On Usage Control in Relational Database Management Systems
Obligations and Their Enforcement in Joining Datasets

Mortaza S. Bargh[1], Marco Vink[1] and Sunil Choenni[1,2]
[1]*Research and Documentation Centre, Ministry of Security and Justice, The Hague, The Netherlands*
[2]*Creating 010, Rotterdam University of Applied Sciences, Rotterdam, The Netherlands*
{m.shoae.bargh, m.e.vink, r.choenni}@minvenj.nl, r.choenni@hr.nl

Keywords: Access Control, Inner Join, Obligations, Privacy, Usage Control.

Abstract: When datasets are collected and accessed legitimately, they must still be used appropriately according to policies, guidelines, rules, laws, and/or the (current) preferences of data subjects. Any inconsistency between the data collection and data usage processes can conflict with many principles of privacy like the transparency principle, no secondary use principle, or intended purpose usage principle. In this contribution we show how the usage control for the inner join operation in vertically separated relational datasets can be characterized as pre and post obligations of the Usage Control (UCON) model. This type of obligations is defined not only by the state of the UCON object (i.e., a dataset) itself, but also with respect to the state of another dataset. Such dependency on two datasets/objects provides a new insight in UCON obligation constructs when applied to the join operation. We describe also a mechanism to realize the identified obligation in a database management system and present an example realization of the proposed mechanism. Furthermore, we enlist a number of methods to determine whether two given datasets can be joined.

1 INTRODUCTION

Currently data are created in an explosive rate with the surge of new services/applications as well as smart and sensory devices. Digitalization and e-administration, e-services, Big Data, Open Data, and Internet of Things are example cases that contribute to this data outpouring and overflow. Consequently, it becomes a common practice in (business) data analytics and data intensive applications to integrate data from different sources, of various types, of large volumes, and/or of high rates. These applications and services aim at easing our daily lives, providing insight in societal phenomena, or creating added values for businesses. Delivering these benefits, however, must not violate or compromise, for example, the privacy, commercial, and intellectual rights of individuals and parties who contribute their data to the data integration process.

For a long time, access control mechanisms have been used to protect the security and privacy of data. An access control mechanism controls the access to the data by granting or rejecting an access request. Although in this way the input datasets for a data integration process may be acquired or accessed legitimately, it is crucial for the output dataset of the data integration process to be legitimate and acceptable for all parties who provided the input datasets. For example, the privacy and business sensitivity requirements of these parties must be preserved. Today, personal devices produce more and more personal data than before. Big data analytics makes it possible to combine these data, resulting in (new) personal data that may expose the private lives of people in quite detail. Such data combinations may result in unexpected and harmful impacts on individuals. Therefore, access control is insufficient in current era of data expulsion.

Given the fact that the access to data is obtained legitimately, one needs to control how the data are used practically. Suppose that a tax officer needs to know the name, the annual income, the spouse's name, and the number of children of a person in order to carry out his/her tasks. It is not, however, the business of the tax officer to find out how many spouses or children per spouse a certain person (like a celebrity) has had. The system, therefore, should note such illegitimate use of attribute values and exclude them from the tax officer's access. Therefore, a query like "find all spouses of singer-X

Bargh, M., Vink, M. and choenni, s.
On Usage Control in Relational Database Management Systems - Obligations and Their Enforcement in Joining Datasets.
DOI: 10.5220/0006209801900201
In *Proceedings of the 3rd International Conference on Information Systems Security and Privacy (ICISSP 2017)*, pages 190-201
ISBN: 978-989-758-209-7

and for each spouse the name of the children" is an improper use of the attribute values and should not be executed.

Determining the (privacy) policies that govern such data integrations become steadily unforeseeable due to availability of vast amount of background information to data receivers and adversaries. For example, one cannot predetermine the datasets that will be encountered and integrated with a given dataset in the future. This makes it difficult to assess the potential risks in combining the released data with any other datasets (i.e., with the background information). This uncertainty relates to the extrinsic characteristics of data, e.g., the (privacy) issues of a given datasets in relation to other datasets. The other datasets exist in outside world due to, for example, sequential data release, multiple data release, continuous data release, collaborative data release, social networks, Big Data, and Open Data.

One may conclude that it is unwise to share data anymore. This policy appears to be too restrictive and unrealistic nowadays. Another solution direction is to devise and realize mechanisms that control compliance with data privacy policies after sharing the data with others, i.e., during the data usage lifecycle. This solution, which can be realized in controllable environments like an organization's Database Management System (DBMS), requires a flexible and adaptive framework that decides based on a data integration policy and enforces the decision at runtime. Hereby it becomes possible to deal with the issue of authorized-access and unauthorized-use of datasets (Choenni et al., 2016). To this end, for example, the Usage Control (UCON) model (Park and Sandhu, 2004) is one of the promising models.

Our research objective is to control the usage of relational datasets in volatile and dynamic settings i.e., when data analysts gradually and unforeseeably gain access to datasets and want to link/integrate a subset of these datasets. We limit our scope to relational databases and those structured datasets that are vertically separated. By vertically separated datasets we mean vertically distributed datasets, as illustrated in (Karr et al., 2007), which are not necessarily at different locations (i.e., they can be collocated as in the case of typical data warehouse environments). We consider the usage control for the inner join operation among these vertically separated datasets. Inspired by the UCON model, we specifically investigate: How the inner join operation can be framed in such a data usage control framework. This investigation results in a new insight in UCON obligation constructs. As our first contribution, we distinguish a new type of obligations where the state of the object (e.g., a dataset) is determined with respect to existence of another dataset. This type of dependency, to the best of our knowledge, has not been identified so far. As our second contribution, we present a mechanism to realize the identified obligation in a DBMS. As our third contribution, we present an example realization to illustrate how the proposed mechanism can be implemented and analyze the results. Furthermore, we enlist a number of methods for determining whether two given datasets can be joined.

The paper starts with a problem statement in Section 2 and provides some background information on access control and usage control in Section 3. Subsequently Section 4 presents our proposed approach and mechanism. Section 5 describes our example realization of the proposed mechanism and discusses its issues. Section 6 presents the related work and Section 7 captures our conclusions and future research directions.

2 PROBLEM STATEMENT

In this contribution we shall focus on the issue of authorized-access and unauthorized-use of datasets that are vertically separated, as described below.

2.1 Motivation

When collected datasets are accessed legitimately, they should still be used appropriately according to policies, guidelines, rules, laws, and/or the (current) preferences of data subjects. For example, in the context of business and public administration, data may be collected for a specific data registration (e.g., for hospital, municipality or judicial administration-purposes), due to a service operation (e.g., the list of website visitors or mobile telephony users), or for a research study (e.g., a study over household or crime victimization). As such, the data can be collected within different legal domains corresponding to regions/countries, public sectors (e.g., healthcare, justice, and trade), etc. Many issues may arise when the data are used in another context than the one they were collected for and accessed to. Such an inconsistency between the data collection and data usage processes can conflict with, for instance, many principles of privacy like the transparency principle, no secondary use principle, or intended purpose usage principle.

Nowadays many cases arise where it is important to deal with unauthorized usage of those datasets that are accessed to in an authorized way. Businesses, organizations and services merge in various public, private and semi public sectors. For example, Google has merged various services like Gmail, Google+, Google Drive; and Facebook has acquired Instagram and WhatsApp. Such strategic merges require integration of information systems, with various datasets that are generally collected for different purposes and within various contexts. There are also Open Data initiatives to release public sector data to citizens as a means of, among others, government transparency, innovation and economic growth stimulator, and public participation in government (Dawes, 2010b)(Dawes, 2010a). Such initiatives motivate and encourage combining data from various sources in order to deliver added value services and insights. In such cases where information systems and data are integrated, there are potential risks of privacy breaches when (self-provided) data of users are combined with data retrieved from elsewhere (Bargh and Choenni, 2013); (Fung et al., 2010).

Within one organization collected data can also be used in an unauthorized way due to, for example, secondary use, i.e., data that are collected for one purpose but are used for another one. Crowdsourcing, for instance, is a means of collecting relevant data in an affordable way. The resulting datasets may encompass some sorts of personal data from participants such as profile data (including their names, email addresses and phone numbers), activity data (indicating their sporting, sleeping, and eating habits), and situational data (revealing their visited locations, adjacency to other users/objects, and conversation buddies). Such personal data must basically be accessible to a limited number of authorized entities (like system administrators and specific services/systems) and be used in an authorized way (like for the specified purpose). Authorized insiders with ill intentions (i.e., those insider intruders or employees with questionable ethics as mentioned in (Agrawal et al., 2002)) may reveal and misuse such personal information that they have access to for their illegitimate purposes like personal satisfaction, financial gains, and political benefits. Revealing personal information makes data subjects (i.e., those individuals and organizations that the data are about) vulnerable to cyber attacks such as identity theft, phishing and spams, and privacy breaches. Therefore, the crowd may become fearful and unwilling to participate in the data collection process due to being subjected to such threats and becoming victims of such attacks. Even when users voluntarily participate in crowdsourcing, they desire sometimes their personal information not to be processed when, for instance, they are at certain situations like during evenings, in the weekends, and during holidays.

Even highly sensitive data attributes may be disclosed or inferred by means of easily accessible data and data linkage. Kosinski et al. (Kosinski et al., 2013) show that easily accessible digital records of behavior, e.g., Facebook Likes, can be used to automatically and accurately predict a range of highly sensitive personal attributes (such as sexual orientation, ethnicity, religious and political views, personality traits, intelligence, happiness, use of addictive substances, parental separation, age, and gender). De Montjoye et al., (2013) analyzed a dataset of fifteen months of human mobility data for 1.5 million individuals. According to (de Montjoye et al., 2013), human mobility traces are highly unique. For example, when the location of an individual is specified hourly at the precision level of mobile network cells, it is possible to uniquely identify 95% of the individuals based on four spatiotemporal points. They also found that even rather highly aggregated datasets provide little anonymity.

2.2 Problem Formalization

For scientific studies, our research center maintains a data warehouse that contains various datasets from several organizations involved in the Dutch justice system. These organizations include the Police, the Public Prosecution Office, the courts, the Central Fine Collection Agency, the agency of correctional institutions (i.e., prisons) and the Probation Service. In some projects the data of more than one organization can be used to measure the performance of the Dutch justice chain by combining the necessary datasets from these organizations. For combining these datasets, a case number is used to uniquely identify a judicial case across all these organizations. Although our data analysts have access to all datasets, they may not combine all datasets due to privacy and other reasons (for instance, as the number and contents of the datasets are growing over time, one may combine old and new data under certain conditions).

Inspired by (Agrawal et al., 2002), we focus on the (privacy) policy violation issues that arise when linking/ integrating datasets in relational datasets during their usage time. Assume a data analyst, who works for project A, obtains access to dataset A at time t_A. At a later time $t_B > t_A$ the data analyst, who now works also for project B, gets access to dataset

B. We shall, therefore, denote these datasets also by $A(t_A)$ and $B(t_B)$ notations, respectively.

Dataset A and B contain data tuples represented by $\boldsymbol{a}_i = (a_{i,1}, a_{i,2}, \cdots, a_{i,m_A})$ and $\boldsymbol{b}_j = (b_{j,1}, b_{j,2}, \cdots, b_{j,m_B})$, respectively, where $i: 1, \cdots, n_A$ and $j: 1, \cdots, n_B$. Every tuple $\boldsymbol{a}_i$ is defined over single valued attributes from set $ATT^A = \{att_1^A, att_2^A, \cdots, att_{m_A}^A\}$. In other words, dataset A is a subset of the Cartesian product of $\text{dom}(att_1^A) \times \text{dom}(att_2^A) \times \cdots \times \text{dom}(att_{m_A}^A)$, in which $\text{dom}(att_k^A)$ is the set of the values that can be assumed by attribute att_k^A, where $k: 1, \cdots, m_A$. A tuple $\boldsymbol{a}_i$ in dataset A is an ordered list of attribute values to which a unique identifier is attached. Without loss of generality, we assume this identifier corresponds to attribute att_1^A, which is drawn from $\text{dom}(att_1^A)$. Similarly, every tuple $\boldsymbol{b}_j$ is defined over single valued attributes from set $ATT^B = \{att_1^B, att_2^B, \cdots, att_{m_B}^B\}$ and dataset B is a subset of the Cartesian product of $\text{dom}(att_1^B) \times \text{dom}(att_2^B) \times \cdots \times \text{dom}(att_{m_B}^B)$. A tuple $\boldsymbol{b}_j$ in dataset B is an ordered list of attribute values to which a unique identifier is attached. Without loss of generality, we assume this identifier corresponds to attribute att_1^B, which is drawn from $\text{dom}(att_1^B)$.

We assume that datasets A and B are partially vertically separated, i.e.,

a) $ATT^A \cap ATT^B \neq \emptyset$
b) $ATT^A \neq ATT^B$; because otherwise datasets A and B become horizontally separated (and thus they provide the very same information about those entities whose tuples appear in both datasets),
c) For some i and j we have $a_{i,1} = b_{j,1}$ where $a_{i,1} \in \text{dom}(att_1^A)$ and $b_{j,1} \in \text{dom}(att_1^B)$. I.e., data tuples $\boldsymbol{a}_i$ and $\boldsymbol{b}_j$ refer to the same entity.

From time t_B on, the data analyst has access to both datasets A and B for two different purposes: For Project A execution and for project B execution. It is foreseeable that combining/joining datasets A and B may not be allowed from the viewpoint of project A, project B or both. This lack of permission for joining datasets A and B can be due to, for example, privacy or information sensitivity reasons.

We define an inner join action A_C as a tuple $\langle A(t_A), B(t_B), P_A(t), P_B(t), IJC, t \rangle$, where

- Parameter IJC represents the criterion, condition or predicate for the inner join action,
- $A(t_A)$ and $B(t_B)$ are datasets A and B, obtained by the data analyst at times t_A and t_B, respectively.
- Parameter t, being $t \geq t_B > t_A$, represents the time of executing the inner join operation.
- $P_A(t)$ and $P_B(t)$ are the (privacy) policies associated with datasets A and B at time t. These policies, which are obtained by the data analyst at times t_A and t_B, respectively, can be adapted during the lifecycle of the corresponding datasets.

In order to allow the inner join action A_C to be carried out at t, there should be two requirements satisfied, namely:

- The resulting dataset should not violate the (privacy) policy of project A. This is denoted by requirement $R_A(B(t_B), t \mid A(t_A), P_A(t))$, where the notation should be read as: Requirement for project A in regard to dataset $B(t_B)$ to be considered for the join operation at time t, given project A's own dataset $A(t_A)$ and own policy $P_A(t)$ which are acquired at times t_A and t, respectively.
- The resulting dataset should not violate the privacy policy of project B. This is similarly denoted by requirement $R_B(A(t_A), t \mid B(t_B), P_B(t))$.

The research questions to be addressed in this contribution are:

- How can the UCON model be characterized for the inner join operation of the datasets?
- How can we determine when the inner join is (dis)allowed?
- How is it possible to realize the resulting restricted join functionality?

3 BACKGROUND

In this section we present the theoretical background on access and usage control models.

3.1 Access Control

Traditionally, gaining access to a resource (e.g., a service, document, computer system, and processing time of a computer) has been realized by a system functionality called 'access control'. Access control can be defined as the ability to permit or deny access to a particular resource by a particular entity (Lazouski et al., 2010). The entity that seeks access to the resource and the resource that is sough by the entity are referred to as *subject* and *object*, respectively, in access control terminology. The access to an object can be in a specific mode like read, write and execute. These are usage permissions, or so-called *rights*, that the subject is allowed to carry out on the object. Note that a particular right is not predefined and it exists at the time of the authorization (Lazouski et al., 2010). In our example, the data analyst is the subject, dataset A for project B and dataset B for project A are the objects, and the 'inner join' is the right. Figure 1 illustrates a traditional access control model. As mentioned in (Hilty et al., 2005) the reference monitor in Figure 1 is a control program that monitors and prohibits actions.

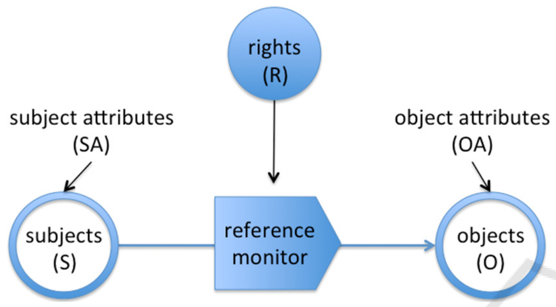

Figure 1: A traditional access control model.

Traditionally Discretionary Access Control (DAC), Mandatory Access Control (MAC), and Role Based Access Control (RBAC) models have been used. The DAC model may use a set of predicates to represents access constraints/rules. These access rules are often stored in an access control matrix, see the survey paper (Lopez et al., 2004) and the references therein for the material presented in the rest of this subsection. For an object there can be a so-called Access Control List that specifies which subjects have which permissions/rights to the object. For a subject, on the other hand, there can be a Capability List to specify the access rights of the subject to various objects.

The MAC model, which originated from military, focuses on the flow of information within a system. The model assigns security labels to objects and subjects, called as Classification Label (to represent the object's sensitivity) and Clearance Label (to represent the subject's trustworthiness), respectively. The model grants a subject with access to an object if their labels match from the viewpoints of their classification (e.g., top secret, confidential, non secret) and category properties (e.g., management level, department level, and project level). Compared to DAC, MAC requires higher implementation costs due to the complexity of planning and management of access rights.

Both DAC and MAC models require considerable amount of time and effort overhead when a new user is introduced into an organization. The new subject, in such cases, should be related to every resource in the organization. This process is also prune to human errors. Therefore RBAC is proposed by introducing roles as a link between subjects and objects. In RBAC subjects are authorized for roles and roles are authorized for objects to hold certain permissions or rights. Hereby instead of establishing subject-object associations one needs to establish two sets of subject-role associations and role-object associations.

There are other access control methods in the literature that we do not mention for brevity of the presentation. Traditional access control models have been used successfully in many application domains for many years. These traditional models are mostly suitable for closed organizational environments, where the subjects and objects are well known and when the sensitivity and trustworthiness of the objects and subjects are well defined and rather static. In modern application settings, where for example social networks, Big Data, and information (sharing) systems across organizations like in our case are dealt with, one needs to cope with rather dynamic environments to authorize (previously unknown) entities who want to access and use objects with dynamic sensitivity, within varying contextual situations, across multiple organizational domains and boundaries, and with a commitment to unprecedented conditions.

3.2 Usage Control

To cope with the shortcomings of the traditional access control models, usage control models (like UCON model (Sandhu and Park, 2003), (Park and Sandhu, 2004) and (Zhang et al., 2005)) are introduced. The UCON model extends the traditional models to include also controlling the access decisions during the object's usage interval (i.e., access decision *continuity*) and to allow also adapting the access criteria before, during and after object usage interval (i.e., attribute *mutability*). The continuity of decisions and mutability of attributes in UCON allow adapting to the changes of subject, object and environmental attributes before, during or after the data usage period. For example, the number of subjects that concurrently may access the object can change depending on the consumption intensity.

As illustrated in Figure, the reference monitor in the UCON model uses three types of decision-making factors. The first type is called *authorizations*. These authorizations include those predicates that put constraints on subject and object attributes. The attributes of the subject (e.g., the name, age, role, and nationality) in UCON are similar to Capability List in DAC and Clearance Label in MAC. The attributes of the object (e.g., document type, content sensitivity, and data ownership) in UCON are similar to Access Control List in DAC and Classification Label in MAC (Park and Sandhu, 2004).

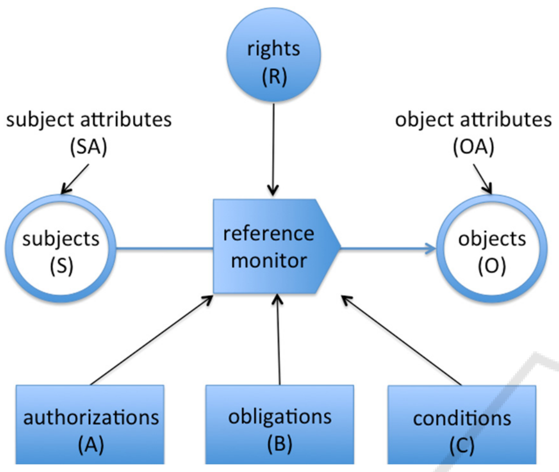

Figure 2: An illustration of the UCON model, adopted with adaption from (Park and Sandhu, 2004) and (Zhang et al., 2005).

The other two types of decision-making factors in the UCON model are conditions and obligations, which are not uniquely defined in the literature (Colombo and Ferrari, 2014). For *conditions*, the authors in (Park and Sandhu, 2004) consider the environmental or system-oriented constraints that should hold before or during the object's usage interval. Examples conditions are those related to the time of the day, room temperature, and disastrous situation. As such, conditions are not dependent of the subject and the object (i.e., the data) directly. We shall elaborate upon *obligations* (especially in the context of relational databases) in Section 4.1.

The reference monitor in UCON controls the access to and usage of the object (e.g., data items) by the subject. Similarly to (Hilty et al., 2005), we regard the UCON "reference monitor rather liberally to describe control programs that can not only monitor and prohibit actions, but can also *trigger* corrective actions such as the application of penalties" (Hilty et al., 2005).

4 DESIGN AND REALISATION

In this section we adapt the UCON model to the problem at hand. First in Subsection 4.1 we frame the data integration scenario as UCON obligations. Subsequently in Subsection 4.2 we describe the policy decision-making component of the reference monitor that determines whether datasets A and B are joinable at a given time. Finally in Subsection 4.3 we formalize the policy decision-enforcement component of the reference monitor for the inner join operation.

4.1 Specifics of Obligations

Obligations are an active area of research currently. Particularly, the enforcement of those obligations that are concerned with fulfilling some tasks and actions during or after the usage of the object (i.e., data) are open research issues (Lazouski et al., 2010).

Obligations mandate those *actions* that someone should execute before, during or after an object's usage interval (Lazouski et al., 2010). For example, the credit card owner must be informed in 30 days after a credit card being used, a license agreement must be signed before data usage, an ad must be watched for 20 seconds, and the document must be downloaded just one time. When the actions are executed appropriately, the subject could access or could continue to use the object. Note that the entity that fulfills the obligation, i.e., carries out the action(s), might be the subject or someone else, depending on the usage scenario. Similarly, the entity on which an obligation activity is carried out might be the object or something else.

In (Colombo and Ferrari, 2014) the authors consider the enforcement of obligations, which are derived from privacy policies, on relational database management operations. They regard obligations as "the *constraints* that refer to *the (expected) state of the data [object]* stored in the database" at the time in which the object is accessed or used (i.e., invoking a SQL code). For example, the bank account balance must be positive after withdrawing.

In summary, *one can regard obligations as the constraints (a) on the state of the object (i.e., the data in the database) or (b) on specific actions being executed by someone. Fulfillment of both constraint types can be required before, during or after an object's usage interval.*

Our usage control on the inner join operation in this contribution can be categorized as "obligation" because the authorization of the right (i.e., the inner join of datasets A and B) is constrained with the state of the objects (e.g., the datasets A and B for projects A and B, respectively). Our first contribution hereto is that we distinguish a new type of obligations where the state of an object (i.e., dataset A or dataset B) is determined with respect to another object. This type of dependency, to the best of our knowledge, has not been identified so far in the UCON literature.

In distributed usage control, where information is disseminated in open networks, post usage obligations are widely applicable (Lazouski et al., 2010). We observe that this is also the case in our centralized usage control, when an operation on a data object (like dataset A in our scenario) is dependent of other (upcoming) data object (like dataset B in our scenario). More specifically, from requirements $R_A\big(B(t_B), t \,\big|\, A(t_A), P_A(t)\big)$ for project A and $R_B\big(A(t_A), t \,\big|\, B(t_B), P_B(t)\big)$ for project B, where $t \geq t_B > t_A$, one can define the data integration obligations for the data analyst as the UCON's subject. These obligations can be of type:

- Pre-obligation for project B when $t = t_B$,
- Post obligation for project A (because $t \geq t_B > t_A$) and for project B when $t > t_B$.

So when $t = t_B$ the constraint on datasets A and B can be of type pre-obligation (for project B) and post-obligation (for project A) simultaneously. This duality is another new insight, to the best of our knowledge, provided in this contribution.

4.2 Decision Making

The reference monitor should decide on whether two datasets A and B can be joined or not based on requirements $R_A\big(B(t_B), t \,\big|\, A(t_A), P_A(t)\big)$ and $R_B\big(A(t_A), t \,\big|\, B(t_B), P_B(t)\big)$ that, in turn, depend on the momentary policies of project A and B (i.e., $P_A(t)$ and $P_B(t)$) as well as on the datasets of projects A and B (i.e., $A(t_A)$ and $B(t_B)$). Joining two datasets may extend the attribute sets ATT^A and ATT^B to set $ATT^{A\cup B} = ATT^A \cup ATT^B = \{att_1^{A\cup B}, \cdots, att_{m_{A\cup B}}^{A\cup B}\}$.

For deciding on the join of two datasets, one could check whether the resulting combination of attributes is allowed or not. A domain expert can control this based on existing laws, regulations, and policies. Alternatively, similarly to (Byun and Li, 2008) and based on the purposes for which datasets $A(t_A)$ and $B(t_B)$ are collected, the reference monitor can check whether the privacy policies of project A and project B allow their data objects to be part of the resulting table or not. This can be done through controlling the possibility of any inconsistency in data collection and usage purposes. For example, if dataset $A(t_A)$ and $B(t_B)$ are collected for commercial and system administration purposes, respectively, then the join should not go on assuming that commercial and administrative purposes are disjoint/inconsistent.

Another way to decide on allowing the join operation is to control whether there would be undesired information leakage due to the join operation or not. To explain this aspect, let assume that every attribute in ATT^A and ATT^B and thus in $ATT^{A\cup B}$ can be represented by a random variable. For example, random variable ATT_i^A corresponds to attribute $att_i^A \in ATT^A$. (NB: In the rest of this section we misuse the notations of sets ATT^A, ATT^B and $ATT^{A\cup B}$ and assume they represent sets of attributes as well as the corresponding attribute random variables.)

Further, let random variable set $\boldsymbol{S} \subset ATT^{A\cup B}$ be the set of those random variable attributes after the join operation that are (privacy) sensitive. In our setting, $\boldsymbol{S}$ includes at least one member, i.e., we have $ATT_1^A \ni \boldsymbol{S}$ (being the same as $ATT_1^B \ni \boldsymbol{S}$, as we assumed). Let random variable set $\boldsymbol{X} = ATT^{A\cup B} \setminus \boldsymbol{S}$ be the set of those (privacy) non-sensitive random variable attributes after the join operation. Thus, sets $\boldsymbol{S}$ and $\boldsymbol{X}$ encompass those attributes that cannot be and can be, respectively, revealed to the data analyst in our scenario according to requirements R_A and R_B.

In order to determine the information leakage in the dataset resulted from the inner join operation A_C as defined by $\langle A(t_A), B(t_B), P_A(t), P_B(t), IJC, t\rangle$, one may use the mutual information function (Sankar et al., 2013)(Wang et al., 2014). The amount of information leaked about random variables in $\boldsymbol{S}$ due to random variables in $\boldsymbol{X}$ can be determined by mutual Information $I(\boldsymbol{S};\, \boldsymbol{X})$, which should ideally be zero. If this value reaches an unacceptably high level due to the join operation, then the join should be disallowed. One can also aim at the information leakage for any $\boldsymbol{S'} \subset \boldsymbol{S}$ and examine whether $I(\boldsymbol{S'};\, \boldsymbol{X})$ reaches an unacceptably high level or not due to the join operation. The thresholds of the unacceptably high level can be determined from the policies of $P_A(t)$ and $P_B(t)$ at or up-to runtime t.

4.3 Decision Enforcement

We defined the inner join action A_C as $\langle A(t_A), B(t_B), P_A(t), P_B(t), IJC, t\rangle$. Datasets A and B are represented by their tuples as $A(t_A) = \{\boldsymbol{a}_i |\, i{:}\, 1, \cdots, n_A\}$ and $B(t_B) = \{\boldsymbol{b}_j \big|\, j{:}\, 1, \cdots, n_B\}$. The inner join action A_C can further be specified as those members of the Cartesian product of sets $A(t_A)$ and $B(t_B)$, i.e., $\{\boldsymbol{a}_i | \, i: 1, \cdots, n_A\} \times \{\boldsymbol{b}_j | \, j: 1, \cdots, n_B\}$, for which

a) The predicate IJC holds for those tuples that $a_{i,1} = b_{j,1}$, and

b) Requirements R_A and R_B hold.

In other words,

$$A_C \triangleq \{\boldsymbol{a}_i \times \boldsymbol{b}_j | \, i: 1, \cdots, n_A; \, j: 1, \cdots, n_B; \, F_{i,j} = T\},$$

where $F_{i,j}$ is a Boolean function defined as the conjunction of the following operands:

- O1: $(IJC(\boldsymbol{a}_i, \boldsymbol{b}_j) = T) \wedge (a_{i,1} = b_{j,1})$,
- O2: $R_A(B(t_B), t \mid A(t_A), P_A(t)) = T$,
- O3: $R_B(A(t_A), t \mid B(t_B), P_B(t)) = T$.

In practice, due to for example privacy requirements, the (personal) identifiers $a_{i,1}$ and $b_{j,1}$ in datasets A and B (i.e., for all i and j) are anonymized. In this section we assume they are pseudonymized by Hash functions $H_A(\cdot)$ and $H_B(\cdot)$, respectively. In this case, operand O1 can be written as:

$$\text{O}_1\text{: } (IJC(\boldsymbol{a}_i, \boldsymbol{b}_j) = T) \wedge \left(H_A(a_{i,1}) \equiv H_B(b_{j,1})\right),$$

where $H_A(a_{i,1}) \equiv H_B(b_{j,1})$ holds if $a_{i,1} = b_{j,1}$. Note that

a) The operator ≡ represents the fact that there is a one-to-one mapping possible between $H_A(a_{i,1})$ and $H_B(b_{j,1})$, and it does not imply equality necessarily; and

b) The probability that $a_{i,1} \neq b_{j,1}$ if $H_A(a_{i,1}) \equiv H_B(b_{j,1})$ is (extremely) negligible for collision-resistant hash functions practically.

In Subsection 5.1 we shall present a technique to realize operator ≡ in DBMSs.

5 EVALUATION

In this section we present an example realization of the proposed mechanism and subsequently discuss its characteristics and limitations.

5.1 Example Realization

To illustrate how to realize the proposed mechanism we explain an example with 3 tables A, B, and C, each containing two attributes: an identification number ID and a data attribute. The data analyst has access to all three tables. We assume that the analyst is authorized to join tables A and B, but (s)he is not authorized to join table A and C (for the generic scenario see Section 2.2). To illustrate the idea we use these simple tables without loss of generality (i.e., instead of a single table one can use a set of tables just like the case of our data warehouse).

Table 1: Table A.

ID	ATTR
1	Source value A1
2	Source value A2
3	Source value A3

Table 2: Table B.

ID	ATTR
1	Source value B1
3	Source value B3

Table 3: table C.

ID	ATTR
2	Source value C2
3	Source value C3

Using the unique identifiers, i.e., attribute ID, as the primary key the data analyst can now easily combine the data from tables A and C:

```
> Select * from a join c on c.id = a.id
```

As the first step of our implementation, we replace the original unique identifiers, which are unique per criminal case, by a new set of global identifiers, which are globally, i.e., among these tables, unique. Alternatively, one could use hash functions, as suggested in Note (b) in Subsection 4.3. Consequently those tuples from different tables, which correspond to the same case/entity, will no longer have the same identifiers in the new dataset. The mapping between the new identifiers of an entity via the old identifier of the entity, see Table 7, is made one-to-one and it is stored in a separate repository, called identifier repository, safely. The identifier repository realizes the operation ≡ in practice (see Note (a) in Subsection 4.3).

The joins in the proposed approach will be made through the identifier repository, where we will use a usage rights table to check whether the tables can be combined. The tables with the new identifiers, the identifier repository, and the usage rights table are shown in Table 4, Table 5, Table 6, Table 7 and Table 8 below.

Table 4: Table DWH_A.

ID	ATTR
N1	Source value A1
N2	Source value A2
N3	Source value A3

Table 5: Table DWH_B.

ID	ATTR
N4	Source value B1
N5	Source value B3

Table 6: table DWH_C.

ID	ATTR
N6	Source value C2
N7	Source value C3

Table 7: Table ID_REP (identifier repository).

ID_SRC	ID_DWH	SRC_DATASET
1	N1	A
2	N2	A
3	N3	A
1	N4	B
3	N5	B
2	N6	C
3	N7	C

Table 8: Table USAGE_RIGHT (the authorization policy).

DATASET1	DATASET2
A	B
B	A

Now direct joining like in the previous SQL query will return an empty result set, since the identifiers no longer match. In the next query the join is performed through the identifier repository, and includes a check on the usage rights:

```
> select dwh_a.*, dwh_b.*
> from dwh_a
    > join id_rep rep1 on rep1.id_dwh
      = dwh_a.id
    > join id_rep rep2 on rep2.id_src
      = rep1.id_src
    > join dwh_b on dwh_b.id =
rep2.id_dwh
    > join usage_right on
      (usage_right.dataset1 =
       rep1.src_dataset and
usage_right.dataset2 =
rep2.src_dataset)
```

For tables DWH_A and DWH_B this will result in a dataset with the combined data. When a similar query is run for table DWH_A and DWH_C there will be an empty result set because the join is not allowed according to the USAGE_RIGHT table.

The final step in our implementation is to set access control to the tables in the example. The data analyst is not allowed to see the identifier repository or to change the usage rights. Joins are carried out through a stored procedure, which has access to the identifier repository. It generates a table with the join of the two given tables if allowed by the USAGE_RIGHT table. The result does not contain the old or new identifiers from the identifier repository, but a new independent set of identifiers is generated. In this way the identifiers in the new dataset cannot easily and readily be tracked back to the original datasets, nor will it be possible to easily link the identifiers of the joined dataset to other (possible future) datasets.

5.2 Discussion and Limitations

As described before the database consists of different datasets. Within a dataset the tables can be joined as usual so the performance will be the same as for standard joins. When tables of different datasets are joined there is a performance penalty since the identifiers have to be looked up in the identifier repository and usage control rights have to be checked. As we have seen this adds three extra joins. Obviously this is less efficient than a single join. However modern database engines are very efficient in doing joins and can be optimized by the use of techniques like indexing.

Furthermore we propose a scenario where the combination of two datasets is generated once with a new identifier set. Within the new combined dataset the joins will have no performance loss. The generation of the new dataset will have a cost, but in a data warehouse setting this is usually not a problem.

The USAGE_RIGHT table is a good location to hook in extra usage control decision factors like authorizations, conditions or obligations. For example only tuples for adults may be combined, the datasets may only be joined during a fixed period, or the requester has to sign a privacy statement. For example:

Table 9: Extended table USAGE_RIGHT.

DATASET1	DATASET2	AUTH	COND	OBLIG
A	D	D.age>18	System.date<15 July	Agree on privacy policy

We created a new identifier (i.e., pseudo ID) for those tuples in the resulting dataset of the join operation. In practice, however, it is possible for a data analyst with ill intentions to infer those identifiers in table A (see Table 4) and in table B (see Table 5) that their records appear also in the resulting table of the join operation. The data analyst can infer the mappings in the identity repository for these records by using/matching the values of the other attributes in Table A, Table B and the resulting table from their join, using techniques described in (Narayanan and Shmatikov, 2008); (Choenni et al., 2010). This can be seen as an attack on the pseudonimization part of the proposed approach. As mentioned in Subsection 5.1, our proposed approach attempts to impede those attackers (i.e., data analysts, the employee with questionable ethics as mentioned in (Agrawal et al., 2002)) who want to directly track the identifiers in the new dataset back to the original datasets.

6 RELATED WORK

This section provides a review of some related works on obligations and on controlling the join operation in relational databases.

6.1 Obligations

Obligations are considered an important means of realizing privacy and security aware systems, nevertheless, as mentioned before, there is still no consensus on the precise meaning of the term obligation (Colombo and Ferrari, 2014).

One misalignment in the literature relates to the concepts of *pre-obligation*, *on-going obligation* and *post-obligation*. In the UCON-ABC model of (Park and Sandhu, 2004) pre-obligations and on-going obligations are recognized. The concept of post-obligation is added to the UCON model in (Katt et al., 2008). Note that, not within the context of the UCON model, others (e.g. (Hilty et al., 2005); (Gama et al., 2006) and (Bettini et al., 2003)) had already considered obligations as requirements that must be fulfilled after data access has been done. In (Ni et al., 2008) the authors introduce pre and post obligations to the Role Based Access Control (P-RBAC). In (Bettini et al., 2003) pre-obligations are characterized as provisions.

In (Hilty et al., 2005) obligations are further classified in two dimensions of being *(un)observational* and being *temporally-(un)bounded*. The observational aspect characterizes whether the reference monitor can observe the fulfillment of the obligation or not. The temporal bound-ability characterizes whether obligations should be fulfilled in a certain time period or not (i.e., should be checked for ever). These criteria define four obligation types:

- Bounded future and observable (e.g., pay a fee within a fixed number of days, data item may not be accessed for x days, the reference monitor must notify the data owner about the access within x days).
- Bounded future and non-observable (e.g., data item must be deleted within x days, data item must not be redistributed in the next x days),
- Unbounded future and observable (e.g., re-access the data at least every x days to maintain freshness of data as demanded by some data protection regulations), and
- Unbounded future and non-observable (e.g., data should be used only for statistical analysis, data should not be distributed further, each usage of the data must be reported immediately, or must be protected with protection level L until it is declassified by the owner).

The obligation for the join operation is temporally unbounded, i.e., it holds for as long as there is a possibility of joining any pair of vertically separated datasets (e.g., in our case $A(t_A)$ and $B(t_B)$). The obligation for the join operation is also unobservable (i.e., in project A one cannot foresee that project B is going to link dataset $A(t_A)$ with its dataset $B(t_B)$ and vice versa). By introducing the reference monitor we ensure the join operation to be observable to the central reference monitor and, eventually, those non-observable data protection requirements to be adhered to. This strategy is also mentioned in (Hilty et al.. 2005), whereby an unobservable obligation is enforced by transforming a non-observable obligation into a set of provisions and observable obligations that prevent unwanted executions. One can think of not only this “strict sense of enforcement”, i.e., “the prevention of unwanted executions of a system through system monitoring and denying actions that would violate the policy”, but also additional corrective or “compensating actions (e.g. penalties) in case the execution violates the policy” (Hilty et al., 2005). Unlike in our case, obligations in (Hilty et al., 2005) are those conditions that must be imposed in the future (i.e., the time after an access is authorized) and (Hilty et al., 2005) uses provisions instead of obligations to refer to those conditions that must be imposed by/at the time of an access being authorized. In our case, furthermore, we showed that it is possible for obligations to be of types pre-obligation/on-going-obligation and post-obligation at the same time.

6.2 Relational Databases

As the work presented in this contribution relates to usage control for relational databases and privacy protection for the join operation in relational DBMS, we review some related works on these topics in the following.

In (Colombo and Ferrari, 2014) the authors consider enforcing obligations, which are derived from privacy policies, on relational database management operations. While Colombo and Ferrari (2014) consider SQL operations in general, we focus on the inner join operation particularly, and zoom in its peculiarities from the viewpoints of the parties (i.e., projects) involved in the operation. Similarly to our work, (Colombo and Ferrari, 2014) considers obligations as constraints on "the [expected] state of the data [(i.e., the object)] stored in the database at the time in which the execution of an action (i.e., SQL code) is invoked (like the account balance after withdrawing must be positive)". We go one step further and take into account also the state of each of the two datasets of the join operation with respect to the other dataset.

Secure Multi Party Computing (SMPC) methods aim at computing a function F on vertically or horizontally distributed datasets for data mining or data processing purposes, without requiring the raw datasets to be shared with a central entity (a Trusted Third Party, TTP) or with the peers. In this way every party learns only the result of function F and its own dataset. SMPC methods are applied in combination with the SQL join operation in multi-party settings in (Laur et al., 2013) for horizontally distributed datasets. As mentioned above, the objective of SMPC is to compute a specific function F on the joined dataset in a privacy preserving way (i.e., without sharing the datasets with a TTP or the peers). For example, the function F in (Laur et al., 2013) delivers the number of rows in the join table (for which the join predicate holds). In our setting, however, the aim is to authorize the inner join operation or not, regardless of which function the data analyst intends to apply to the resulting datasets in the future. As such, our approach acts as a sort of on-fly access control (thus a usage control) mechanism rather than a privacy preserving data mining or data processing mechanism.

7 CONCLUSION

To deal with the issue of authorized-access and unauthorized-use of datasets, there is a need for a flexible and adaptive framework to decide on and enforce the data integration policy at runtime. We motivated this need for the inner join operation in vertically separated relational datasets where one cannot predetermine which datasets would be encountered and integrated with a given dataset.

We characterized the usage control model of the inner join operation by the obligations of the UCON model. Here the authorization of the right (i.e., the inner join of datasets A and B) is constrained with the state of the object. In this study we distinguished a new type of obligations where the state of the object (i.e., dataset A or dataset B) is determined with respect to another dataset. These obligations can be of both pre-obligation and post-obligation types simultaneously, depending on the timing of the join operation with respect to the moments of datasets A and B availability. This duality is another new insight provided in this contribution.

We proposed a few methods for making decision whether two datasets A and B can be joined or not. The decision can be based on whether the resulting combination of attributes is allowed or not using the domain knowledge, comparing the data collection and data usage purposes of datasets A and B, or information leakage about the sensitive attributes due to the join operation. Finally we proposed a mechanism to enforce the obligations and realized it in an example realization. The reference monitor of the proposed usage control is realized as a stored procedure that maps the pseudo identifiers from the identifier repository to the original identifiers, checks the usage rights to determine if a join is allowed, and joins the data if that is the case.

Our scheme uses different pseudo identifiers for the input and output datasets of the join operation and relies on a secure lookup table to map among these pseudo identifiers during the realized join functionality. This solution creates a first barrier against the threat of inferring pseudo identifiers. Searching for a more robust and secure solution, the future research can be directed towards, for example, adopting and adapting the method of Polymorphic Encryption and Pseudonymisation (Verheul et al., 2016).

REFERENCES

Agrawal, R. et al., 2002. Hippocratic databases. *Proceedings of the 28th international conference on Very Large Data Bases*, 4(1890), pp.143–154.

Bargh, M.S. & Choenni, S., 2013. On preserving privacy whilst integrating data in connected information systems. In *Proceedings of International Conference on Cloud Security Management (ICCSM'13)*. Guimarães, Portugal.

Bettini, C. et al., 2003. Provisions and Obligations in Policy Rule Management. *Journal of Network and Systems Management*, 11(3), pp.351–372.

Byun, J. & Li, N., 2008. Purpose based access control for privacy protection in relational database systems. *The VLDB Journal*, pp.603–619.

Choenni, S. et al., 2016. Privacy and security in smart data collection by citizens. In J. R. Gil-Garcia, T. A. Pardo, & T. Nam, eds. *Smarter as the New Urban Agenda*. Springer, pp. 349–366.

Choenni, S., Dijk, J. van & Leeuw, F., 2010. Preserving privacy whilst integrating data: Applied to criminal justice. *Information Polity*, 15(1–2), pp.125–138.

Colombo, P. & Ferrari, E., 2014. Enforcing obligations within relational database management systems. *IEEE Transactions on Dependable and Secure Computing*, pp.1–14.

Dawes, S.S., 2010a. Information Policy Meta-Principles: Stewardship and Usefulness R. H. Sprague Jr., ed. *Proceedings of the 43rd Hawaii International Conference on System Sciences (HICSS-43)*, pp. 1–10.

Dawes, S.S., 2010b. Stewardship and usefulness: Policy principles for information-based transparency. *Government Information Quarterly*, 27(4), pp.377–383.

Fung, B.C.M. et al., 2010. Privacy-preserving data publishing. *ACM Computing Surveys*, 42(4), pp.1–53.

Gama, P., Ribeiro, C. & Ferreira, P., 2006. Heimdhal: A History-based Policy Engine for Grids. In *Sixth IEEE International Symposium on In Cluster Computing and the Grid (CCGRID)*.

Hilty, M., Basin, D. & Pretschner, A., 2005. On obligations. *Computer Security–ESORICS 2005*, pp.98–117.

Karr, A.F. et al., 2007. Secure, privacy-preserving analysis of distributed databases. *Technometrics*, 49(3), pp.335–345.

Katt, B. et al., 2008. A general obligation model and continuity: enhanced policy enforcement engine for usage control. *Proceedings of the 13th ACM symposium on Access control models and technologies (SACMAT '08)*, pp.123–132.

Kosinski, M., Stillwell, D. & Graepel, T., 2013. Private traits and attributes are predictable from digital records of human behavior. *Proceedings of the National Academy of Sciences of the United States of America*, 110(15), pp.5802–5.

Laur, S., Talviste, R. & Willemson, J., 2013. From oblivious AES to efficient and secure database join in the multiparty setting. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7954 LNCS, pp.84–101.

Lazouski, A., Martinelli, F. & Mori, P., 2010. Usage control in computer security: A survey. *Computer Science Review*, 4(2), pp.81–99.

Lopez, J., Oppliger, R. & Pernul, G., 2004. Authentication and authorization infrastructures (AAIs): a comparative survey. *Computers & Security*, 23(7), pp.578–590.

de Montjoye, Y.-A. et al., 2013. Unique in the Crowd: The privacy bounds of human mobility. *Scientific reports*, 3, p.1376.

Narayanan, A. & Shmatikov, V., 2008. Robust de-anonymization of large sparse datasets open datasets. In *IEEE Symposium on Security and Privacy (SP'08)*. pp. 111–125.

Ni, Q., Bertino, E. & Lobo, J., 2008. An obligation model bridging access control policies and privacy policies. *Proceedings of the 13th ACM symposium on Access control models and technologies - SACMAT'08*, p.133.

Park, J. & Sandhu, R., 2004. The UCON ABC usage control model. *ACM Transactions on Information and System ...*, 7(1), pp.128–174.

Sandhu, R. & Park, J., 2003. Usage Control : A Vision for Next Generation Access Control. , pp.17–31.

Sankar, L., Rajagopalan, S. & Poor, H., 2013. Utility-Privacy Tradeoff in Databases: An Information-theoretic Approach. *IEEE Transactions on Information Forensics and Security*, pp.1–1.

Verheul, E. et al., 2016. *Polymorphic Encryption and Pseudonymisation for Personalised Healthcare*, Available at: https://www.semanticscholar.org/paper/Polymorphic-Encryption-and-Pseudonymisation-for-Verheul-Jacobs/7dfce578644bc101ae4ffcd0184d2227c6d07809 .

Wang, W., Ying, L. & Zhang, J., 2014. On the relation between identifiability, differential privacy and mutual-information privacy. In *In 52nd IEEE Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. pp. 1086–1092. Available at: http://arxiv.org/abs/1402.3757.

Zhang, X. et al., 2005. Formal model and policy specification of usage control. *ACM Transactions on Information and System Security*, 8(4), pp.351–387.