of items, is a potential quasi-identifier. Thus either the attacker has to exert enough effort to collect information about all the items appearing in the target victim's transaction (which is not commonly the case), or the attacker's background knowledge can be limited to some extent. K^m-anonymity assumes that the attacker knows about a target victim's transaction at most m items. The goal then becomes making any transaction with any combination of up to m items indistinguishable from at least $K - 1$ other transactions in the data set. To achieve this goal, Terrovitis et al. [45] used generalization, by which items are generalized to more abstract values. They present a suite of three algorithms to anonymize transaction data: *optimal anonymization* which is inapplicable for real-life data sets due to its expensive computational cost; and two heuristics: *direct anonymization* and *apriori anonymization*.

In [45] the authors do not account for the existence of a separate column for sensitive values in the published data, e.g., medical condition, as they consider any item to be potentially sensitive. The works presented in [17], [52], and [53] however, consider the items in U as public quasi-identifiers used by an attacker to identify a target victim's transaction and/or to infer associated sensitive information. To combat such attacks, i.e., record and attribute linkages, [53] proposes a privacy notion to protect the published data under some privacy requirements. *(h,k,p)-coherence* is a privacy model that restricts the number of items an attacker could have previously gained

about a target victim to p; following the intuition that it is realistic to assume an attacker's partial knowledge about a particular transaction in the data set. In addition, the privacy model requires the existence (or none at all) of at least $k-1$ other transactions in the data set that share any combination of up to p items. Within these k transactions, the liklihood of inferring a particular sensitive item must not exceed h percent. If a data set adheres to the *(h,k,p)* privacy requirements, the data set is said to be *(h,k,p)-coherent.* Consequently, an *(h,k,p)-coherent* data set, with attacker's knowledge being up to p items, limits the probability of a successful record linkage to $1/k$ and a successful attribute linkage to h [53].

Xu et al.([52] and [53]) employ the same privacy notion *(h,k,p)-coherence.* However, the former work targets preserving *frequent itemsets* by eliminating *moles* and withholding as much *nuggets* as possible. Frequent itemsets are sets of items that frequently occur in the data set and are essential for data mining purposes. Nuggets are up to p-length itemsets (public or sensitive) that occur at least k times in the data set. Moles, on the other hand, are public itemsets that violate the *(h,k,p)* privacy requirements. Instead of enumerating all possible moles and nuggets, Xu et al. [52] overcome this bottleneck by introducing a novel *border-based* representation of moles and nuggets which significantly improves scalability. Moles/nuggets are bounded by maximal and minimal itemsets from which they are derived.

Unlike Terrovitis et al. in [45] who use generalization to achieve a privacy requirement, both [53] and [52] use global item suppression. This technique has a prominent advantage over generalization; i.e., suppression is performed on a single (public) item by deleting it from a transaction. Moreover, suppressing an item is performed globally, that is, when suppressed, all occurrences of that item in the data set will be suppressed as well. Therefore, consistency is preserved for

all the items appearing in the published data, which shall maintain truthfulness in data analysis or data mining applications. Generalization, however, is applied to all the children of a subtree node, which causes more information loss as the taxonomy tree would be characterized as flat (wide and short) and fan-out (many children for a single node). One last point is that preparing a taxonomy tree for generalization is not a lightweight task, especially in high-dimensional data. For the above mentioned reasons, adopting item suppression is often more practical than generalization.

In [17], Ghinita et al. introduce a permutation-based approach that takes advantage of the sparseness nature of transaction data. Rows (transactions) and columns (every item in a population U) are re-arranged in such a way that items in each transaction appear diagonally. This organization helps to group transactions with similar items, with these items being considered the group's QID. Each group of transactions is then anonymized based on its relevant QID. Sensitive items are treated separately, and any transaction can have none, one, or multiple sensitive items. After anonymization, any transaction within a certain group is associated to any sensitive item in that group with a confidence of up to a maximum (pre-specified) threshold.

2.2.1 Transaction Data vs. RFID Data

In this subsection, we denote the distinction between transaction data and RFID data, and show why the proposed anonymization approaches for the former data type are inapplicable when anonymizing RFID data. Despite the fact that both data types share the characteristic of being high-dimensional and sparse, one essential difference is how items are represented in a record (transaction or RFID record). Basically, a transaction is a *set* of items, e.g., merchandise bought by an individual, and an RFID record consists of a *sequence* of "items". In the next chapter we will see

that an "item" in an RFID record is a *pair* composed of a location and a timestamp. A sequence of *pairs* in an RFID record is called a *path.* Therefore, transactions {a, b} and {b, a} are equivalent, in contrast to *paths* $\langle a, b \rangle$ and $\langle b, a \rangle$.

In addition to the difference between the nature of transactions and paths, a method for anonymizing transaction data may not account for attribute linkages [45], i.e., there is no consideration for associating a particular individual in the data set to some sensitive information (e.g., medical condition). As for anonymizing RFID data sets, we will see later that our model considers both attacks; record and attribute linkages.

2.3 Trajectory Data

Trajectory data or moving objects data [1] [44] [55] [28] [36] is a collection of *trajectories* relating to some moving objects collected at certain points of times. Each trajectory belongs to a unique moving object and consists of a *sequence* of locations the moving object was reported at. An example of how trajectories are collected is the Octopus electronic system [2] [44] for Hong Kong residents where each Octopus smart RFID card holder gets to use the card to pay for transportation, shopping, parking, etc.

The trajectory data set can later be published for behavioral or pattern analysis, for example. Of course, users' IDs will be removed along with other identifying pieces of information so that no distinct user can be identified. However, publishing the data as collected may pose privacy threats to users. For instance, if a trajectory in a published data set shows that its pertinent Octopus card user, say Alice, frequently uses the same parking spot every morning for five days a week,

[2] www.octopuscards.com

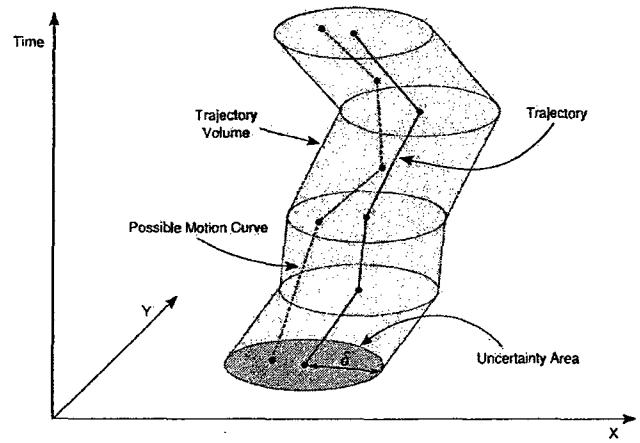


Figure 3: A graphical representation of an uncertainty trajectory volume [1]

it is highly probable that it is Alice's work place and it is located somewhere near the parking area. Alice's neighbor, Eve, knows where Alice works, and she knows that Alice uses the Octopus system. Moreover, Eve notices that in the same trajectory that person often makes purchases at the neighborhood store. Being unique, the trajectory was not only successfully linked to Alice, but also Eve now can see what other places Alice has been to at different times, revealing personal information Eve is not entitled to acquire.

A few recent works [1] [44] [55] present methods for anonymizing high-dimensional trajectory data based on the concept of K-anonymity [37]. In their work entitled $\mathcal{N}$ever $\mathcal{W}$alk $\mathcal{A}$lone, Abul et al. [1] exploited the uncertainty in the moving objects' whereabouts via their proposed privacy model (K, δ)-*anonymity*. In their 3-dimensional space representation, a trajectory is a polyline where the coordinates (x, y, t) of each point in the polyline represent the moving object's location (x, y) at a specific time t, as depicted in Figure 3.

Abul et al.'s (K, δ)-*anonymity* model requires that within a δ-radius proximity to each trajectory, a minimum of $K-1$ other trajectories co-exist, i.e., the cylindrical volume of radius δ centered

by a trajectory should encompass at least K trajectories so that no trajectory can successfully be linked to a specific individual with a probability of more than $1/K$. To achieve the previous privacy requirement for a target data set, Abul et al. [1] use *space translation* to change the location coordinates of some points on certain polylines.

Terrovitis et al. [44] present a privacy model that treats every location in a trajectory as sensitive information. This is unlike the set of QID in relational data, which is considered public knowledge that an attacker may acquire by different means. The authors assume that different adversaries hold different partial background knowledge about individuals' trajectories. Hence their objective is to prevent adversaries from uniquely identifying individuals based on their background knowledge about target victims' trajectories and, thus further compromising privacy by learning additional information about the target individuals' locations. The privacy model requires a user-defined threshold of breach probability P_{br} and uses suppression to achieve an anonymous data set. The experimental results in [44] suggest that higher values of P_{br} (equivalently, less anonymization) result in suppressing fewer location points.

Similar to the work in [44] where the different adversaries' partial knowledge about individuals' trajectories are considered quasi-identifiers, Yarovoy et al. [55] "personalize" quasi-identifiers for different trajectories. In other words, quasi-identifiers are chosen based on what locations and times adversaries may use to uniquely identify individuals. To produce an anonymous version of the original data set, K-anonymity is modeled by means of *space generalization*. That is, at certain times (QID), locations of a trajectory are generalized to a larger region of locations that contains at least $K-1$ other individuals. Consequently, the target victim's exact location becomes indistinguishable from the rest of the other individuals' locations that co-exist in the same region.

2.3.1 Trajectory Data vs. RFID Data

There is not a great difference between trajectory and RFID data types. In fact, the latter is one type of the former, since the movement of the RFID tag holder is reported at different times. We describe here how our proposed method varies from the aforementioned ones.

In the model proposed by Abul et al. [1], namely $\mathcal{N}$ever $\mathcal{W}$alk $\mathcal{A}$lone ($\mathcal{NWA}$), the authors assume a continuous trajectory in which the expected location is known at each point of time. As we will see later, this assumption does not commonly hold for RFID data because readers are placed in specific locations where RFID tags are detected and the location of any moving object between two consecutive detections is unknown. Moreover, the $\mathcal{NWA}$ framework uses space translation, which entails changing the actual locations of the moving objects to achieve (K, δ)-*anonymity*, rendering the data untruthful for further analyses. Our model, however, preserves data truthfulness by using suppression.

Another major aspect is the consideration of QID. While our model is built on the assumption that any combination of location and timestamp is public knowledge and may be used by adversaries to lunch attacks, both [44] and [55] require the different adversaries' background knowledge to already be known to the data holder in order to perform the anonymization. In real-life scenarios, that assumption becomes unrealistic because it requires exploring potential adversaries and their prior knowledge about each moving object.

One last important point is that none of the three works [1] [44] [55] consider the presence of a sensitive attribute, and thus ignore attribute linkages. In fact, [44] considers any location in a trajectory to be sensitive information. Our model not only anonymizes RFID data, but it is also for anonymizing sequential data, as we will see in Chapter 5.

2.4 Summary

To summarize this chapter, we discussed different types of data that pose privacy threats to individuals when published directly as collected. We presented some of the prominent methods that attempt to tackle the privacy issues concerning each type of data set. Some of the works take the K-anonymity model as a building block and extend upon it to satisfy certain privacy requirements. There are more techniques and approaches exist in the literature for *privacy-preserving data publishing* (*PPDP*); however, it does not serve our objective in this context to discuss these techniques. Fung et al. [11] survey recent and existing techniques for PPDP using descriptive insight and illustrative examples. In the following chapter, we present a formal definition of the problem of anonymization in RFID data, i.e., we formally define the privacy threats and our LKC-privacy model.

Chapter 3

Problem Definition

In this chapter, we formally define our new privacy model – LKC-privacy. We begin by defining the structure of RFID data in the table to be released, which we call an object-specific path table (Section 3.1). Then, in Section 3.2, we formalize the privacy threats that we mentioned in Chapter 1, namely record linkage and attribute linkage, in the content of RFID data. Next, we formalize our privacy model and the problem statement in Sections 3.3 and 3.4, respectively.

3.1 Object-Specific Path Table

In this work, we assume that an RFID tag is either attached to a moving object or placed on an item that is carried by a moving person, for example, passengers in a transit system. An RFID reader is placed at location loc. At any time t when a reader detects an RFID-tagged object, the reader creates an RFID data record of the form $\langle EPC, loc, t \rangle$, where EPC is a unique electronic product code for that object. We assume that each row in an RFID data table represents a distinct

person, and thus, the streams of RFID records relating to the same EPC are grouped together in chronological order constituting a *path*. A *path* is a sequence of *pairs*, where a *pair* $(loc_i t_i)$ indicates that the object has visited location loc_i at time t_i. In a path, denoted by $\langle (loc_1 t_1) \dots (loc_n t_n) \rangle$, pairs are grouped in a chronological order according to their timestamps. t_i is considered a timestamp if it is the object's entry time to a certain location. The assumption is that an object stays in the same location until the next pair shows that the object has entered another location at a later timestamp [18]. Therefore, consecutive pairs (duplicates) indicating an object's presence in the same location are removed from the path. There is no restriction on visiting locations; an object can revisit the same location, however, at later timestamps. As an illustration, consider the path $\langle a1 \rightarrow b3 \rightarrow b4 \rightarrow b6 \rightarrow c7 \rightarrow b8 \rangle$, pairs $b4$ and $b6$ should be removed because the object stayed in location b for times 3, 4, and 6, thus only the entrance timestamp 3 is considered. Moreover, notice that despite removing $b4$ and $b6$, we kept $b8$ because the object was in a different location, as the pair just before shows. In any path, timestamps are always increasing, so sequences such as $a1 \rightarrow b1$ are not valid as an object can only be at one location at any point of time.

From this point onwards, whenever we refer to a "record" in the context of data sets or tables, we mean a record of the form

$$\langle (loc_1 t_1) \rightarrow \dots \rightarrow (loc_n t_n) \rangle : s_1, \dots, s_p : d_1, \dots, d_m,$$

where $\langle (loc_1 t_1) \rightarrow \dots \rightarrow (loc_n t_n) \rangle$ is a moving object's path, $s_i \in S_i$ are sensitive attributes, and $d_i \in D_i$ are attributes containing the object data that form the set of quasi-identifier (QID). In this work, we target the paths and the sensitive attributes. The set of QID attributes is considered to be of a relational data type, and it can be anonymized by using existing anonymization methods [16] [30] [26] [27] [38] [48] for such data. We represent a given data set in an *object-specific path table*

T. An *object-specific path table* T is a collection of records taking the abovementioned form, however, we only consider the paths and the sensitive attributes.

3.2 Privacy Threats

In this section, we formally define the attacks on individuals' privacy when the data holder releases an object-specific path table T, e.g., for statistical study or data analysis. To "protect" individuals from being identified, the data holder removes explicit identifiers such as EPC, name, SSN, and DoB. However, we shall see that simply removing explicit identifying information does not protect individuals' privacy. The paths and the object-specific attributes are kept intact because we assume that they are essential for achieving the objective of publishing the data, e.g., data analysis. As we explained in Chapter 1, an adversarial recipient of published data T can attempt to compromise individuals' privacy by identifying their paths and/or sensitive values. We refer to an individual in T for whom privacy is compromised as a *target victim* V. We also assume that the adversary has gathered information about a target victim V in the form of location and timestamp pairs previously visited by the victim. We refer to such information as the adversary's *background knowledge*. Discussing the methods that an adversary uses to gather background knowledge about victims is not within the scope of this research. However, a simple example is that people, such as family members, friends, and neighbors, commonly share various types of information about each other. This example can be included under the general term of social engineering [41] [51].

The assumption is that an adversary's background knowledge about a victim V contains at most L pairs, which form a subsequence of the victim's path in T. The background knowledge is

denoted by $\kappa = \langle (loc_1 t_1) \rightarrow \ldots \rightarrow (loc_z t_z) \rangle$, where $z \leq L$. κ narrows down all the records in T to a group of records $G(\kappa)$ that "matches" κ. In other words, κ is a subsequence of each path in the identified group of records. The notion of *matching* is formally defined as follows.

Definition 3.2.1 (Matching) A pair $(loc_i t_i)$ *matches* a pair $(loc_j t_j)$ if $loc_i = loc_j$ and $t_i = t_j$. A path p_x *covers* a path p_y if, for every pair $(loc_y t_y)$ in p_y, there exists a pair $(loc_x t_x)$ in p_x that matches $(loc_y t_y)$. A record *matches* κ if the path of the record covers κ. ■

For example if we choose $\kappa = \langle e4 \rightarrow c7 \rangle$ in Table 1, we would have a match in $EPC\#1$: $\langle a1 \rightarrow d2 \rightarrow b3 \rightarrow e4 \rightarrow f6 \rightarrow c7 \rangle : On - welfare$ but not in $EPC\#4$: $\langle d2 \rightarrow f6 \rightarrow c7 \rightarrow e8 \rangle : Retired.$

In Chapter 1, Example 1.1.1 illustrated two types of privacy attacks: record linkage and attribute linkage. Next, we formally define these privacy attacks. An attacker exploits his background knowledge about a target victim V to build a set of candidate records $G(\kappa)$. One of the records in $G(\kappa)$ is V's record. Based on $G(\kappa)$, an adversary could perform:

1. *Record linkage*: the size of the set of the candidate records $G(\kappa)$ is denoted by $|G(\kappa)|$. Record linkage takes places when the adversary's background knowledge κ renders $|G(\kappa)|$ small. If victim V's record is successfully identified, the adversary would be able to learn about all of the pairs in V's path, along with V's sensitive value.

2. *Attribute linkage*: this privacy threat arises from the ability of an adversary to infer V's sensitive value s. The confidence of inferring s from the set $G(\kappa)$ is denoted by $Conf(s|G(\kappa))$, and is computed as follows:

$$Conf(s|G(\kappa)) = \frac{|G(\kappa \bigcup s)|}{|G(\kappa)|},$$

where the numerator is the number of records from $G(\kappa)$ containing s. The inference confidence $Conf(s|G(\kappa))$ is therefore the fraction of records that contain the sensitive value s in $G(\kappa)$. The privacy threat resulting from this type of attack is proportional to the value of $Conf(s|G(\kappa))$; high values imply high privacy risks.

3.3 Privacy Models

Our objective is to provide a method that "processes" the object-specific path table T so that it is transformed from its raw state to another version. By "processing" table T, we are referring to anonymization, which is defined in Section 3.4. The resultant version of the anonymized table T, T', should thus be protected against the privacy threats we defined in the previous section. Next, we formalize our privacy model, *LKC-privacy*, to produce the anonymous table T'. But first, we divide LKC-privacy into two privacy models: *LK-anonymity* and *LC-dilution*. Each of these privacy models protects against a different privacy threat; record linkages and attribute linkages, respectively. *LK-anonymity* and *LC-dilution* are formally defined next. As mentioned earlier, κ is the adversary's background knowledge of at most L pairs in length. Thus, any subsequence q having a length of up to L pairs from any path in T could be a potential instance of κ.

Definition 3.3.1 (LK-anonymity) *An object-specific path table T* satisfies LK-anonymity *if and only if $|G(q)| \geq K$ for any subsequence q with $|q| \leq L$ of any path in T, where K is a positive anonymity threshold.* ■

Definition 3.3.2 (LC-dilution) *Let S be a set of data holder-specified sensitive values from sensitive attributes $S_1, \ldots, S_m$. An object-specific path table T* satisfies LC-dilution *if and only if*

$Conf(s|G(q)) \leq C$ *for any* $s \in S$ *and for any subsequence* q *with* $|q| \leq L$ *of any path in* T*, where* $0 \leq C \leq 1$ *is a confidence threshold.* ■

Definition 3.3.3 (LKC-privacy) *An object-specific path table* T satisfies LKC-privacy *if* T *satisfies both the* LK*-anonymity and the* LC*-dilution conditions.* ■

When applying LKC-privacy, an adversary's chance to succeed using record linkage and attribute linkage attacks is bounded to $\leq 1/K$ and $\leq C$, respectively. The former probability is due to LK-anonymity and the latter is due to LC-dilution. Higher values of the parameters L, K, and C provide greater protection against privacy threats. This is further studied in Chapter 5.

It is worth mentioning that the probability of a successful attribute linkage attack in T', which is $\leq C$, represents the confidence of inferring *only* those sensitive values previously set by the data holder. A demonstration Follows. The domain $\{On - welfare, \ldots, Part - time\}$ of the sensitive attribute Employment Status S_{es} in Table 1 includes sensitive information $s_i \in S_{es}$ about passengers. Based on the data holder's degree of sensitivity, the data holder specifies a set of values S (Definition 3.3.2) from S_{es} to be considered more sensitive than others, and thus in need of more protection. For example, *On-welfare* could be a sensitive value while *Part-time* may not be. The maximum probability of $C = 50\%$ in the anonymized Table 2 is, therefore, the probability of successfully inferring the sensitive value *On-welfare* about a target victim V. Allowing a data holder-specified set of sensitive values S reflects the flexibility in our LKC-privacy that allows it to accommodate different privacy requirements.

3.4 Problem Statement

In the previous section, we stated that our goal is to anonymize an object-specific path table T, which is susceptible to the privacy attacks defined in Section 3.2, by applying our LKC-privacy model. The anonymization process results in table T' and bounds the probabilities of successful privacy attacks to certain thresholds, depending on the input privacy requirement. In order to apply LKC-privacy on T, we enforce a sequence of *suppressions* on certain pairs. Suppressing a pair p means removing it from a path. The method for selecting pairs for suppression is discussed in Chapter 4. The pairs selected pairs to be suppressed are added to the set Sup. Our method employs *global suppression*, which means that for each pair $p_i \in Sup$, all instances of p_i in T are completely removed from T. For example, when we suppressed $a1$, $d2$, and $e4$ from Table 1, we removed all their instances and the resulting Table 2 did not contain any of the suppressed pairs.

In Subsection 2.1.1, we discussed two anonymization techniques: generalization and suppression. We use pair suppression to anonymize RFID data for the following reasons. First, generalization requires a pre-defined taxonomy tree (Figure 2) that classifies all domain values into a certain hierarchy. Preparing a taxonomy tree is an extra burden on the data holder, and taxonomy trees are not commonly available. Second, if generalization is used, and a target value is to be generalized to its parent node (given a taxonomy tree), then all siblings of that target value are generalized to the same parent node. This means that generalization affects not only the target pair but also its "neighbors" causing unneeded data loss. Suppression, on the other hand, provides more flexibility since only selected pairs (in Sup) will be removed without affecting other pairs. However, the actual information loss is dependent on the distribution of data and the availability of the hierarchy.

Having that said, solving the problem of anonymization by applying LKC-privacy converges to efficiently performing suppressions in T. A formal definition follows.

Definition 3.4.1 (Anonymization for RFID) *Given an object-specific path table T, an LKC-privacy requirement, and a set of sensitive values S, the problem of* anonymization for RFID *is to identify a transformed version T' that satisfies the LKC-privacy requirement by suppressing a minimal number of instances of pairs in T.* ■

Our LKC-privacy model offers flexibility over K-anonymity [38] and Confidence bounding [48], which we introduced in Chapter 2. That is, setting the parameters of our model to $L = \infty$ and $C = 100\%$ metamorphoses the privacy model to K-anonymity, where only the anonymity threshold K is the input parameter. Similarly, fixing the parameters to $L = \infty$ and $K = 1$ transforms LKC-privacy into confidence bounding, where only the inference confidence is the input parameter.

The best-case scenario is to find T' such that T' satisfies the given LKC-privacy requirement *and* has the highest utility due to suppressing the minimum number of pairs. Such a scenario is described as the optimal solution of the anonymization problem (T', in this case, would be the best solution); meaning that T' is anonymous with respect to a given LKC-privacy requirement, and, at the same time, has the least number of suppressed pairs when compared to all other possible anonymous versions of T. However, achieving optimal LKC-privacy, i.e., finding the best solution, is NP-hard because it has been proven that achieving optimal K-anonymity and optimal confidence bounding is NP-hard [29] [48]. In order to achieve the best solution, we need to enlist all 2^n possible pair suppressions in T and choose T' as stated above, where n is the number of

distinct pairs in T. Needless to say, such a naïve method requires exhaustive searches as the number of possible combinations grows exponentially with n. Hence, in the next chapter we propose a heuristic approach to efficiently use suppression in order to find the best sub-optimal solution. In this case, the achieved T' satisfies a given LKC-privacy requirement, but may not be the most useful table among other possible solutions.

3.5 Summary

In this chapter, we formalized the representation of RFID data as object-specific path table T, which is a collection of records. Each record in T belongs to a distinct individual, and is composed of the moving object's path combined with its sensitive information. Then, we formally defined the privacy threats in the context of RFID data, namely record and attribute linkages. After that, we presented a formal definition for our privacy model LKC-privacy, followed by a formalization of the anonymization problem for RFID data. An optimal solution implies that the anonymized table T' satisfies a given LKC-privacy requirement *with* the least possible amount of data loss. We reasoned that achieving optimal solution is NP-hard. In the next chapter we propose an efficient algorithm that greedily searches for a "good" solution instead of "the best".

Chapter 4

The Anonymization Method

In order to enforce LKC-privacy on an object-specific path table T, we argued in the previous chapter that we use suppression to leave out selected pairs $\in Sup$ from T. In order to build up Sup, we first identify all the possible subsequences of any path in T that *violate* a given LKC-privacy requirement. Notice that our goal is not only to produce an anonymized RFID table T', but also to ensure T''s usefulness for further diversified data usage. Thus, the dual objective is to efficiently remove all *violations* from T to satisfy the LKC-privacy requirement, and to maintain data usefulness. Section 4.1 defines the notion of violation and critical violation. Section 4.2 presents our proposed greedy algorithm for efficiently identifying and removing all (critical) violations.

4.1 Identifying Violations

Given an object-specific path table T, we want to remove all violations from T so that it satisfies a given LKC-privacy requirement. An adversary knows at most L pairs about a target victim's path.

Informally speaking, any subsequence q of any path in T, with q having a maximum length of L, is a violation if q does not adhere to LK-anonymity and/or LC-dilution (Definitions 3.3.1 and 3.3.2). In other words, a subsequence q in T is a violation if at least one of the following conditions holds true: (1) q does not occur in at least K records, and (2) the confidence of inferring a sensitive value $s \in S$ within the group of records containing q, $G(q)$, exceeds the maximum threshold C. By removing all violations from T, no adversary with a background knowledge κ can perform record or attribute linkage attacks because κ could be any of the eliminated violations. The anonymized table T' is, therefore, immune against privacy attacks. A formal definition of violation, followed by an example is presented next.

Definition 4.1.1 (Violation) *Let q be a subsequence of a path in T with $|q| \leq L$ and $|G(q)| > 0$. q is a* violation *with respect to an LKC-privacy requirement if $|G(q)| < K$ or $Conf(s|G(q)) > C$.* ■

Example 4.1.1 In Table 1, we want to apply the following LKC-privacy requirement: $L = 2$, $K = 2, C = 50\%$, and $S = \{On - welfare\}$. A sequence $q_1 = \langle e4 \rightarrow c7 \rangle$ is a violation because it violates LK-anonymity since $|G(q_1)| = 1 < 2$. A sequence $q_2 = \langle d2 \rightarrow f6 \rangle$ is a violation because it violates LC-dilution, since $Conf(On - welfare|G(q_2)) = 67\% > 50\%$. ■

Here, we mention two properties of violation. **Property 1**: if a subsequence q violates LK-anonymity, i.e., $|G(q)| < K$, then all super sequences of q are violations as well. Let q' be a super sequence of q; according to property 1, q' is a violation because $|G(q')| \leq |G(q)| < K$. We note two implications for this property. The first implication suggests that since any super sequence q' of a violation q is also a violation, the total number of violations is potentially huge. As a result, enumerating all violations and removing them from T is not a practical solution. The

second implication of property 1 contains the opposite semantic to that of the previous implication. That is, if T satisfies L_2K-anonymity, then it satisfies L_1K-anonymity as well, for $L_1 \leq L_2$. This is because $|G(q_1)| \geq |G(q_2)| \geq K$. **Property 2**: assume that q violates only LC-dilution; that is, $Conf(s|G(q)) > C$ and $|G(q)| \geq K$. q', which is a super sequence of q, is not necessarily a violation since $Conf(s|G(q')) \geq Conf(s|G(q))$ is not necessarily true. We ensure that a table T satisfies $Conf(s|G(q)) \leq C$ not only for every $|q| = L$, but for every $|q| \leq L$.

As we mentioned earlier, identifying all possible violations in T for a given LKC-privacy requirement is unfeasible. We argue that we can save a significant amount of effort in searching for all violations by stopping the search for further violations at a certain point in the search process. We reach such a point the moment we encounter "critical violations". We then show that in table T, violations exist if and only if critical violations exist. Critical violations are minimal sequences, and are defined below. We follow the definition with an example illustrating the difference between a violation and a critical violation.

Definition 4.1.2 (Critical Violation) *A violation q is a* critical violation *if every proper subsequence of q is a non-violation.* ■

Example 4.1.2 In Table 1, assume that $K = 2, C = 50\%$, and $S = \{On-welfare\}$. An example of a critical violation would be the sequence $q_1 = \langle e4 \rightarrow c7 \rangle$ because: (1) $|G(q_1)| = 1 < 2$, and (2) both subsequences of q_1: $\langle e4 \rangle$ and $\langle c7 \rangle$ are non-violations. On the other hand, the sequence $q_2 = \langle d2 \rightarrow e4 \rightarrow c7 \rangle$ is an example of a violation but not a critical violation, since there is a subsequence of q_2, $\langle e4 \rightarrow c7 \rangle$, which is actually a violation. ■

From Definition 4.1.2, we can conclude that a violation is a super sequence of a critical violation. This leads us to the following observation.

Observation 4.1.1 *A table T' satisfies LKC-privacy if and only if T' contains no critical violation, because each violation is a super sequence of a critical violation. Thus, if T' contains no critical violations, then T' contains no violations.* ■

Observation 4.1.1 clearly sets out the goal for our next algorithm. Algorithm 1 describes our proposed methodology for efficiently generating all critical violations in table T that does not satisfy a given LKC-privacy requirement. According to Definition 4.1.2, we basically properly extend subsequences that are non-violations of size i to super sequences of size $i + 1$. That is, one additional pair is added to the current non-violation, and the resultant subsequence q' is checked against the given LKC-privacy requirement. If q' is a violation, then it is a critical violation. Critical violations (if any) appear in the extended subsequences $i+1$. We denote non-violations of size i by U_i and critical violations of size $i + 1$ by V_{i+1}.

Algorithm 1 includes a summary of the steps required to generate critical violations. Initially, a set of inputs is provided by the data holder: a raw RFID path table T, an LKC-privacy requirement, and a set of data holder's sensitive values S. In Line 1, we first gather in the candidate set $Cand_1$ all of the unique pairs of all the paths in T. Then in Line 4, we scan T once to find $|G(q)|$ for every $q \in Cand_i$ and also compute $Conf(s|G(q))$ for every $s \in S$. In Lines 5 to 13, we verify if each candidate subsequence $q \in Cand_i$ violates the given privacy requirement, i.e., $|G(q)| < K$ or $Conf(s|G(q)) > C$. If candidate q passes the verification, then it is a (critical) violation and is put in the critical violation set V_i; otherwise q is put in the non-violation set U_i. We note that if $Cand_i$,

Algorithm 1 Generate Critical Violations (GenViolations)

Input: Raw RFID path table T
Input: Thresholds L, K, and C.
Input: Sensitive values S.
Output: Critical violations V.

```
1: let candidate set Cand_1 be the set of all distinct pairs in T;
2: i = 1;
3: repeat
4:    scan T once to obtain |G(q)| and Conf(s|G(q)) for every sequence q ∈ Cand_i and for every
      sensitive value s ∈ S;
5:    for all sequence q ∈ Cand_i do
6:       if |G(q)| > 0 then
7:          if |G(q)| < K or Conf(s|G(q)) > C for any s ∈ S then
8:             add q to V_i;
9:          else
10:            add q to U_i;
11:         end if
12:      end if
13:   end for
14:   ++i;
15:   generate candidate set Cand_i by U_{i-1} ⋈ U_{i-1};
16:   for all sequence q ∈ Cand_i do
17:      if q is a super sequence of v for any v ∈ V_{i-1} then
18:         remove q from Cand_i;
19:      end if
20:   end for
21: until i > L or Cand_i = ∅
22: return V = V_1 ⋃ ... ⋃ V_{i-1};
```

where $i = 1$, contains a violation, then this violation is a critical violation because its subsequence, an empty set, is a non-violation. After increasing the value of i by 1, Line 15 self-joins the set of non-violations U_{i-1} to create a candidate set $Cand_i$. Note that all critical violations in V_{i-1} are excluded when generating $Cand_i$ because violations (if any) in $Cand_i$ must only be those that are critical, according to Definition 4.1.2. To clarify how two sequences of size i can be joined together to create a super sequence of size $i + 1$, we present the following definition:

Definition 4.1.3 (Sequences Joining) *Sequence* $q_x = \langle (loc_1^x t_1^x) \rightarrow \ldots \rightarrow (loc_i^x t_i^x) \rangle$ *can be joined*

with sequence $q_y = \langle (loc_1^y t_1^y) \rightarrow \ldots \rightarrow (loc_i^y t_i^y) \rangle$ *only when the first* $i-1$ *pairs of both sequences are identical and* $t_i^x < t_i^y$. *The resulting joined sequence is* $\langle (loc_1^x t_1^x) \rightarrow \ldots \rightarrow (loc_{i-1}^x t_{i-1}^x) \rightarrow (loc_{i-1}^y t_{i-1}^y) \rangle$. ■

In Lines 16 to 20, we verify that each candidate sequence $q \in Cand_i$ is not a super sequence of a critical violation $v \in V_{i-1}$. If it is, then q is simply removed from $Cand_i$. This is done in compliance to Observation 4.1.1. In Line 21, we terminate the running loop for generating all critical violations in T when i exceeds the maximum length of the adversary's background knowledge L, or when self-joining the set of non-violations U_{i-1} yields an empty candidate set $Cand_i$. Lastly, Algorithm 1 returns all of the entire critical violations found in T by combining the sets $V_1, \ldots, V_{i-1}$ into a unified set V. Following is a demonstrative example.

Example 4.1.3 Suppose that the data holder of the raw Table 1 wishes to generate all the critical violations for $L = 2$, $K = 2$, $C = 50\%$, and $S = \{On - welfare\}$. First, we generate the candidate set that includes all distinct pairs in Table 1 $Cand_1 = \{a1, d2, b3, e4, c5, f6, c7, e8, e9\}$. Then, we identify all critical violations in $Cand_1$ (by scanning Table 1 for $|G(q)| < K$ and $Conf(On - welfare|G(q)) > C$ for every $q \in Cand_1$). Among all the pairs in $Cand_1$, $a1$ is a critical violation and is put in $V_1 = \{a1\}$. The non-violations are placed in $U_1 = \{d2, b3, e4, c5, f6, c7, e8, e9\}$. Next, we self-join the set of non-violations U_1 to generate $Cand_2 = \{d2b3, d2e4, d2c5, d2f6, d2c7, d2e8, d2e9, b3e4, b3c5, b3f6, b3c7, b3e8, b3e9, e4c5, e4f6, e4c7, e4e8, e4e9, c5f6, c5c7, c5e8, c5e9, f6c7, f6e8, f6e9, c7e8, c7e9, e8e9\}$. Again, we scan Table 1 once to identify the critical violations in $Cand_2$ and put them in $V_2 = \{d2b3, d2e4, d2f6, d2e8, d2e9, e4c7, e4e8\}$. The entire critical violations based on the enforced privacy requirement on the raw Table 1 are returned in the set $V = \{a1, d2b3, d2e4, d2f6, d2e8, d2e9, e4c7, e4e8\}$. ■

4.2 Anonymization Algorithm

In this section, we propose Algorithm 2 for greedily performing a sequence of suppressions on some selected pairs to transform a raw RFID data table T into an anonymous version T' that complies with a given LKC-privacy requirement. The intuition behind our algorithm is that it should not only produce an anonymous table T' that satisfies a given LKC-privacy requirement, but also ensure the usefulness of the data in T' for further data analysis, study, ..., etc. After identifying all of the critical violations, suppression is performed on a selected pair p in each iteration. Generally speaking, suppressing a pair p delivers more privacy to T because by suppressing p we are removing critical violations; on the other hand, suppressing p also causes information loss since all instances of p are removed from T. We use a greedy selection function to pick a pair p, where p would be the best candidate for suppression in the current iteration. By "the best candidate" we mean that, based on the selection function, suppressing p results in removing the maximum number of critical violations *and* the minimum number of pair instances in T. Next, we formally define $Score(p)$, our greedy selection function.

$$Score(p) = \frac{PrivGain(p)}{InfoLoss(p)}, \tag{2}$$

where $PrivGain(p)$ represents the total number of critical violations that contain the pair p, and $InfoLoss(p)$ represents the total number of times pair p occurs in T (or the support of q). Thus, in each iteration Algorithm 2 chooses a pair with the maximum $PrivGain(p)$ to $InfoLoss(p)$ ratio.

An explanation of the anonymization process for RFID data that we constructed in Algorithm 2 follows. The input data include a raw RFID path table T, an LKC-privacy requirement, and a set of

Algorithm 2 RFID Data Anonymizer

Input: Raw RFID path table T
Input: Thresholds L, K, and C.
Input: Sensitive values S.
Output: Anonymous T' that satisfies LKC-privacy.

1: V = Call GenViolations(T, L, K, C, S) in Algorithm 1;
2: build the Critical Violation Tree (CVT) with Score Table;
3: **while** Score Table is not empty **do**
4: select winner pair w that has the highest $Score$;
5: delete all critical violations containing w in CVT;
6: update $Score$ of a candidate x if both w and x were contained in the same critical violation;
7: remove w in Score Table;
8: add w to Sup;
9: **end while**
10: for every $w \in Sup$, suppress all instances of w from T;
11: **return** the suppressed T as T';

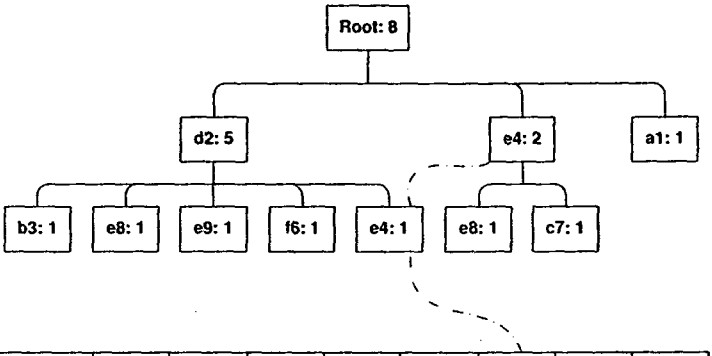
sensitive values S. First, in Line 1, Algorithm 2 generates all the critical violations in T by calling Algorithm 1; V contains the critical violations. Next, in Line 2, the algorithm structures V in a tree-like representation, called a *Critical Violation Tree (CVT)*, and builds a Score Table. The CVT and the Score Table will be discussed shortly in Subsection 4.2.1. In Lines 3 to 9, the algorithm iteratively goes through the Score Table to identify the pair p that maximizes the $PrivacyGain(p)$ and minimizes the $InfoLoss(p)$ – the pair with the highest $Score(p)$ – and removes that pair from all of the critical violations containing it. We refer to such a pair as the winner pair w. By removing w from all of the critical violations containing it, we eliminate these critical violations from the CVT. After removing w from the CVT, the algorithm updates the $Score$ of each pair x that co-existed in the same critical violation with w. Then, w is removed from the Score Table and added to Sup, the set of pairs to be suppressed from T. Suppressing all the pairs in Sup is achieved in Line 10, where T is scanned once to suppress every instance of $w \in Sup$. Finally in Line 11, the algorithm returns the anonymized table T' that satisfies the given LKC-privacy requirement.

4.2.1 Critical Violation Tree

Identifying critical violations containing w and updating the affected pairs in the Score Table (Lines 5 and 6 of Algorithm 2, respectively) are two operations that require careful handling. In order to maintain efficiency in our method, we devise a convenient way to perform these operations in an efficient manner. We propose the concept of a *Critical Violation Tree (CVT)*. CVT is a data structure that keeps track of critical violations, as depicted in Figure 4. A definition of CVT is presented nexr.

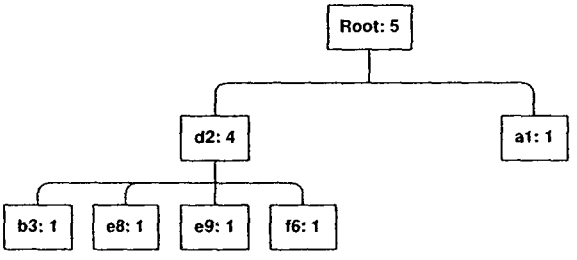
Definition 4.2.1 (Critical Violation Tree (CVT)) *A CVT is a tree structure that represents each critical violation as a tree path from root to leaf. Each node keeps track of a count of critical violations sharing the same prefix. The count at the root is the total number of critical violations. A CVT has a Score Table that maintains every pair p that is a candidate for suppression, together with its $PrivGain(p)$, $InfoLoss(p)$, and $Score(p)$. Each candidate pair p in the Score Table has a link, denoted by $Link_p$, that links up all the nodes in a CVT containing p. $PrivGain(p)$ is the sum of the counts of critical violations on $Link_p$.* ■

Figure 4 shows a CVT structure of the critical violations generated in Example 4.1.3. The root node contains the number (count) 8, meaning that there are 8 critical violations in total. Each node below the root represents a pair and contains a number (count) that indicates the number of critical violations that share the same prefix. For example, node $d2 : 5$ indicates that there are 5 distinct critical violations sharing $d2$ as a prefix (in this case, $d2$ is the first pair): $d2b3$, $d2e8$, $d2e9$, $d2f6$, and $d2e4$. The Score Table, below the tree, iteratively keeps track of every pair's $Score$. Next, we exemplify the process of deleting a winner pair w from the CVT and updating the Score Table.



Pair	d2	e8	b3	e9	f6	e4	c7	a1
PrivGain	5	2	1	1	1	3	1	1
InfoLoss	4	3	3	3	6	2	5	1
Score	1.3	0.7	0.3	0.3	0.2	1.5	0.2	1

Figure 4: CVT of all identified critical violations



Pair	d2	e8	b3	e9	f6	a1
PrivGain	4	1	1	1	1	1
InfoLoss	4	3	3	3	6	1
Score	1	0.3	0.3	0.3	0.2	1

Figure 5: CVT after deleting the winner pair $e4$ and updating the Score Table

In the initial Score Table of Figure 4, $e4$ is the winner pair because it has the highest $Score$ among the rest of the candidate pairs. Line 5 of Algorithm 2 is achieved by removing each $e4$ node from the CVT, and thus removing the critical violations that contain $e4$. As mentioned in Definition 4.2.1, $Link_{e4}$ is used to conveniently traverse the path that links up all the $e4$ nodes. Deleting each $e4$ node from the CVT entails removing the entire subtree rooted by that $e4$ node. Removing critical violations in this manner provides efficiency in our method. Continuing in Algorithm 2, next we need to update the $Score$ of the affected pairs (contained in the same critical violation with $e4$) in the Score Table. When removing a node n_x from CVT, the number held in n_x's parent node is decremented by the number contained in n_x. Figure 5 shows the updated Score Table after $e4$ has been removed from the CVT in Figure 4. Consequently, the node $d2 : 5$ is decremented by 1, the entire branch rooted by the node $e4 : 2$ is deleted, and the node $Root : 8$ is decremented to $Root : 5$. Deleting all nodes of a pair p means that $PrivGain(p) = 0$. Any pair with $PrivGain(p) = 0$ is removed from the updated Score Table, as demonstrated in Figure 5 showing the removal of $e4$. Similarly, any affected pair x with an updated $PrivGain(x) = 0$ is also removed from the Score Table; e.g., deleting $e4$ causes $PrivGain(c7) = 0$ and thus removes $c7$ from the updated Score Table in Figure 5.

4.3 Summary

In this chapter, we presented our novel algorithm for anonymizing a raw RFID data table. The result of the anonymization process is a new version of the raw table that has been transformed to satisfy a given LKC-privacy requirement along with the consideration of the information utility

factor. We first introduced the notion of violation and critical violation. Our goal was to remove all critical violations from the raw table. We presented our algorithm for generating all critical violations. Next, we discussed our greedy algorithm for anonymization. Critical violations were represented in a data structure called a Critical Violation Tree (CVT) for the sake of efficiently performing suppression. We removed critical violations by eliminating a particular pair, which we called the winner pair w, from the CVT. The choice of w was based on a greedy selection function called $Score$ that considers both criteria, the privacy gain and the information loss, of any candidate pair p in the Score Table. Pair w is the pair with the highest score among other candidate pairs, meaning that eliminating w will remove more critical violations and at the same time will suppress the minimal number of pair instances from the raw table. At the end, the anonymized table is produced. In the next chapter, we will test our algorithm by using two different data sets, and compare our method with that of traditional K-anonymity.

Chapter 5

Empirical Study

In Chapter 4, we devised a novel algorithm for anonymizing an RFID data table. In this chapter, we carry out several evaluations of the performance of our method and present a comparison between our LKC-privacy and the traditional K-anonymity model. We begin by introducing and explaining the data sets we used in our experiments (Section 5.1). Then, we present the different evaluation criteria that were considered throughout the experimental evaluation (Section 5.2). Finally, we provide a broad summary of the empirical result in Section 5.3.

We used an Intel Core2 Quad 2.4GHz PC with 2GB of RAM to conduct our experiments. In all of the experiments, the $Score$ is calculated based on Equation 2, unless otherwise specified. We considered three evaluation criteria: the quality of the anonymized data set vis-à-vis information loss, efficiency, and the scalability of our anonymization technique. We employed two raw data sets throughout our evaluation procedure: *Subway* and *MSNBC*. Below, we explain each data set in detail.

5.1 Data Sets

Two data sets were employed for evaluation. The first data set, called *Subway*, is a simulation of a subway transit system containing 20,000 passengers. The data set simulates the travel route each passenger follows when moving between the 26 stations of the transit system for a period of 24 hours [1]. The object-specific path table consists of 20,000 records; each corresponds to the route of a distinct passenger. The dimensionality of this data set is obtained by the total possible combinations between stations and timestamps, i.e., 26×24 gives a total of 624 dimensions. In order to present diversity in the simulated traveling patterns, routes are constructed as follows: the majority of the passengers (16,000) do not exceed a path length of four pairs, a smaller grouping of passengers (3,500) do not exceed a path length six pairs, and only 500 passengers use the transit system every hour thus having a maximum path length of 24 pairs.

The other data set is a real-life web log data set, called *MSNBC* [3], which captures the web pages visited by users for a time period of 24 hours. Each of the 989,818 records in this data set shows the history of the visited web pages, classified into 17 categories (e.g., Sports, Business, Weather, etc.). Although the 17 categories are not physical locations, this data set is high-dimensional, which shares the same property of a typical RFID data set. For both data sets, we specified one attribute in the raw object-specific path table to be a sensitive attribute (similar to Table 1). The sensitive attribute describes the record owner's medical condition and consists of five domain values. One value among these five, namely *Cancer*, is chosen to be a sensitive value. Domain values of the sensitive attribute are randomly assigned to each record.

[1] In real-life subway transit systems, data records that describe routes of passengers are constructed (and collected later) when passengers use a smart/RFID card, e.g., http://www.carteopus.info/, to enter a station. We assume a similar environment in our simulation.

5.2 Evaluation Criteria

In the following subsections, we address the different evaluation criteria for our method and discuss the empirical results we achieved.

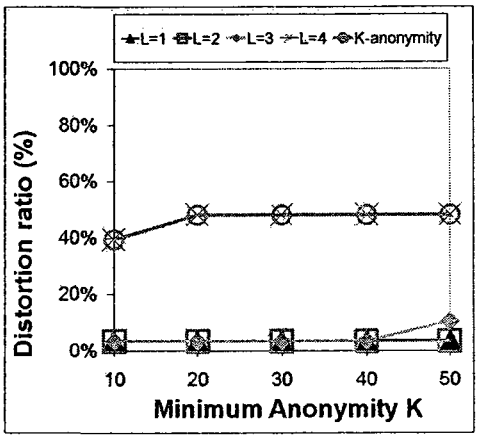
5.2.1 Data Quality

The purpose of this experiment is to measure the data quality (or utility) of a table T' that had been anonymized by applying the LKC-privacy model. Furthermore, we compare the result from applying our method to the result from applying traditional K-anonymity. In order to measure the data quality in T', we measure the information loss from the raw data table T. Information loss occurs due to suppression during the anonymization process. We use the *distortion ratio* as our metric for measuring information loss. The *distortion ratio* is the percentage of lost pair instances caused by suppression during the anonymization process for a given LKC-privacy requirement. Let $N(T)$ denote the number of pair instances in the raw data table T, and $N(T')$ denote the number of pair instances in the anonymized table T'. Hence, the distortion ratio is calculated as follows:

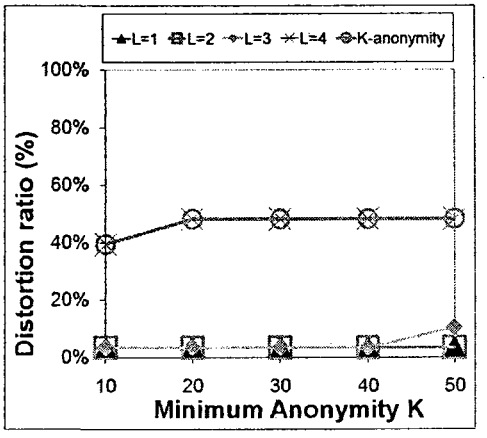
$$Distortion\ Ratio = \frac{N(T) - N(T')}{N(T)}. \tag{3}$$

Intuitively, achieving a high distortion ratio is not desirable because it implies low data quality. There is no specific benchmark for classifying the level of data quality; however, we do compare our method with the prominent K-anonymity model.

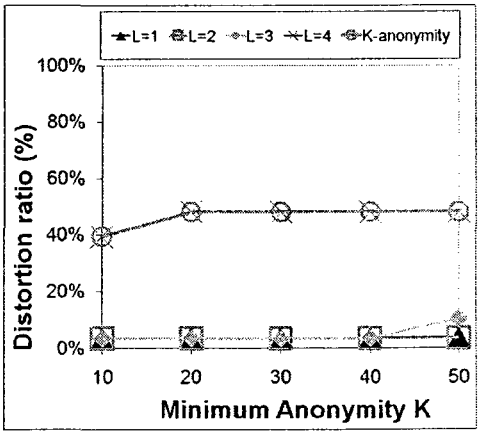
The series of experiments showing the distortion ratio on the *Subway* data set are depicted in Figure 6 [10]. The configuration of each set of these experiments is as follows: the maximum



(a) $C = 20\%$

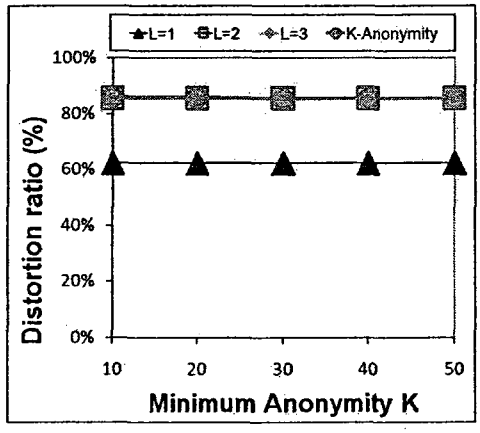


(b) $C = 60\%$

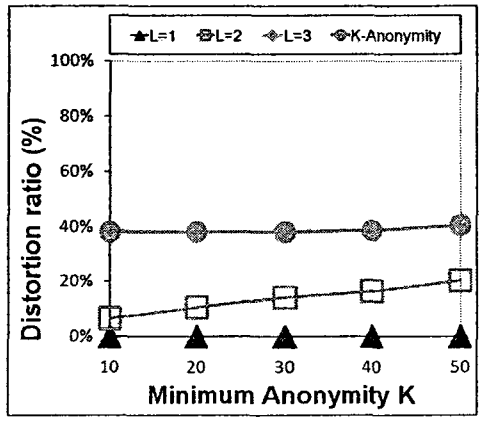


(c) $C = 100\%$

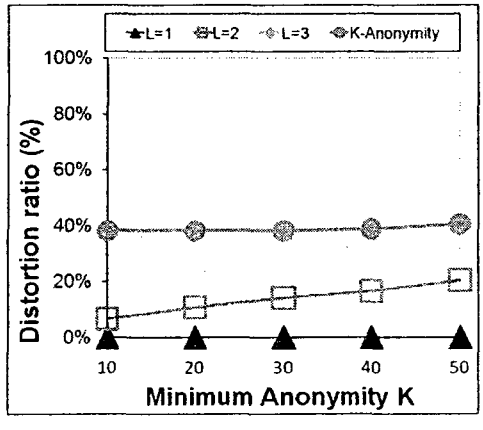
Figure 6: *Subway*: Distortion ratio vs. K



(a) $C = 20\%$



(b) $C = 60\%$



(c) $C = 100\%$

Figure 7: *MSNBC*: Distortion ratio vs. K

path length is set to be $1 \leq L \leq 3$, the anonymity threshold is increased by $10 \leq K \leq 50$ each round, and the different confidence thresholds $C = 20\%, 60\%, 100\%$ are assigned to each set of experiments, respectively. Figure 6 also depicts the distortion ratio when traditional K-anonymity is applied to the *Subway* data set. Generally speaking, the anonymity threshold K does not have a significant influence on the distortion ratio, which ranges between 3% and 10% for $1 \leq L \leq 3$, because even with a subsequence q with length $L = 3$ and $K = 50$, q would easily be shared by at least 50 records since the total size of the data set is 20,000 records. On the other hand, the distortion ratio caused by applying traditional K-anonymity is much higher; in fact, it never goes below 40%. This suggests that our method is capable of handling high-dimensional data more effectively by minimizing information loss.

We went one step further in our experiments on the *Subway* data set (Figure 6) and increased L to 4. The significant increase in the resulting distortion ratio is due to the fact that 80% of the records are four pairs in length (as mentioned in Section 5.1). Hence, setting $L = 4$ closely simulates a K-anonymity requirement. We also note an interesting observation; the distortion ratio of the *Subway* data set is not sensitive to the value of C. This observation is manifested in Figure 6(c), where setting $C = 100\%$ means that the LC-dilution is ignored. Therefore, suppression is primarily driven by LK-anonymity.

Figure 7 depicts the distortion ratio of the *MSNBC* data set [10]. Experiments were performed for the same values of confidence threshold C and anonymity threshold K as in the *Subway* data set, with maximum path length $1 \leq L \leq 3$. Figure 7(a) shows that when $C = 20\%$, the distortion ratio experiences a steady behavior for the different values of K because the anonymity threshold $K = 50$ is relatively small compared to the total number of records in the data set. Increasing K

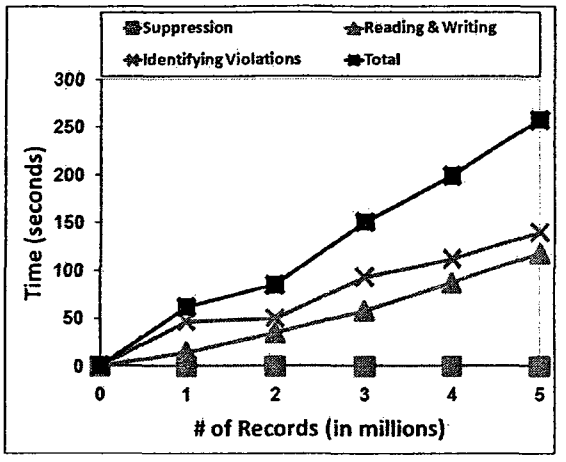


Figure 8: Scalability ($L = 3$, $K = 30$, $C = 60\%$)

from 10 to 50 does not have much impact. However, the distortion ratio stays above 60% when $C = 20\%$. This is due to the nature of how the values of the sensitive attribute are assigned to each record. As we mentioned earlier, there are five domain values in the sensitive attribute, and each record is randomly assigned one value. Therefore, the probability of a record having the sensitive value $Cancer$ is 20%, which is the same as the confidence threshold in this experiment. Thus, more suppression is required in order to satisfy LC-dilution. This fact is also reflected in Figures 7(b) and (c), in which the distortion ratio is significantly lower.

5.2.2 Efficiency and Scalability

Next, we evaluate the efficiency and the scalability of the proposed anonymization method. We tested our method against large data sets and monitored the runtime variations. Figure 8 depicts the test cases of five large synthetic RFID data sets, from one million records to five million records, and their corresponding runtimes. We captured variations of behaviours by having a variable maximum path length throughout all records in a data set. For example, in a data set, among several

groups of records, one group contains records with a maximum path length of four, while another group contains records with a maximum path length of six. The path length in any group ranges from one to the group's maximum path length. In general, the path length in each data set ranges from one to six, and the average maximum path length is four. We set $L = 3$, $K = 30$, and $C = 60\%$. Our method took 258 seconds to anonymize the five million-record data set, in which identifying critical violations took 140 seconds and reading the raw data file and writing the anonymous file took 118 seconds. Thanks to our Critical Violation Tree (CVT) structure in our proposed anonymization algorithm, suppression is done effectively, as it takes less than one second to complete.

5.3 Summary

Based on the observations from our experiments, we can summarize the results as follows. (1) The distortion ratio tends to increase with an increase of the maximum length L. (2) Changing the anonymity threshold K has an unnoticeable impact on the distortion ratio. However, this is not the case when K is increased to exceptionally large values, e.g., $K > 1000$. (3) The distortion ratio could be sensitive to the change of confidence threshold C, but this depends on the distribution of the sensitive value. (4) All the test cases described in this chapter and shown in Figures 6, 7, and 8, suggest that our method is effective and scalable for handling large data sets.

In this chapter, we performed intensive experiments to test our proposed anonymization method. We set up three evaluation criteria and provided a comparison between our method and that of traditional K-anonymity. We used two data sets to evaluate the performance of our method. The

first data set, *Subway*, is a simulation of the travel routes of 20,000 passengers, and the other data set, *MSNBC*, is a high-dimensional real-life web log that captures visited web pages by 989,818 users. To measure the effectiveness of our method on preserving data quality after anonymization, we presented the notion of distortion ratio. A lower distortion ratio means better data quality. We observed a significant decrease in the distortion ratio when evaluating our model compared to traditional K-anonymity. Thanks to the notion of adversary's background knowledge L in our model, the distortion ratio was decreased by up to 40% compared to that of traditional K-anonymity. Furthermore, we tested our method for efficiency and scalability. Five large synthetic RFID data sets – one million to five million records – were generated to examine the anonymization runtime and its variation. The results suggest that our method can handle large data sets efficiently.

Chapter 6

Conclusion

Radio Frequency IDentification (RFID) is a technology used for automatic object identification. It is being applied in many sectors, including manufacturing, healthcare, and transportation. Such data is rich in content and may be used for various real-world applications, e.g., academic research, statistical studies, and information mining. RFID data publishing plays an important role in these applications, due to its benefits and versatility. However, data publishing raises real privacy concerns. In this thesis, we demonstrate two types of privacy threats in the context of RFID data publishing: record linkage and attribute linkage. An adversary could utilize his background knowledge about a target victim to learn about the target victim's path (record linkage) or to infer the victim's sensitive value (attribute linkage).

In this thesis, we present the importance and applications of RFID data publishing, and discuss the potential privacy threats caused by RFID data publishing. We formally define the problem and present our proposed method for circumventing potential privacy threats.

6.1 Summary of Contributions

Given a raw RFID data set, we want to transform the data into a new anonymized version that is immunized against privacy attacks in accordance with a given privacy requirement. In addition, we want to ensure the usefulness of the anonymized data by keeping distortion to a minimum.

In this thesis, we formally define the privacy threats that arise in RFID data publishing. We argue that RFID data is characterized by being high-dimensional and sequential; this presents an obstacle in the anonymization problem. We propose and formally define *LKC-privacy*, a privacy model that addresses these challenges. We devise an efficient and scalable algorithm for achieving this privacy model. We carry out intensive experiments to evaluate the performance of our method. Our experiments on different data sets, including real-life data, suggest that our method is scalable and efficient, and that it can effectively handle extremely large data sizes. We also compared our method to traditional K-anonymity, and observed that the former yields a significant increase in data quality.

6.2 Future Work

We provide the following list of possible future research directions for the work we presented in this thesis:

- Implementing a local suppression strategy: locally suppressing a critical violation v means that only partial instances of v need to be removed from T. This will decrease the amount of information loss as the rest of the instances remain intact.

- Considering publishing updated versions of T: in practice, it is common that the data holder may want to publish an updated version of table T_1 as new RFID data arrives. The anonymized table T_1' had already been published and is out of the data holder's control. The updated table $T_2 = T_1 \pm records$ is to be published. We need to ensure that neither T_1' nor T_2' leak information that might help an attacker to crack the anonymity of the other anonymized table.

- Implementing a solution that "safely" anonymizes a table owned by more than one data holder: multiple data holders may want to collaborate and work together, e.g., to improve their services. Each data holder owns a different set of attributes of the same record owners. The goal is to merge the different tables into a single integrated one in a way that preserves individuals' privacy by not allowing any data holder to acquire any more detailed information (from other data holders) about individuals other than what the integrated table discloses.

Bibliography

[1] O. Abul, F. Bonchi, and M. Nanni. Never walk alone: Uncertainty for anonymity in moving objects databases. In *Proc. of the 24th IEEE International Conference on Data Engineering (ICDE)*, pages 376–385, April 2008.

[2] C. C. Aggarwal. On k-anonymity and the curse of dimensionality. In *Proc. of the 31st Very Large Data Bases (VLDB)*, pages 901–909, 2005.

[3] A. Asuncion and D.J. Newman. UCI machine learning repository, 2007.

[4] M. Barbaro, T. Zeller, and S. Hansell. A Face Is Exposed for AOL Searcher No. 4417749. *New York Times*, Aug 6, 2006.

[5] R. J. Bayardo and R. Agrawal. Data privacy through optimal k-anonymization. In *Proc. of the 21st IEEE International Conference on Data Engineering (ICDE)*, pages 217–228, Tokyo, Japan, 2005.

[6] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, 1981.

[7] Lawrence H. Cox. Suppression methodology and statistical disclosure control. *Journal of the American Statistical Association*, 75(370):377–385, June 1980.

[8] T. Dalenius. Finding a needle in a haystack - or identifying anonymous census record. *Journal of Official Statistics*, 2(3):329–336, 1986.

[9] J. Domingo-Ferrer and V. Torra. A critique of k-anonymity and some of its enhancements. In *ARES '08: Proceedings of the 2008 Third International Conference on Availability, Reliability and Security*, pages 990–993, Washington, DC, USA, 2008. IEEE Computer Society.

[10] B. C. M. Fung, K. Al-Hussaeni, and M. Cao. Preserving RFID data privacy. In *Proc. of the 2009 International Conference on RFID*, pages 200–207, Orlando, FL, April 2009. IEEE Communications Society.

[11] B. C. M. Fung, K. Wang, R. Chen, and P. S. Yu. Privacy-preserving data publishing: A survey on recent developments. *ACM Computing Surveys*, in press.

[12] B. C. M. Fung, K. Wang, A. W. C. Fu, and J. Pei. Anonymity for continuous data publishing. In *Proc. of the 11th International Conference on Extending Database Technology (EDBT)*, March 2008.

[13] B. C. M. Fung, K. Wang, L. Wang, and M. Debbabi. A framework for privacy-preserving cluster analysis. In *Proc. of the 2008 IEEE International Conference on Intelligence and Security Informatics (ISI)*, Taipei, Taiwan, June 2008.

[14] B. C. M. Fung, K. Wang, L. Wang, and P. C. K. Hung. Privacy-preserving data publishing for cluster analysis. *Data & Knowledge Engineering (DKE)*, in press.

[15] B. C. M. Fung, K. Wang, and P. S. Yu. Top-down specialization for information and privacy preservation. In *Proc. of the 21st IEEE International Conference on Data Engineering (ICDE)*, pages 205–216, Tokyo, Japan, April 2005.

[16] B. C. M. Fung, Ke Wang, and P. S. Yu. Anonymizing classification data for privacy preservation. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 19(5):711–725, May 2007.

[17] G. Ghinita, Y. Tao, and P. Kalnis. On the anonymization of sparse high-dimensional data. In *Proc. of the 24th IEEE International Conference on Data Engineering (ICDE)*, pages 715–724, April 2008.

[18] H. Gonzalez, J. Han, and X. Li. Flowcube: Constructing RFID flowcubes for multi-dimensional analysis of commodity flows. In *Proc. of the International Conference on Very Large Data Bases (VLDB)*, pages 1–19, Seoul, Korea, September 2006.

[19] H. Gonzalez, J. Han, and X. Li. Mining compressed commodity workflows from massive RFID data sets. In *Proc. of the International Conference on Information and Knowledge Management (CIKM)*, November 2006.

[20] K. Hafner and T. Zeller. Researchers Yearn to Use AOL Logs, but They Hesitate. *New York Times*, Aug 23, 2006.

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

[22] V. S. Iyengar. Transforming data to satisfy privacy constraints. In *Proc. of the 8th ACM SIGKDD*, pages 279–288, Edmonton, AB, Canada, July 2002.

[23] Markus Jakobsson, Ari Juels, and Ronald L. Rivest. Making mix nets robust for electronic voting by randomized partial checking. In *Proc. of the 11th USENIX Security Symposium*, pages 339–353, 2002.

[24] A. Juels. RFID security and privacy: a research survey. *IEEE Journal on Selected Areas in Communications*, 24(2):381– 394, February 2006.

[25] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Incognito: Efficient full-domain k-anonymity. In *Proc. of ACM SIGMOD*, pages 49–60, Baltimore, ML, 2005.

[26] N. Li, T. Li, and S. Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and ℓ-diversity. In *Proc. of the 21st IEEE International Conference on Data Engineering (ICDE)*, Istanbul, Turkey, April 2007.

[27] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. ℓ-diversity: Privacy beyond k-anonymity. In *Proc. of the 22nd IEEE International Conference on Data Engineering (ICDE)*, 2006.

[28] B. Malin and E. Airoldi. The effects of location access behavior on re-identification risk in a distributed environment. In *Proc. of the 6th Workshop on Privacy Enhancing Technologies (PET)*, pages 413–429, 2006.

[29] A. Meyerson and R. Williams. On the complexity of optimal k-anonymity. In *Proc. of the 23rd ACM PODS*, pages 223–228, Paris, France, 2004.

[30] N. Mohammed, B. C. M. Fung, P. C. K. Hung, and C. K. Lee. Anonymizing healthcare data: A case study on the blood transfusion service. In *Proc. of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 1285–1294, Paris, France, June 2009. ACM Press.

[31] N. Mohammed, B. C. M. Fung, K. Wang, and P. C. K. Hung. Privacy-preserving data mashup. In *Proc. of the 12th International Conference on Extending Database Technology (EDBT)*, Saint-Petersburg, Russia, March 2009. ACM Press.

[32] R. Motwani and Y. Xu. Efficient algorithms for masking and finding quasi-identifiers. In *Proc. of VDLB '07*, September 2007.

[33] M. E. Nergiz and C. Clifton. Thoughts on k-anonymization. volume 63, pages 622–645, Amsterdam, 12 2007. Elsevier Science.

[34] M. E. Nergiz, C. Clifton, and A. E. Nergiz. Multirelational k-anonymity. In *Proceedings of the 23rd IEEE International Conference on Data Engineering (ICDE 2007)*, Istanbul, Turkey, 2007.

[35] A. Øhrn and L. Ohno-Machado. Using boolean reasoning to anonymize databases. *Artificial Intelligence in Medicine*, 15(3):235–254, 1999.

[36] S. Papadimitriou, F. Li, G. Kollios, and P. S. Yu. Time series compressibility and privacy. In *VLDB '07: Proceedings of the 33rd international conference on Very large data bases*, pages 459–470. VLDB Endowment, 2007.

[37] P. Samarati. Protecting respondents' identities in microdata release. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 13(6):1010–1027, 2001.

[38] P. Samarati and L. Sweeney. Generalizing data to provide anonymity when disclosing information. In *Proc. of the 17th ACM PODS*, page 188, June 1998.

[39] P. Samarati and L. Sweeney. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. Technical report, SRI International, March 1998. http://citeseer.ist.psu.edu/samarati98protecting.html.

[40] S. E. Sarma, S. A. Weis, and D. W. Engels. RFID systems and security and privacy implications. In *Proc. of the 4th International Workshop of Cryptographic Hardware and Embedded Systems (CHES)*, pages 1–19, San Diego, 2003.

[41] A. Stone. Natural-language processing for intrusion detection. *Computer*, 40(12):103–105, Dec. 2007.

[42] L. Sweeney. Datafly: A system for providing anonymity in medical data. In *Proc. of the IFIP TC11 WG11.3 11th International Conference on Database Securty XI: Status and Prospects*, pages 356–381, August 1998.

[43] L. Sweeney. k-Anonymity: a model for protecting privacy. In *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, volume 10, pages 557–570, 2002.

[44] M. Terrovitis and N. Mamoulis. Privacy preservation in the publication of trajectories. In *Proc. of the 9th International Conference on Mobile Data Management (MDM)*, pages 65–72, April 2008.

[45] M. Terrovitis, N. Mamoulis, and P. Kalnis. Privacy-preserving anonymization of set-valued data. *Proc. VLDB Endow.*, 1(1):115–125, 2008.

[46] The RFID Knowledgebase. Oyster transport for london tfl, card uk, January 2007. http://rfid.idtechex.com/knowledgebase/en/casestudy.asp?freefromsection =122.

[47] K. Wang and B. C. M. Fung. Anonymizing sequential releases. In *Proc. of the 12th ACM SIGKDD*, Philadelphia, PA, August 2006.

[48] K. Wang, B. C. M. Fung, and P. S. Yu. Handicapping attacker's confidence: An alternative to k-anonymization. *Knowledge and Information Systems (KAIS)*, 11(3):345–368, April 2007.

[49] K. Wang, P. S. Yu, and S. Chakraborty. Bottom-up generalization: a data mining solution to privacy protection. In *Proc. of the 4th IEEE International Conference on Data Mining (ICDM)*, November 2004.

[50] S. L. Warner. Randomized response: A survey technique for eliminating evasive answer bias. volume 60, pages 63–69, 1965.

[51] S.M. White. Social engineering. In *Engineering of Computer-Based Systems, 2003. Proceedings. 10th IEEE International Conference and Workshop on the*, pages 261–267, April 2003.

[52] Y. Xu, B. C. M. Fung, K. Wang, A. W. C. Fu, and J. Pei. Publishing sensitive transactions for itemset utility. In *Proc. of the 8th IEEE International Conference on Data Mining (ICDM)*, December 2008.

[53] Y. Xu, K. Wang, A. W. C. Fu, and P. S. Yu. Anonymizing transaction databases for publication. In *Proc. of the 14th ACM SIGKDD*, August 2008.

[54] Z. Yang, S. Zhong, and R. N. Wright. Anonymity-preserving data collection. In *Proc. of the 11th ACM SIGKDD*, pages 334–343, 2005.

[55] M. Yarovoy, F. Bonchi, L. V. S. Lakshmanan, and W. H. Wang. Anonymizing Moving Objects: How to Hide a MOB in a Crowd? In *EDBT '09: Proceedings of the 12th International Conference on Extending Database Technology*, pages 72–83, New York, NY, USA, 2009. ACM.