Cor-Split: Defending Privacy in Data Re-Publication from Historical Correlations and Compromised Tuples

Daniele Riboni Claudio Bettini

D.I.Co., Università di Milano, Italy
{riboni,bettini}@dico.unimi.it

Abstract. Several approaches have been proposed for privacy preserving data publication. In this paper we consider the important case in which a certain view over a dynamic dataset has to be released a number of times during its history. The insufficiency of techniques used for one-shot publication in the case of subsequent releases has been previously recognized, and some new approaches have been proposed. Our research shows that relevant privacy threats, not recognized by previous proposals, can occur in practice. In particular, we show the cascading effects that a single (or a few) compromised tuples can have in data re-publication when coupled with the ability of an adversary to recognize historical correlations among released tuples. A theoretical study of the threats leads us to a defense algorithm, implemented as a significant extension of the *m-invariance* technique. Extensive experiments using publicly available datasets show that the proposed technique preserves the utility of published data and effectively protects from the identified privacy threats.

1 Introduction

There are many data repositories that store time-dependent data and that require recurrent release of recently acquired data to third parties. Many papers have addressed the problem of anonymizing datasets for one-time publication ([1–4] among many others). The main defense technique consists in providing anonymity by generalizing the values of quasi-identifier (QI) attributes, so that each released tuple belongs to a group (called QI-group) having the same value for the QI attributes. This intuitively guarantees that within a group the tuple respondents cannot be distinguished. The cardinality of the group as well as the distribution of sensitive attribute values in each group are relevant parameters for the achieved anonymity. However, less attention has been given to privacy threats that can occur upon re-publication of the same database after updates have been performed. Indeed, it has been recognized that additional privacy issues arise if the adversary obtains a history of tables anonymized as described above. For example, by understanding that tuples t_1 and t_2 in different releases refer to the same (anonymous) individual, the intersection of the candidate respondents for t_1 and t_2 can lead to a privacy violation.

Table 1. Original and generalized microdata at the first and second release

(a) Original microdata at time τ_1

Name	Age	Gender	Zip	Disease
Alice	33	Female	12000	cancer
Betty	31	Female	11000	bronchitis
Carl	35	Male	12000	AIDS
Doris	40	Female	13000	cancer
Erica	41	Female	14000	AIDS
Fiona	37	Female	13000	bronchitis

(b) Generalized microdata: 1st release

QI-group	Age	Gender	Zip	Disease
1	[31,35]	×	[11k,12k]	cancer
1	[31,35]	×	[11k,12k]	bronchitis
1	[31,35]	×	[11k,12k]	AIDS
2	[37,41]	Female	[13k,14k]	cancer
2	[37,41]	Female	[13k,14k]	AIDS
2	[37,41]	Female	[13k,14k]	bronchitis

(c) Original microdata at time τ_2

Name	Age	Gender	Zip	Disease
Carl	35	Male	12000	AIDS
Doris	40	Female	13000	cancer
Fiona	37	Female	13000	bronchitis
Erica	41	Female	14000	AIDS
Grace	42	Female	13000	bronchitis
Hanna	42	Female	13000	cancer

(d) Generalized microdata: 2nd release

QI-group	Age	Gender	Zip	Disease
3	[35,40]	×	[12k,13k]	AIDS
3	[35,40]	×	[12k,13k]	cancer
3	[35,40]	×	[12k,13k]	bronchitis
4	[41,42]	Female	[13k,14k]	AIDS
4	[41,42]	Female	[13k,14k]	bronchitis
4	[41,42]	Female	[13k,14k]	cancer

As a motivating example, we consider data about patients and their cause of hospitalization (called *disease* for simplicity in the rest of this paper) frequently released by a hospital to certain institutions for data analysis. Each released table contains one tuple for each patient hospitalized during the last L months. In this scenario, certain tuples may be present in multiple releases, some tuples that never appeared before can appear in new releases, and other tuples may disappear in subsequent releases. Hence, we consider updates involving both insertion and removal of tuples. For the sake of simplicity, we assume that when a tuple appears in multiple releases, the corresponding private value remains the same.[1] We consider the realistic case that some tuples may be compromised; for example, the actual disease of some patient may be known to the adversary (note that, at least, every patient is aware of her own disease). The following examples illustrate privacy threats that can occur in such a scenario.

Example 1. Consider the original microdata at time τ_1 and τ_2, shown in Tables 1(a) and 1(c), and the generalized microdata in Tables 1(b) and 1(d). Note that each generalization guarantees anonymity according to state-of-the-art techniques (k-anonymity [1] with $k = 3$, l-diversity [3] with $l = 3$, and t-closeness [4] with $t = 0$); moreover the generalization at time τ_2 also satisfies m-invariance [5] with $m = 3$, a technique specifically designed for data re-publication.

The respondents of tuples belonging to QI-group 3 in Table 1(d) are Doris, Fiona and Carl, and the set of their candidate private values is {cancer, bronchitis, AIDS}. Suppose that the tuple about Carl has been compromised, hence revealing to the adversary that Carl has AIDS. This leads the adversary to derive by exclusion that either Doris was hospitalized for cancer and Fiona for bronchitis, or vice versa.

[1] The assumption, also made in [5], can be easily relaxed by associating an id to each tuple as often happens for real data. Referring to our example, we can use a different id for a new hospitalization of the same patient (possibly for a different disease) to enable the same kind of attack.

Now, note that two new tuples have been inserted at τ_2, namely those regarding Grace and Hanna, while the tuples regarding Alice and Betty have been removed. In order to understand what we mean by historical correlation, consider the histories of QI-groups of Doris' and Fiona's tuples, i.e., QI-group 2 in Table 1(b) and QI-group 3 in Table 1(d). The set of respondents of the first group is {Doris, Erica,Fiona} while for the second is {Carl, Doris, Fiona}, and the corresponding set of private values is {cancer, bronchitis, AIDS} for both. By assumption, the presence in both releases of tuples for Doris and Fiona imply that each of them preserved her private value across releases. Hence, the possible private values for Doris and Fiona, considering that Carl's tuple has been compromised, were {cancer, bronchitis} even at τ_1. Since at τ_1 the possible private values for the QI-group 2 (including a tuple whose respondent must be Erica) were {cancer, bronchitis, AIDS}, the adversary can conclude that Erica was hospitalized for AIDS.

One of the first attempts to address privacy issues in data re-publication can be found in [6]. That work proposes a technique to preserve a weak form of l-diversity when multiple versions of the same table are released over time and the table is updated by insertions only. One shortcoming of that technique is that microdata publishing is postponed until the conditions for guaranteeing the required level of l-diversity are met. A similar scenario is addressed in [7] and in [8] in the case in which tuples are released together with their unique identifier or not, respectively. Even if the solutions proposed in those works do not require delaying data publication, they are restricted to the case in which tables are updated with insertions only, and cannot be applied when tuples are removed. The first work to address privacy-preserving data re-publication when both insertions and deletions are allowed is [5], in which the m-invariance property is proposed. That property ensures that *i)* all the QI-groups in which a tuple appears have the same set of private values (the cardinality of such set must be greater than or equal to m), and *ii)* the set of private values of QI-groups maximize the level of diversity (i.e., each QI-group does not contain tuples having the same private value). However, m-invariance is prone to privacy threats (as the ones exemplified above) that were not identified before, for which we propose both a theoretical study and a defense algorithm.

In this paper we consider the same scenario considered in [5], admitting both insertion and removal of tuples, but also considering the case that some tuples may be compromised. We show that, even when a very small percentage of tuples is compromised, the correlation that can be identified between tuples in different releases can lead to serious privacy leaks.

The main contributions of our work are the following: a) We perform a probabilistic analysis showing that in realistic cases the application of state-of-the-art techniques for privacy preservation in data re-publication can fail to protect the privacy of individuals. b) We propose the *Cor-Split* algorithm as a defense technique, inspired by m-invariance, against attacks exploiting compromised tuples and historical correlations. The algorithm is proved to correctly provide protection. c) We show experimental results on public data directly comparing Cor-Split with m-invariance: From the experiments we can conclude that the

extra protection offered by our algorithm has negligible costs over previously known techniques.

The rest of this paper is organized as follows. Section 2 formalizes the privacy threat we are considering. Section 3 reports a probabilistic analysis showing the actual risk of a privacy breach according to the value of some parameters. Section 4 illustrates the Cor-Split defense algorithm and its formal properties. Section 5 shows experimental results, and Section 6 concludes the paper.

2 Model of privacy threats

In this section we formally model the notions of privacy breach, and the functions that may be used by an adversary to restrict the set of private values associated to each candidate tuple respondent. We call such functions *private value restriction functions*.

2.1 Preliminary definitions

In this paper we denote by T_j an original table at time τ_j, and by T_j^* the generalization of T_j released by the data publisher; we denote by $\mathcal{H}_{1,j}^* = \langle T_1^*, T_2^*, \ldots, T_j^* \rangle$ a *history* of released generalized tables. We say that a tuple t has lifespan $\mathcal{L}$ if the generalization t^* of t appeared in each table T_i^* with $i \in \mathcal{L}$; we denote by $t.r$ the respondent of t.

Tables are generalized by a *generalization function* $G : \mathcal{T} \times \widetilde{\mathcal{H}} \times \mathcal{R} \times \Theta \rightarrow \mathcal{T}^*$, where $\mathcal{T}$ is the set of possible original tables, $\widetilde{\mathcal{H}}$ is the set of possible histories of original microdata tables, $\mathcal{R}$ is the collection of possible sets of tuples respondents, Θ is the set of functions that map each respondent r to her set of possible private values S_r, and $\mathcal{T}^*$ is the set of possible generalized tables. We denote by $S_{r,j}$ the set of possible private values of respondent r at time τ_j.

We assume that the schema of tables in $\mathcal{H}_{1,j}^*$ remains unchanged throughout the release history, and we classify the table columns into a set A^{qi} of quasi-identifier attributes (for the sake of simplicity we assume that categorical values are transformed in numeric ones), and into a single private attribute A^s having domain S; $t[A]$ is the projection of t onto A. Columns that do not act as either quasi-identifier or private value are irrelevant with respect to privacy preservation and therefore they are ignored in the rest of the paper. Tuples in T^* are partitioned into *QI-groups*; i.e., sets of tuples having the same values for their quasi-identifier attributes. We denote by $Q.R$ the set of respondents of tuples belonging to a QI-group Q. The *signature* $Q.sig$ of Q is the set of private values of tuples belonging to Q.

We assume that the background knowledge available to an adversary is composed of the generalization function G, and the set R of respondents, as well as their QI values and their sets of possible private values. Moreover, as usual in related work, we assume that, given a QI-group Q, an adversary may get to know the exact set of respondents of tuples in Q. Note that the latter is a conservative assumption, since in general the generalized QI values of tuples in Q could match more users than the actual respondents of Q's tuples.

Definition 1 (privacy breach). *A* privacy breach *occurs when an adversary knows the* sensitive association *between a user and one or more of her private values.*

2.2 Threats deriving from compromised tuples

As shown by Example 1, the presence of a compromised tuple in a QI-group Q can be used by an adversary to restrict the set of possible private values of the respondents of other tuples in Q. This kind of adversarial inference can be modeled as a *compromised tuples-based private value restriction (ct-pvr) function.*

Definition 2 (ct-pvr function). *Given a history of released tables $\mathcal{H}^*_{1,j}$, the respondent r of a tuple t^* in $\mathcal{H}^*_{1,j}$, r having prior set of possible private values S_r, the set $\mathcal{H}^*_{1,j}(Q,t)$ of QI-groups published in $\mathcal{H}^*_{1,j}$ and containing a generalization of t, and a set $C_{\mathcal{H}^*_{1,j}}$ of compromised tuples published in $\mathcal{H}^*_{1,j}$, a* ct-pvr function *is a function ct-pvr$: R \times 2^S \times 2^{2^{T^*}} \times 2^{T^*} \rightarrow 2^S$ such that:*

$$\begin{aligned} &\textit{ct-pvr}(r, S_r, \mathcal{H}^*_{1,j}, C_{\mathcal{H}^*_{1,j}}) = \\ &= S_r \setminus \{a \in S \mid \exists\, Q \in \mathcal{H}^*_{1,j}(Q,t), \forall u^* \in Q, u^*[A^s] = a \Rightarrow u^* \in C\}. \end{aligned}$$

Note that in order to discard a value for a respondent of a tuple belonging to Q, every tuple in Q having that value should be associated by the adversary to a different respondent. The example below shows how the case of a compromised tuple reported in Example 1 applies to Definition 2.

Example 2. Referring to Example 1, consider the history of released tables $\mathcal{H}^*_{1,2}$ corresponding to Tables 1(b) and 1(d), and the respondent r=Doris having set of possible private values S_r = {cancer, AIDS, bronchitis}; the set $C_{\mathcal{H}^*_{1,2}}$ of compromised tuples known by the adversary includes the first tuple in Table 1(d). Hence, by applying the ct-pvr function considering the QI-group 1, the adversary can discard the value a=*AIDS* from S_r, since *AIDS* is known to be the private value of Carl's tuple. Consequently, after the application of the ct-pvr function the set of Doris' possible private values contain only *cancer* and *bronchitis*. The same reasoning can be applied with r=Fiona.

2.3 Threats deriving from re-published microdata

As shown in Example 1, re-published microdata is prone to a specific class of adversarial inference. In order to model this kind of privacy threats we introduce the notion of *historical correlation.*

Definition 3 (historical correlation). *Given a history of released tables $\mathcal{H}^*_{1,j}$, and two QI-groups $Q_1 \subseteq T^*_i \in \mathcal{H}^*_{1,j}$ and $Q_2 \subseteq T^*_l \in \mathcal{H}^*_{1,j}$ ($i \neq l$), a historical correlation between two sets of respondents R_1 and R_2 can be recognized if there exist two QI-groups Q_1 and Q_2 ($Q_1 \neq Q_2$ and $Q_1.S = Q_2.S$, where $Q_1.S$ and $Q_2.S$ are the multisets composed of private values of tuples in Q_1 and in Q_2, respectively) such that all the following conditions hold:*

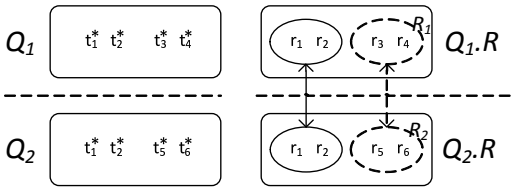


Fig. 1. Derivation of a historical correlation

c1) $Q_1.R \supset R_1$
c2) $Q_2.R \supset R_2$
c3) $Q_1.R \setminus R_1 = Q_2.R \setminus R_2$

where $Q_1.R$ and $Q_2.R$ are the sets of respondents of tuples belonging to Q_1 and Q_2, respectively.

Theorem 1 (historical correlations). *If two sets of respondents R_1 and R_2 are in a historical correlation, then the multiset composed of the private values of respondents in R_1 is equal to the multiset composed of the private values of respondents in R_2.*

As it was shown in Example 1, historical correlations can be used to narrow the set of candidate private values of a respondent, possibly leading to the exact derivation of her private value. Such correlations are called *historical* since they rely on the presence of the same sets of tuples in multiple views belonging to a history of releases. For instance, in Example 1 an adversary was able to find a historical correlation between R_1={Carl} and R_2={Erica} by observing QI-group 2 and QI-group 3, released at time τ_1 and τ_2, respectively:

c1) (QI-group 3).$R \supset$ {Carl}
c2) (QI-group 2).$R \supset$ {Erica}
c3) (QI-group 3).$R \setminus$ {Carl} = (QI-group 2).$R \setminus$ {Erica}

Similarly, a historical correlation between {Alice, Betty} and {Doris, Fiona} could be discovered observing QI-group 1 and QI-group 3.

Example 3. Consider Figure 1, which depicts two QI-groups $Q_1 = \{t_1^*, t_2^*, t_3^*, t_4^*\}$ and $Q_2 = \{t_1^*, t_2^*, t_5^*, t_6^*\}$, and the sets of respondent of Q_1 and Q_2, which are $Q_1.R = \{r_1, r_2, r_3, r_4\}$ and $Q_2.R = \{r_1, r_2, r_5, r_6\}$, respectively. In this situation a historical correlation between $R_1 = \{r_3, r_4\}$ and $R_2 = \{r_5, r_6\}$ can be recognized. Indeed, the set of respondents $\{r_1, r_2\}$ (enclosed in a solid ellipse) appears in both $Q_1.R$ and $Q_2.R$ and, since the private value of each tuple cannot change, the private values of r_1 and r_2 are the same in Q_1 and in Q_2. As a consequence, the set of private values of $\{r_3, r_4\}$ is the same as that of $\{r_5, r_6\}$ (the sets of tuples related by a historical correlation are enclosed in a dashed ellipse).

The adversarial inference that exploits historical correlations to the aim of restricting the set of possible private values of a tuple respondent can be modeled according to the following *historical correlation-based private value restriction (hc-pvr) function.*

Definition 4 (hc-pvr function). *Given a history of released tables $\mathcal{H}^*_{1,j}$, the respondent r of a tuple t^* in $\mathcal{H}^*_{1,j}$, r having initial set of possible private values S_r, and the set $\mathcal{R}(\mathcal{H}^*_{1,j}, r)$ of sets of respondents that are linked to r by a historical correlation in $\mathcal{H}^*_{1,j}$, a* hc-pvr function *is a function hc-pvr* : $R \times 2^S \times 2^{2^R} \rightarrow 2^S$ *such that:*

$$\textit{hc-pvr}(r, S_r, \mathcal{R}(\mathcal{H}^*_{1,j}, r)) = S_r \setminus \{a \in S \mid \exists\, R \in \mathcal{R}(\mathcal{H}^*_{1,j}, r), \forall r' \in R,\, a \notin S_{r',j}\}.$$

The following example shows how the adversarial inference presented in Example 1 applies to Definition 4.

Example 4. Referring to Example 1, consider the history of released tables $\mathcal{H}^*_{1,2}$ corresponding to Tables 1(b) and 1(d), and the respondent r=Erica having set of possible private values $S_r = \{\text{cancer, AIDS, bronchitis}\}$; $\mathcal{R}(\mathcal{H}^*_{1,2}, \textit{Erica})$ includes the set {Carl} (i.e., a historical correlation relating Erica and Carl was discovered). The set of Carl's possible private values includes neither *cancer* nor *bronchitis* (i.e., the adversary knows that the private value of his tuple is *AIDS*). Hence,

$$\begin{aligned}&\text{hc-pvr}(\textit{Erica}, S_{\textit{Erica}}, \mathcal{R}(\mathcal{H}^*_{1,2}, \textit{Erica})) = \\ &\quad = \{\text{cancer, AIDS, bronchitis}\} \setminus \{\text{cancer, bronchitis}\} = \{\text{AIDS}\}.\end{aligned}$$

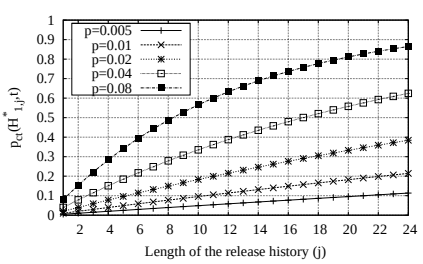
Then, after the second release the adversary derives that the set of possible private values of Erica is {AIDS}. As a consequence, the sensitive association between Erica and AIDS is discovered.

3 Probabilistic analysis

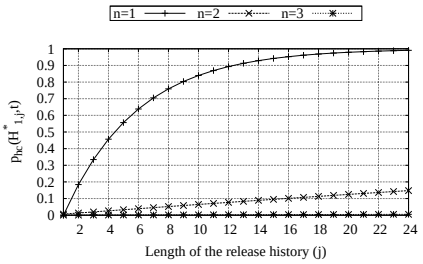
The actual risk of a privacy breach due to the threats we are considering depends on several parameters. In this section we perform a probabilistic analysis of this risk, assuming that each tuple has probability p of being compromised. Without loss of generality, we also assume that released microdata satisfy the m-invariance principle, since – as it will be shown later in this section – this is a worst case for our probabilistic analysis. As a consequence, we also assume that the set S of private values includes at least as many values as the enforced level m of m-invariance. We also assume that each tuple is released at most L times; i.e., for each tuple t, L is an upper limit for the cardinality of its lifespan $\mathcal{L}$. The probability of privacy breach is measured, depending on the parameters p, m, L as well as others, as the result of the application of the private value restriction functions described in Sections 2.2 and 2.3.

3.1 Probability of excluding private values due to compromised tuples

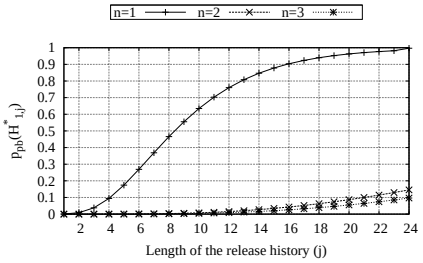
As illustrated in Section 2.2, given a QI-group Q, if all the tuples in Q having private value a are compromised, then an adversary can conclude that no other respondent of tuples in Q has private value a. Of course, the probability that such an event occurs decreases the higher is the number of occurrences of tuples



(a) $p_{ct}(\mathcal{H}^*_{1,j}, t)$



(b) $p_{hc}(\mathcal{H}^*_{1,j}, t)$ with $p = 0.04$ and $m = 6$



(c) $p_{pb}(\mathcal{H}^*_{1,j}, t)$ with $p = 0.04$ and $m = 6$

Fig. 2. Probabilistic analysis

in Q having private value a. Hence, in order to analyze the worst case we assume that each private value is owned by at most one tuple in a single QI-group. This property is called m-uniqueness in [5], and it is guaranteed by the enforcement of the m-invariance principle. The corresponding privacy threat is quantified by the following lemma.

Lemma 1. *Given a history of released tables $\mathcal{H}^*_{1,j}$ satisfying the m-invariance principle, a set $C_{\mathcal{H}^*_{1,j}}$ of compromised tuples belonging to $\mathcal{H}^*_{1,j}$, and a tuple t having respondent r and lifespan $\mathcal{L}_{\mathcal{H}^*_{1,j}}$ in $\mathcal{H}^*_{1,j}$ with $\max\{|\mathcal{L}_{\mathcal{H}^*_{1,j}}|\} \leq j$, the probability that a private value is discarded from the set S_r of possible private values of r through the application of function ct-pvr$(r, S_r, \mathcal{H}^*_{1,j}, C_{\mathcal{H}^*_{1,j}})$ is:*

$$p_{ct}(\mathcal{H}^*_{1,j}, t) = 1 - (1-p)^{|\mathcal{L}_{\mathcal{H}^*_{1,j}}|},$$

where p is the probability of a generic tuple to be compromised.

The plot shown in Figure 2(a) shows the value of $p_{ct}(\mathcal{H}^*_{1,j}, t)$ with respect to the number of releases containing the tuple (here we assume that the lifespan of t covers the entire history) for different values of p. As expected, $p_{ct}(\mathcal{H}^*_{1,j}, t)$ grows with the length of the release history and with the value of p.

3.2 Probability of excluding private values due to historical correlation

The probability that a private value is excluded from the set of candidate private values due to historical correlations is given by the following lemma.

Lemma 2. *Given a history of released tables $\mathcal{H}^*_{1,j}$ satisfying the m-invariance property, a tuple t having respondent r and lifespan $\mathcal{L}_{\mathcal{H}^*_{1,j}}$ in $\mathcal{H}^*_{1,j}$ with $\max\{|\mathcal{L}_{\mathcal{H}^*_{1,j}}|\} \leq j$, and the set $\mathcal{R}(\mathcal{H}^*_{1,j}, r)$ of sets of respondents that are linked to r by a historical correlation in $\mathcal{H}^*_{1,j}$, the probability that a private value a is discarded from the set S_r of possible private values of r through the application of function hc-pvr$(r, S_r, \mathcal{R}(\mathcal{H}^*_{1,j}, r))$ is:*

$$p_{hc}(\mathcal{H}^*_{1,j}, t) = 1 - \left(1 - \left(p - \frac{p}{m}\right)^n\right)^{|\mathcal{L}_{\mathcal{H}^*_{1,j}}| \cdot \lfloor \frac{m}{n} \rfloor},$$

where $n = \min_{R \in \mathcal{R}(\mathcal{H}^*_{1,j}, r)} \{|R|\}$ is the minimum cardinality of sets of respondents in $\mathcal{R}(\mathcal{H}^*_{1,j}, r)$, and p is the probability of a generic tuple to be compromised.

The plot shown in Figure 2(b) shows the value of $p_{hc}(\mathcal{H}^*_{1,j}, t)$ with respect to the length j of the release history (assuming the lifespan of t covers the entire history) and to the minimum cardinality n of sets of respondents involved in historical correlations. The released tables are assumed to satisfy the m-invariance property with $m = 6$, and the probability of a tuple to be compromised is set to $p = 0.04$. As expected, $p_{hc}(\mathcal{H}^*_{1,j}, t)$ grows with the length of the release history of the tuple and it is higher for smaller values of n.

3.3 Probability of privacy breach due to combined threats

After having separately considered the threats deriving from functions ct-pvr and hc-pvr we can quantify the probability of privacy breach deriving from the application of both functions.

Theorem 2 (probability of privacy breach). *Given a history of released tables $\mathcal{H}^*_{1,j}$ satisfying the m-invariance principle, a tuple t having respondent r and lifespan $\mathcal{L}_{\mathcal{H}^*_{1,j}}$ in $\mathcal{H}^*_{1,j}$ with $\max\{|\mathcal{L}_{\mathcal{H}^*_{1,j}}|\} \leq j$, and the set $\mathcal{R}(\mathcal{H}^*_{1,j}, r)$ of sets of respondents that are linked to r by a historical correlation in $\mathcal{H}^*_{1,j}$, if $n = \min_{R \in \mathcal{R}(\mathcal{H}^*_{1,j}, r)} \{|R|\}$ is the minimum cardinality of sets of respondents in $\mathcal{R}(\mathcal{H}^*_{1,j}, r)$, and p is the probability of a generic tuple to be compromised, then the probability that a privacy breach is determined by functions ct-pvr and hc-pvr at time j is:*

$$p_{pb}(\mathcal{H}^*_{1,j}, t) = \left(1 - (1-p)^{|\mathcal{L}_{\mathcal{H}^*_{1,j}}|} \cdot \left(1 - \left(p - \frac{p}{m}\right)^n\right)^{|\mathcal{L}_{\mathcal{H}^*_{1,j}}| \cdot \lfloor \frac{m}{n} \rfloor}\right)^{m-1}.$$

The plot shown in Figure 2(c) shows the value of $p_{pb}(\mathcal{H}^*_{1,j}, t)$ with respect to the length j of the release history (assuming the lifespan of t covers the entire history) and to the minimum cardinality n of sets of respondents involved in historical correlations. The released tables are assumed to satisfy the m-invariance property with $m = 6$, and the probability of a tuple to be compromised is set to $p = 0.04$. It can be observed that the probability of privacy breach is very high with $n = 1$. The value of p_{pb} is lower than 0.15 with $n = 2$; with $n = 3$ the probability of privacy breach is lower than 0.1.

4 Defense

4.1 Safety against private value restriction functions

In order to defend re-published microdata against private value restriction functions based on compromised tuples and historical correlations, our technique consists in enforcing a generalization principle – which we name *(m, n)-historical safety* – with parameters that guarantee that the probability of privacy breach is below a certain threshold h. Before defining (m, n)-historical safety it is necessary to introduce a novel principle, which we name *weak m-invariance*. As it will we shown in Section 4.2, this principle can be applied to avoid the disclosure of historical correlations while minimizing the number of counterfeits.

Definition 5 (weak m-uniqueness). *A generalized table T^*_j satisfies* weak m-uniqueness *if each QI-group $Q \subseteq T^*_j$ contains at least m tuples with different private values, and the number of occurrences in Q of tuples with a given private value is the same for every private value belonging to the signature of Q.*

Definition 6 (weak m-invariance). *A history of released tables $\mathcal{H}^*_{1,j}$ satisfies* weak m-invariance *if:*

- $\forall i \in [1, j]$, T^*_i *satisfies weak m-uniqueness, and*
- $\forall i, i' \in [1, j]$, $t \in T_i, t' \in T_{i'}$, *if* $t^* \in Q_i$ *and* $t'^* \in Q_{i'}$, *then* $Q_i.sig = Q_{i'}.sig$.

Weak m-invariance is a weaker version of the m-invariance principle. Indeed, while m-invariance requires that all the tuples in a QI-group have different private values, according to weak m-invariance QI-groups can contain tuples with duplicate private values, provided that their multiplicity is the same. Hence, it is easy to verify that m-invariance is a particular case of weak m-invariance in which the multiplicity of tuples having a given private value in a given QI-group is always 1. With respect to privacy preservation, it can be observed that weak m-invariance provides the same level of diversity as the one provided by m-invariance. On the other hand, the obvious shortcoming of having multiplicities of private values greater than 1 in a QI-group is that in most cases the degree of generalization of QI values of that QI-group would grow with the multiplicity. Hence, as it will be shown in Section 4.2, an objective of our devised generalization algorithm is to minimize the multiplicity of private values in QI-groups.

Definition 7 (hc-safety). *Given a history of released tables $\mathcal{H}^*_{1,j}$ and a QI-group $Q \subseteq T^*_i \in \mathcal{H}^*_{1,j}$, Q is* hc-safe with degree n *if either:* i) *no set of respondents is related with the respondents of tuples in Q by a historical correlation in $\mathcal{H}^*_{1,j}$, or* ii) *the cardinality of each set of respondents that is related with the respondents of tuples in Q by a historical correlation in $\mathcal{H}^*_{1,j}$ is greater than or equal to n.*

Definition 8 ((m,n)-historical safety). *Given $m, n \in \mathbb{N}$, $n \leq m$, a generalization function G is* (m,n)-historically safe *if, for each table T_{j+1}, for each history of released tables $\mathcal{H}^*_{1,j}$ satisfying weak m-invariance, and*

$$T^*_{j+1} = G(T_{j+1}, \mathcal{H}^*_{1,j}, R, \vartheta)$$

(R is the of tuples respondents, and ϑ is the function that maps each respondent in R into her set of possible private values), the following conditions hold:

i) *$\langle \mathcal{H}^*_{1,j}, T^*_{j+1} \rangle$ satisfies weak m-invariance;*
ii) *each QI-group $Q \subseteq T^*_{j+1}$ is hc-safe with degree n with respect to $\langle \mathcal{H}^*_{1,j}, T^*_{j+1} \rangle$.*

The above definition states that, in order to be (m,n)-historically safe, a generalization function must *i)* preserve weak m-invariance and *ii)* generate QI-groups such that the cardinality of sets involved in historical correlations that can be derived from them is greater than or equal to n. Condition *i)* is imposed to protect against the attacks identified in [5]; condition *ii)* is imposed to protect against historical correlations.

In order to guarantee that the probability of privacy breach for a tuple t is below a certain threshold h it is necessary to have an estimate of the probability p of released tuples to be compromised, and to determine the maximum cardinality L of its lifespan (i.e., the maximum number of times that t can be republished). Note that L is an upper bound for the cardinality of $\mathcal{L}_{\mathcal{H}^*_{1,j}}$ shown in the definitions of private value restriction functions (see Section 3). In general, the values of p and L depend on the domain of the data. For instance, a hospital releasing microdata about patients and diseases may estimate that an adversary may get to know the sensitive association about no more than 4% of its patients (hence, $p = 0.04$), and it can decide to republish each tuple at most 24 times (hence, $L = 24$). Once values for p and L have been determined it is possible to express the concept of safety of a generalization function against a threshold h.

Definition 9 (pvr-safe generalization function). *A generalization function G is* pvr-safe *with threshold $h \in (0,1]$ if, for any history $\mathcal{H}^*_{1,j}$ of tables generalized by G, $p_{pb}(\langle T^*_1, \ldots, T^*_i \rangle, t) < h$ for each $i \in [1,j]$ and for each tuple $t \in T_i$.*

4.2 The *Cor-Split* algorithm

Given parameters p and L, the chosen level m of weak m-invariance to be enforced, and the required threshold h, the goal of the algorithm proposed in this paper is to enforce (m,n)-historical safety with the smallest possible value of

Input: Parameters p, L, m, h.
Output: Parameter n.

1: $f(p, L, m, n) = \left(1 - (1-p)^L \cdot \left(1 - \left(p - \frac{p}{m}\right)^n\right)^{L \cdot \lfloor \frac{m}{n} \rfloor}\right)^{m-1}$
2: **if** $\nexists\ n' \in [1, m] \mid f(p, L, m, n') < h$ **then**
3: $\quad \overline{n} := -1$
4: **else**
5: $\quad \overline{n} := \min(n' \in \mathbb{N}^+ \mid f(p, L, m, n') < h)$
6: **end if**
7: **return** $\overline{n}$

Fig. 3. The *n-Choose* algorithm for determining the value of n

Input: T_{j+1} is the microdata table at time τ_{j+1}; $\mathcal{H}^*_{1,j}$ is the history of released tables; $\mathcal{H}$ is the history of original tables; R is the set of tuples respondents; ϑ is the function that maps each respondent in R into her set of possible private values; $m, n \in \mathbb{N}$ are the parameters for historical safety; A^{qi} is the set of QI attributes.
Output: the generalized table T^*_{j+1}.

1: $T^*_{j+1} := \emptyset$
2: $S_{\text{-}} := \{t \in T_{j+1} \mid \forall\ T \in \mathcal{H},\ t \notin T\}$
3: $S_{\cap} := T_{j+1} \setminus S_{\text{-}}$
4: $\mathcal{B} := \text{Division}(S_{\cap})$
5: **for all** buckets $B \in \mathcal{B}$ **do**
6: $\quad \text{Balancing}(B, S_{\text{-}})$
7: **end for**
8: $\mathcal{B}' := \text{Assignment}(\mathcal{B}, S_{\text{-}}, m, \vartheta)$
9: **for all** buckets $B \in \mathcal{B}'$ **do**
10: $\quad T^*_{j+1} := \text{Cor-Partition}(T^*_{j+1}, B, A^{qi}, n)$
11: **end for**
12: **return** T^*_{j+1}

Fig. 4. The *Cor-Split* algorithm

n that guarantees that the probability of privacy breach is below h. Since we assume that those parameters do not change during the release history, the parameter n is chosen before the generalization of the first table, and it remains unchanged throughout the release history. The algorithm for choosing n is shown in Figure 3. Note that for certain values of p, L, m, and for a required threshold h, the probability of privacy breach could be higher than h for every possible value of n. In this case, microdata would not be released unless the value of parameters L or m are changed. In the other case, microdata are generalized using the value of n determined by the algorithm in Figure 3. Our devised generalization algorithm, shown in Figure 4, is a significant modification of the algorithm for m-invariant generalization proposed in [5]. Note that the algorithm in [5], though enforcing weak m-invariance, does not provide guarantees about the cardinality of sets of respondents involved in historical correlations. Moreover, our empirical study (reported in Section 5) shows that QI-groups generated by that algorithm

may allow an adversary to derive several historical correlations between small sets of respondents, determining severe privacy threats.

Overview of our generalization algorithm. Given the original table T_{j+1} at time τ_{j+1}, the history $\mathcal{H}^*_{1,j}$ of generalized tables published before τ_{j+1}, the set R of respondents, and function ϑ, the output of our generalization algorithm is the generalized table T^*_{j+1}. Our algorithm can be roughly divided into 4 phases. While the first 3 phases are essentially identical to the ones of the algorithm in [5], Phase 4 is different, since in that phase (m, n)-historical safety is enforced. Adopting the notation of [5] we call $S_\cap$ the set of tuples in T_{j+1} that have been released before τ_{j+1}, and $S_\text{-}$ the remaining tuples in T_j.

- **Phase 1: Division.** This phase consists in partitioning the set of tuples in $S_\cap$ into *buckets*. Each bucket is uniquely identified by a signature among the ones of tuples in $S_\cap$, and it contains only tuples that appeared in $\mathcal{H}^*_{1,j}$ in QI-groups having the same signature of the bucket.
- **Phase 2: Balancing.** The balancing phase is applied in turn to each bucket. Its goal is to guarantee that every private value of the bucket's signature is represented by the same number of tuples in the bucket. Buckets are balanced by inserting tuples belonging to $S_\text{-}$ as long as this is possible; if no other tuples in $S_\text{-}$ can be used to balance the bucket, counterfeit tuples are inserted.
- **Phase 3: Assignment.** In this phase, the remaining tuples in $S_\text{-}$ are assigned to the existing buckets as long as they remain balanced. If no other tuple can be assigned to the existing buckets without violating balancing, new buckets are created, and the remaining tuples are assigned to the new buckets such that the new buckets are balanced. The cardinality of the signature of new buckets is greater than or equal to m.
- **Phase 4: Cor-Partition** This phase is applied in turn to each bucket. In this phase, buckets are partitioned into weak m-invariant QI-groups such that, if a novel historical correlation can be identified by matching the new QI-groups with those released during $\mathcal{H}^*_{1,j}$, then the cardinality of the sets of respondents involved in it is greater than or equal to n; i.e., QI-groups are hc-safe with degree n. For brevity, when the degree n of hc-safety is clear, we say that a QI-group is hc-safe (or hc-unsafe), omitting the degree of hc-safety. This phase is described in detail in the following of this section.

Cor-Partition. The algorithm pseudo-code is illustrated in Figure 5. Consider a generic bucket B composed of $s \cdot l$ tuples ($s \geq m$), where s is the cardinality of the signature of B (named $B.sig$), and $l \geq 1$. After an initialization phase (line 1), the algorithm creates, for each QI attribute in A^{qi}, a list of the tuples in B partially ordered according to their value for that attribute (line 2). We denote L_i the list regarding attribute A^{qi}_i, and $\overline{L}$ the set of such lists. Then, a cycle is repeated until every tuple in B is assigned to a QI-group (lines 3 to 20).

At first, each list is traversed in turn to obtain a weak m-invariant QI-group Q_i (lines 6 to 8) by selecting, for each private value belonging to the signature of

Input: parameters T^*_{j+1}, B, A^{qi}, n obtained from the Cor-Split algorithm.
Output: T^*_{j+1} incremented with the anonymization of tuples in B.

1: int $c := 0$; $\overline{L} := \emptyset$; $Q_i^{(old)} := \emptyset$
2: **for all** $A_i^{qi} \in A^{qi}$ **do:** List $L_i :=$ order(B, A_i^{qi}); $D_i := \emptyset$; $\overline{L} := \overline{L} \cup \{L_i\}$
3: **repeat**
4: $\quad \overline{Q} := \emptyset$; $\overline{SP} := \emptyset$
5: $\quad$ **for all** $L_i \in \overline{L}$ **do**
6: $\quad\quad$ **repeat**
7: $\quad\quad\quad$ QI-group $Q_i :=$ createQIG($L_i, D_i, B.sig$)
8: $\quad\quad$ **until** hcSafe(Q_i, n, j) $\vee$ ($|Q_i| < |B.sig|$)
9: $\quad\quad$ **if** $|Q_i| < |B.sig|$ **then:** $Q_i :=$ createQIG($L_i, \emptyset, B.sig$); $Q_i^{(temp)} := Q_i$; $c' := c$
10: $\quad\quad$ **while** $\big(\neg$ hcSafe(Q_i, n, j)$\big) \wedge \big(c' > 0\big)$ **do**
11: $\quad\quad\quad c'_i := c' - 1$; $Q_i := Q_i \cup QIG_{c'}$
12: $\quad\quad$ **end while**
13: $\quad\quad$ **if** $\neg$ hcSafe(Q_i, n, j) **then:** $Q_i := Q_i^{(temp)}$; Q_i.setCounterfeits()
14: $\quad$ **end for**
15: $\quad i' := i \in \mathbb{N} \,|\, Q_i.sp = \min_{\forall j \in \mathbb{N}} \{Q_j.sp\}$
16: $\quad QIG_c := Q_{i'}$; T^*_{j+1}.removeDuplicates(QIG_c) $T^*_{j+1} := T^*_{j+1} \cup QIG_c$
17: $\quad B := B \setminus QIG_c$; $c := c + 1$
18: $\quad$ **for all** $L_i \in \overline{L}$ **do:** $L_i := L_i$.remove(QIG_c); $D_i := \emptyset$
19: **until** $B = \emptyset$
20: **return** T^*_{j+1}

Fig. 5. The *Cor-Partition* algorithm

the bucket, the first tuple in L_i having that value, temporarily discarding those tuples L_i that would determine hc-unsafe QI-groups. Temporarily discarded tuples are randomly chosen from tuples in Q_i and replaced with other tuples in B having the same private value, as long as either the resulting QI-group is hc-safe, or no more tuples are available from B. In the latter case (lines 9 to 12), an hc-unsafe QI-group $Q_i^{(temp)}$ is created, and it is merged with a growing number of QI-groups previously created from the same bucket, until either the resulting QI-group is hc-safe, or no other available QI-group remains. In the latter case (line 13), $Q_i^{(temp)}$ is transformed by substituting a growing number of tuples in it with counterfeits, until the resulting QI-group Q_i is hc-safe. Note that counterfeit tuples are inserted only in the case in which no other operation is possible to generate a hc-safe QI-group.

Hence, after line 14, for each QI attribute in A^{qi} a hc-safe QI-group is available. For each of these QI-groups we call *semiperimeter* the sum of the normalized lengths of the interval of each QI value of tuples in it. Obviously, smaller semiperimeters correspond to finer-grained generalization. For this reason, the QI-group Q_j having the smallest semiperimeter is chosen (line 15). Then (lines 16 to 18), tuples in Q_j are removed from B, as well as from the lists in $\overline{L}$, and are added to T^*_{j+1}, after having possibly removed duplicate tuples (indeed, in line 11, Q_j could have been merged with QI-groups already inserted into T^*_{j+1}).

Input: parameters Q, n, j obtained from the Cor-Split algorithm.
Output: *true* if Q is hc-safe with degree n; *false* otherwise.

```
1: for int i := 0 to j do
2:    multiset (C, ω) := multiset(∅, ω)
3:    for all generalized tuple t* ∈ Q do
4:       respondent r := t*.r
5:       original tuple t := r.t
6:       QI-group Q' := t.QI(i)
7:       if Q' ≠ null then
8:          C := C ∪ Q'.id
9:       end if
10:   end for
11:   l := max_{c∈C}(ω(c))
12:   if |Q| − n < l < |Q| then
13:      return false
14:   end if
15: end for
16: return true
```

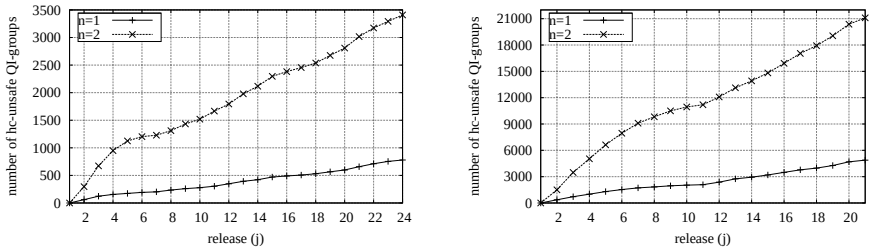
Fig. 6. Algorithm *hcSafe* for checking the hc-safety of a QI-group.

After that, if the bucket contains other tuples the algorithm continues from line 4; otherwise it returns the generalized table T^*_{j+1}.

Checking hc-safety. The algorithm for checking the hc-safety of a QI-group to be released in T^*_{j+1} is illustrated in Figure 6. Given a release history $\mathcal{H}^*_{1,j}$, a QI-group Q and a degree $n \in \mathbb{N}$, it follows from Definition 3 that in order to check whether Q is hc-unsafe it is sufficient to check whether it exists a set of l respondents whose tuples belonged to the same QI-group both in release T^*_j and $T^*_i \in \mathcal{H}^*_{1,j}$, with l smaller than $|Q|$ and greater than $|Q|-n$. Hence (lines 2 to 10), for each release $T^*_i \in \mathcal{H}^*_{1,j}$ the algorithm creates a multiset that contains, for each respondent r of tuples in Q, the unique identifier of the QI-group that included r's tuple in release T^*_i (if it exists). Then (line 11), the maximum multiplicity l of the elements of the multiset is calculated; i.e., l is the maximum number of respondents of tuples in Q whose tuples also belonged to the same QI-group in T^*_i. If it exists at least one release T^*_i such that the value l is smaller than $|Q|$ and greater than $|Q| - n$ the algorithm determines that Q is hc-unsafe with respect to the degree n; otherwise Q is hc-safe with degree n.

pvr-safety. The following lemma states a sufficient condition to ensure that a generalization function is pvr-safe.

Lemma 3 (sufficient condition for pvr-safety). *Let p be the probability of released tuples to be compromised, L the maximum number of times that a single tuple can be republished, $h \in (0, 1]$ the threshold for pvr-safety, and $m \in \mathbb{N}^+$ the*



(a) $m = 6$, $|T^*| = 60,000$ (b) $m = 6$, $|T^*| = 200,000$

Fig. 7. Number of hc-unsafe QI-groups generated by the m-invariance algorithm

required level of weak m-uniqueness. Then, if G is a generalization function enforcing (m, n)-historical safety with $n =$ n-Choose(p, L, m, h)*, then G is* pvr-safe *with threshold h.*

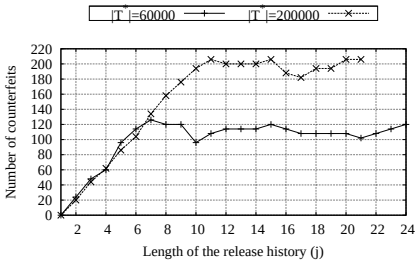
The soundness of the *Cor-Split* algorithm is proved by the following theorem.

Theorem 3 (pvr-safety of the *Cor-Split* algorithm). *The* Cor-Split *algorithm computes a pvr-safe generalization function.*

5 Experimental evaluation

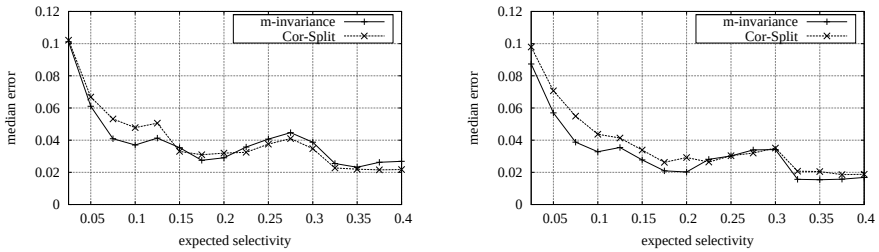
Experiments were performed using a real census dataset published by the *Minnesota Population Center* and available at `http://ipums.org/`. The dataset is composed of $600,000$ tuples. Each tuple stores information about an individual; it includes 4 QI-attributes (age, birthplace, education, gender) and one private attribute *income* having 50 possible values, each one representing an income range. In order to evaluate our technique with respect to different scenarios, we simulated insertions and deletions from the dataset at different rates. Hence, we started with a table T_1 including $200,000$ (resp. $60,000$) tuples, and we obtained a table T_2 by randomly deleting 10% (resp. 33%) of T_1's tuples and inserting the same number of tuples randomly chosen from unpublished tuples. The same procedure was repeated with the subsequent tables to obtain a history having length 21 (resp. 24). In these experiments we assumed that the probability of a tuple to be compromised is $p = 0.04$, the enforced level of (weak) m-invariance is $m = 6$, the maximum length of the release history of each tuple is $L = 21$ (resp. $L = 24$), and the safety threshold is $h = 0.1$. Given these parameters, the level n of (m, n)-historical safety to be enforced is $n = 3$. Results of experiments with different values for parameters m and n ($m = 4 \div 10$, $n = 1 \div 5$) are not reported here for lack of space; however, they essentially lead to the same conclusions as the ones reported below.

Historical correlations determined by m-invariance. The first set of experiments aimed at evaluating the threat determined by private value restriction



(a) $m = 6$, $n = 3$

Fig. 8. Number of counterfeits introduced by the *Cor-Split* algorithm



(a) $m = 6$, $n = 3$, $|T^*| = 60000$ (b) $m = 6$, $n = 3$, $|T^*| = 200000$

Fig. 9. Query error

functions when microdata are generalized applying the m-invariance technique. In particular, we adopted the algorithm in [5] to generalize microdata tables, and at any release we counted the number of released QI-groups that were hc-unsafe with respect to the degree $n = 3$; hc-unsafe QI-groups were recognized using the algorithm reported in Figure 6. Results in the considered scenarios are illustrated in Figure 7, and show that many released QI-groups may allow an adversary to derive historical correlations between small sets of respondents (having cardinality 1 or 2), determining relevant privacy threats.

Counterfeits and query error. The second set of experiments was performed on microdata generalized by a Java implementation of the *Cor-Split* algorithm. Tuples were generalized in order to enforce (m, n)-historical safety with $m = 6$ and $n = 3$. At first, we measured the number of counterfeit tuples introduced by *Cor-Split*. Results are illustrated in Figure 8 and show that in both scenarios the algorithm introduced a few counterfeits. Then, we compared the utility of microdata generalized by *Cor-Split* and by the algorithm for m-invariance in terms of the precision in answering aggregate queries (e.g., *count the number of individuals in the table whose QI-values belong to certain ranges*). Queries were randomly generated according to different values of *expected selectivity*, i.e., expected ratio of tuples to be returned by the query. For each value of expected

selectivity, $10,000$ queries were randomly generated. The imprecision in query answering was calculated in terms of the median error of query answers. The results reported in Figure 9 show that the accuracy of query answering obtained by *Cor-Split* is very close the to one observed with the use of the generalization algorithm for m-invariance, with the advantage of protecting from the threats we have identified.

6 Conclusions and future work

In this paper we have addressed privacy threats that may arise when a certain view over a dynamic dataset has to be released multiple times during its history. We have shown the limits of existing techniques to protect privacy in the case an adversary is able to recognize correlations between sets of tuples released in different views and even a small percentage of tuples is compromised. After having formalized the problem, we have provided a probabilistic study of the identified threats, and we have proposed a sound defense algorithm that has been experimentally validated. Future work includes the study of other attacks based on correlation between different releases. In particular, we are currently investigating the case in which tuples for the same respondent in different releases can have different private values; in this scenario our defense can still be effective against the attacks considered in this paper, but the adversary may exploit a different kind of historical correlation, based on private values associated to candidate respondents in a history of released tuples.

Acknowledgments

This work has been partially supported by the Italian Ministry of University and Research under grant PRIN-2007F9437X (project *ANONIMO*).

References

1. Samarati, P.: Protecting Respondents' Identities in Microdata Release. IEEE Transactions on Knowledge and Data Engineering **13**(6) (2001) 1010–1027
2. LeFevre, K., DeWitt, D.J., Ramakrishnan, R.: Incognito: Efficient Full-domain k-Anonymity. In: Proc. of SIGMOD 2005, ACM Press (2005) 49–60
3. Machanavajjhala, A., Gehrke, J., Kifer, D., Venkitasubramaniam, M.: l-Diversity: Privacy Beyond k-Anonymity. In: Proc. of ICDE 2006, IEEE Comp. Soc. (2006)
4. Li, N., Li, T., Venkatasubramanian, S.: t-Closeness: Privacy Beyond k-Anonymity and l-Diversity. In: Proc. of ICDE 2007, IEEE Comp. Soc. (2007) 106–115
5. Xiao, X., Tao, Y.: m-Invariance: Towards Privacy Preserving Re-publication of Dynamic Datasets. In: Proc. of SIGMOD 2007, ACM Press (2007) 689–700
6. Byun, J.W., Sohn, Y., Bertino, E., Li, N.: Secure Anonymization for Incremental Datasets. In: Proc. of SDM 2006, Third VLDB Workshop, Springer (2006) 48–63
7. Pei, J., Xu, J., Wang, Z., Wang, W., Wang, K.: Maintaining k-Anonymity against Incremental Updates. In: Proc. of SSDBM 2007, IEEE Comp. Soc. (2007)
8. Fung, B.C.M., Wang, K., Fu, A.W.C., Pei, J.: Anonymity for Continuous Data Publishing. In: Proc. of EDBT 2008, ACM (2008) 264–275