Privacy in GLAV Information Integration

Alan Nash[1,⋆] and Alin Deutsch[2,⋆⋆]

[1] IBM Almaden Research Lab
anash@us.ibm.com
[2] University of California San Diego
deutsch@cs.ucsd.edu

Abstract. We define and study formal privacy guarantees for information integration systems, where sources are related to a public schema by mappings given by source-to-target dependencies which express inclusion of unions of conjunctive queries with equality. This generalizes previous privacy work in the global-as-view publishing scenario and covers local-as-view as well as combinations of the two.

We concentrate on *logical* security, where malicious users have the same level of access as legitimate users: they can issue queries against the global schema which are answered under "certain answers" semantics and then use unlimited computational power and external knowledge on the results of the queries to guess the result of a secret query ("the secret") on one or more of the sources, which are not directly accessible. We do not address issues of *physical* security, which include how to prevent users from gaining unauthorized access to the data.

We define both absolute guarantees: how safe is the secret? and relative guarantees: how much of the secret is additionally disclosed when the mapping is extended, for example to allow new data sources or new relationships between an existing data source and the global schema? We provide algorithms for checking whether these guarantees hold and undecidability results for related, stronger guarantees.

1 Introduction

We define and analyze formal privacy guarantees for information integration systems. Such guarantees have been recently studied for the case of *database publishing* where views of the underlying sources are exposed to users (see Related Work). This corresponds to the global-as-view closed-world scenario. Here we extend this study to include the case of *database integration*.

We study the case where sources are related to a public schema by mappings given by source-to-target dependencies which express inclusion of unions of conjunctive queries with equality. Such framework is also known as *global-local-as-view (GLAV)* and was introduced in [12] and studied in [4,5,15,11] as a generalization of *global-as-view (GAV)* and *local-as-view (LAV)* integration [22,14,16]. Users may issue queries against the public schema for which the information integration system returns the certain answers [22,14,16].

⋆ Work performed while the author was at the University of California, San Diego.

⋆⋆ Funded by an Alfred P. Sloan fellowship and by NSF CAREER award IIS-0347968.

T. Schwentick and D. Suciu (Eds.): ICDT 2007, LNCS 4353, pp. 89–103, 2007.

We consider the case where the attacker is a malicious user who has no further access to the sources than any other user. All the attacker can do is issue queries against the integration system and apply arbitrary computational power on the answers to these queries together with external knowledge to obtain some information ("the secret") which the defender wishes to conceal. We do not address such security issues as how to prevent unauthorized access to the data sources.

The goal of the defender is to determine to what extent the information system is vulnerable to attacks of this kind. The defender specifies the secret as a query against one or several data sources. The objective of the attacker is to obtain the answer to or at least partial information on the answer to the secret on the data sources which the defender wants to conceal.

Database Publishing versus Data Integration. Prior work on privacy in databases has not addressed the data integration setting, but has focused on database *publishing*, in which materialized views of the underlying source are exposed, thus corresponding to a global-as-view, closed-world integration scenario [14]. In database publishing, the attacker can access the full extent of any view V by simply issuing the identity query SELECT * FROM V. Therefore every attack strategy considered in the literature assumes the availability of all view extents. This assumption no longer holds in a data integration setting, where there is no materialized view instance and queries posed by the attacker are answered under so-called *certain answers* semantics [16]. Consequently, the intuitive attack based on the identity query is in general ineffective.

Example 1. Assume there is one source S in the system over a private schema consisting of the single binary relation $PA(patient, ailment)$, recording what ailment each patient is treated for. The information integration system exports the public schema which consists of two binary relations $PD(patient, doctor)$ and $DA(doctor, ailment)$, connecting patients to doctors they see and doctors to ailments they treat. The source S is registered via the single source-to-target constraint ϕ: $\forall p, a\, PA(p, a) \rightarrow \exists d\, PD(p, d) \wedge DA(d, a)$. This registration is a standard source-to-target embedded dependency [11], and it is an equivalent way to capture the local-as-view registration [14] using the conjunctive query view $PA(p, a) :- PD(p, d), DA(d, a)$. The registration basically means that the private database owner cannot provide doctor information, but states that each patient is treated by some doctor for the ailment.

Now consider an attack modeled after the classic privacy breach strategy in database publishing. It would start by issuing the queries $Q_{PD}(p, d) :- PD(p, d)$ and $Q_{DA}(d, a) :- DA(d, a)$, in an attempt to find the patient-doctor and doctor-ailment associations in order to subsequently join them and to thus reduce the possible patient-ailment combinations to guess among [18,8].

However, the certain answers to Q_{PD}—which are the tuples in $Q_{PD}(T)$ for *every* target instance T satisfying constraint ϕ—give precisely the empty set, regardless of the extent of source instance S. To see why, notice that the doctor name is not specified, so for each particular name constant, there is at least one possible T which does not contain it. A similar argument shows that Q_{DA} has no certain answers either.

Therefore our first task in studying the defense strategies in data integration is to identify what queries the attacker should pose to gain the most insight into the secret. For Example 1 above, it turns out that one well-chosen query suffices.

Example 2. In the setting of Example 1, all the attacker needs to do to reveal the source completely is to issue the query $Q_\phi(p, a) :- PD(p, d), DA(d, a)$. Under certain answer semantics, the result is precisely the extent of source table PA. Therefore, the query Q_ϕ is optimal for the attacker, since after obtaining the extent of PA she may compute *any* secret query on PA.

In general however, determining the "optimal" queries to start the attack with is challenging. It is a priori not even clear that there exists one single set of queries leading to the highest privacy breach. Even if this were the case, it is not clear that such a set would be finite; an infinite series of queries (each possibly depending on the answer to its predecessors) could potentially outperform any finite series of queries.

Contributions. We study privacy in the context of information integration systems, which introduces substantial new aspects over data publishing. To the best of our knowledge, this is the first such study. Our specific contributions include the following.

(a) We identify optimal attack (and therefore defense) strategies. In particular, we show that there is a finite set of unions of conjunctive queries which the attacker can issue that are optimal in the sense that no further information is gained by issuing additional queries. The required queries are very different in LAV and GAV integration scenarios, but our approach unifies the attack strategy extending it to a GLAV setting.

(b) We define absolute and relative privacy guarantees, dependent on the source S_0, and provide corresponding algorithms to check them against the optimal attack strategy. The absolute guarantees depend only on the current state of the information integration system, while the relative ones relate the state of the information integration system before and after a change in the mapping between the data sources and the public schema. Such a change may arise for example as a result of introducing new data sources, or if a source owner decides to publish additional proprietary data. The guarantees (formalized in Section 4) are:

1. The source is not completely exposed (i.e. the attacker cannot infer its exact extent without resorting to external knowledge).
2. The secret is not completely exposed.
3. The secret has not been further exposed (i.e. nothing new can be learned about it) by extending the source-target mapping to export further information.
4. The source has not been further exposed by extending the source-target mapping.

Note that Guarantee 1 does not depend on the secret; if the source is completely exposed, the attacker may compute the result of any query whatsoever against it. Furthermore, we identify Guarantee 4 as the natural adaptation to data integration of the notion of *perfect privacy* introduced in data publishing [18].

(c) While in general the complexity of our algorithms ranges from $\mathbf{NP}$ to $\mathbf{\Pi_2^P}$ in the size of the source instance, we identify a practically relevant PTIME case.

(d) We define additional guarantees corresponding to the ones above, but defined in terms of *all possible sources*. These guarantee flavors are of significant interest as they do not require re-checking after each update on the sources. We show however that all

but one of them are undecidable (we do not know whether the latter is decidable or not). These results establish the source-dependent guarantees as the best we can hope for in the trade-off between strength of guarantees and their decidability.

Paper Outline. The remainder of this paper is organized as follows. In Section 2 we introduce the required notation. In Section 3, we model the general strategy the attacker follows and in Section 4 we present the guarantees that the defender can provide. In Section 5 we provide algorithms to check the guarantees and our theoretical results, which include correctness and complexity of the algorithms. We discuss related work in Section 6 and conclude in Section 7. The proofs are shown in the extended version of this paper [19].

2 Preliminaries

Queries. Unless otherwise specified, all our queries are $\mathrm{UCQ}^{=}$ queries; that is, unions of conjunctive queries with equalities (we also allow constants). We only consider safe queries (i.e. with all head variables appearing in their body). Given a query Q and a database D, $Q(D)$ is the answer to Q on D.

Constraints. A *constraint* is a boolean query. We denote sets of constraints with capital Greek letters and individual constraints with lowercase Greek letters. We consider constraints of the form $\forall \bar{x}(P(\bar{x}) \rightarrow Q(\bar{y}))$, where $\{\bar{y}\} \subseteq \{\bar{x}\}$, where $\{\bar{x}\}$ is the set of free variables in the *premise* P, where $\{\bar{y}\}$ is the set of free variables in the *conclusion* Q, and where P and Q are $\mathrm{UCQ}^{=}$ queries. In constraints, we allow Q to be unsafe; intuitively, the safety of the constraint comes from the safety of P. Such constraints are similar to and generalize *embedded dependencies* [1] by allowing disjunctions; we call them $\mathrm{IC}(\mathrm{UCQ}^{=})$ constraints because they express containment of $\mathrm{UCQ}^{=}$ queries. Unless otherwise specified, all our constraints are of this kind. We write $D \models \Sigma$ if the database D satisfies the set of constraints Σ.

Information Integration Systems. As in [16], we define an *information integration system* to consist of four parts $(\sigma_S, \sigma_T, \Sigma, S)$: σ_S is the *source* or *private* schema, σ_T is the *target* or *public* schema, Σ is a finite set of constraints over the joint schema $\sigma_S \cup \sigma_T$ specifying how the sources relate to the targets, and S is the source. We assume that σ_S and σ_T are disjoint. We say that T is a *possible target* of S under Σ if $(S, T) \models \Sigma$. That is, if the database obtained from putting S and T together satisfies Σ. We define the *certain answers* to a query Q over σ_T under the constraints Σ on source S to be $\mathrm{cert}^{Q}_{\Sigma}(S) := \bigcap_{(S,T)\models\Sigma} Q(T)$. That is, $\mathrm{cert}^{Q}_{\Sigma}(S)$ is the set of tuples which appear in $Q(T)$ for every possible target T of S. This corresponds to what is known as the *open world assumption*. We say that Σ is *source-to-target* if every constraint in Σ contains only relation symbols from σ_S in the premise and relation symbols from σ_T in the conclusion. All our information systems are given by source-to-target mappings, in the spirit of the Clio system [11].

GAV, LAV and GLAV Integration. This setting generalizes two very important particular cases occurring frequently in practice and in the literature [14,16]. In Global-As-View (GAV) integration systems, the conclusion of each constraint in Σ is a *single* relational atom, with all variables appearing in the premise (see Example 4 below). In

Local-As-View (LAV) integration, the premise is a single relational atom, with all variables appearing in the conclusion (as seen in Example 1). The general case is therefore also known as Global-Local-As-View (GLAV).

3 Modeling Attacks Against the Integration System

Recall that we consider the scenario where each of the user's queries Q is processed by the information integration system $\mathcal{I} := (\sigma_S, \sigma_T, \Sigma, S_0)$ and the certain answers $\mathrm{cert}^Q_\Sigma(S_0)$ are returned to the user. The user has no other access to the source S_0.

The attacker is a malicious user whose objective is to obtain the answer to or at least partial information on secret, specified as the answer to a query Q_Z against the source S_0. The attacker has no further access to the sources than ordinary users.

However, we consider that all users know the source schema and how it relates to the target schema using source-to-target constraints. It has been argued before even in the context of database publishing [18] that assuming otherwise would be naive. After all, the only way of communicating to users the meaning of data contributed by a source is via a source schema (be it the real one, or an abstract, conceptual one) and its relationship to the target schema. For instance, even if users do not know the names of the hospital database tables and their attributes, they understand enough about the application domain to assume that these include patients, doctors, ailments, and they can easily observe whether patient names and ailments are hidden or not. It is therefore prudent to assume that in most applications, attackers can reverse-engineer a source schema, or an abstraction thereof which is equivalent with respect to information capacity. The attack against privacy can then be conducted using the real or the reverse-engineered schema.

Since the attacker understands the semantics of the source schema, she will have no trouble formulating the query Q_Z which specifies the secret.[1] The only obstacle in her way is the integration system's rejection of queries which are not formulated against its target schema. Instead, the attacker may issue several queries against the integration system, then apply arbitrary computational power on the answers in order to obtain information about the secret $Q_Z(S_0)$.

Possible sources and secrets. Note that the attacker cannot distinguish among sources that lead to the same answers for the queries she issued. She thus reduces the set of possible sources/secrets to those which are indistinguishable w.r.t. the issued queries, applying external knowledge to distinguish among the reduced set. Clearly, the optimal outcome for the attacker is to reduce the set of sources/secrets to guess from as much as possible by posing the "right" queries. To state our guarantees, we formalize the notions of "possible sources" and "possible secrets." Intuitively, possible sources and possible secrets are those which cannot be distinguished, respectively, from the source S_0 and the secret $Q_Z(S_0)$ exclusively by issuing queries to $\mathcal{I}$; discriminating among them requires the attacker to use external knowledge.

We say that S is a *possible source* if the certain answers to *any* query Q for $(\sigma_S, \sigma_T, \Sigma, S)$ are exactly the same as the certain answers to Q for $(\sigma_S, \sigma_T, \Sigma, S_0)$. That is,

[1] For brevity, we refer to Q_Z as the "secret query", though we assume that only its answer is secret, not its definition.

for all queries Q, $\mathrm{cert}^Q_\Sigma(S) = \mathrm{cert}^Q_\Sigma(S_0)$. We say that Z is a *possible secret* if it is the result $Q_Z(S)$ of the secret query Q_Z on some possible source S. In particular, the source S_0 is a possible source and the secret $Q_Z(S_0)$ is a possible secret. Clearly, any source which has the same possible targets as S_0 is a possible source.

The attacker wishes to obtain a set of secrets/sources which approximates as best as possible the set of possible secrets/sources. She will then distinguish among these using external knowledge and, if necessary, randomly guess among the secrets/sources which remain indistinguishable even by external knowledge.

Attacker's external knowledge. The attacker's external knowledge has been modeled in the literature as additional constraints on the secrets or on the sources, or as a probability distribution on them [18,8]. Here, we abstract away from the particular representation, modeling it with two "black box" oracles PICKSOURCE and PICKSECRET. These represent any means of reducing the input possibilities based on external knowledge, followed by a random pick from the reduced set (if it is not a singleton).

PICKSECRET accepts as input a finite description of a set $\mathcal{Z}$ which is an approximation of the set of possible answers to the query Q_Z, and picks one secret from $\mathcal{Z}$. The following is a general strategy for the attacker in case PICKSECRET is available:

Procedure ATTACKSECRET

1. Issue several queries $Q_1, \ldots, Q_k$ against $\mathcal{I}$ to obtain $A_1, \ldots, A_k$ where $A_i := \mathrm{cert}^{Q_i}_\Sigma(S_0)$.
2. Using $A_1, \ldots, A_k$, compute a finite description $\Sigma_\mathcal{Z}$ which approximates as well as possible the set $\mathcal{Z}$ of possible secrets (that is, the set of answers to $Q_Z(S)$ for those sources S which satisfy $A_i = \mathrm{cert}^{Q_i}_\Sigma(S)$).
3. Return PICKSECRET($\Sigma_\mathcal{Z}$)

Similarly, PICKSOURCE accepts as input a finite description $\Sigma_\mathcal{S}$ of a set which approximates the possible sources and picks one of them. The following is a general strategy for the attacker in case only PICKSECRET is available:

Procedure ATTACKSOURCE

1. Issue several queries $Q_1, \ldots, Q_k$ against $\mathcal{I}$ to obtain $A_1, \ldots, A_k$ where $A_i := \mathrm{cert}^{Q_i}_\Sigma(S_0)$.
2. Using $A_1, \ldots, A_k$, compute a finite description $\Sigma_\mathcal{S}$ which approximates as well as possible the set $\mathcal{S}$ of possible sources S (that is, those which satisfy $A_i = \mathrm{cert}^{Q_i}_\Sigma(S)$).
3. Set $S :=$ PICKSOURCE($\Sigma_\mathcal{S}$)
4. Return $Q_Z(S)$.

The attacker's access to PICKSECRET, but not to PICKSOURCE models the case when she has no external knowledge about the possible sources, but may have sufficient independent knowledge to form an opinion about the possible secrets. We assume that PICKSOURCE may use PICKSECRET as a subroutine whenever both are available and that the attacker chooses to use PICKSOURCE whenever it is available.

4 Privacy Guarantees

The goal of the defender is to determine to what extent the information system $\mathcal{I} = (\sigma_S, \sigma_T, \Sigma, S_0)$ is vulnerable to attacks of the kind outlined in Section 3. The defender specifies the secret as a query Q_Z over σ_S. We analyze what kinds of guarantees the defender can provide and how he can verify whether they hold. We consider both

absolute guarantees, pertaining to how private the secret is for $\mathcal{I}$ and *relative* guarantees, pertaining to whether the secret has been exposed further in going from $\mathcal{I}$ to a new system $\mathcal{I}'$.

Our privacy guarantees focus on the crucial steps 1 and 2 in the general attack strategies. In these, the attacker attempts to facilitate the task of the oracle as much as possible by restricting the set of options to guess from. The fewer options are obtained, the less external knowledge is needed to guess the secret.

In Section 5, we investigate how good an approximation of possible secrets and sources the attacker can obtain. We obtain there the following surprising result:

Corollary 1 (of Theorem 1 in Section 5). *There exists a finite set of queries whose certain answers can be used to construct a finite axiomatization of the sets of possible sources and secrets.*

A conservative defender must therefore assume that any attacker is able, by posing a carefully chosen set of queries, to obtain a *precise* description of the sets of possible sources and secrets. This is why we focus our guarantees on these sets.

Absolute Guarantees. We now introduce two minimal guarantees guarding against full disclosure of the source, respectively secret. The worst case for the defender is when the certain answers to some finite set of queries $\mathcal{Q}$ are sufficient to determine the source S_0 exactly. In this case, the attacker may obtain not only the secret $Q_Z(S_0)$, but any information she wishes of the source under our assumptions.

Guarantee 1. *The source S_0 is not completely exposed by the information system $\mathcal{I}$. That is, there are at least two possible sources.*

Even if the source is not completely exposed, the secret might be. That is, there is more than one possible source, but the result of the secret query on all of them is the same (in short, there is only one possible secret). In this case the attacker may not know the source S_0, but she may learn the secret $Q_Z(S_0)$.

Guarantee 2. *The secret $Q_Z(S_0)$ is not completely exposed by the information system $\mathcal{I}$. That is, there are at least two possible secrets.*

Relative Guarantees. Guarantees 1 and 2 only avoid a complete privacy breach in which source, respectively secret are fully exposed. This is of course the weakest guarantee one could provide. Ideally, we would like the guarantee that nothing can be "learned" about the secret given the information system. The following example however shows that such a guarantee is unreasonably strong and is violated by most systems, which is why we need to set our sights on more relaxed guarantees.

Example 3. Consider an information system whose only source relation contains tuples associating the patient with the ailment he suffered from and the doctor who treated him: PDA(*patient,doctor,ailment*). The secret, as in Example 1, is the association between patients and their ailment: $Q_Z(p, a) :- PDA(p, d, a)$. The source registration only exports the projection of this source relation on its doctor attribute: $\forall p, d, a\ PDA(p, d, a) \rightarrow D(d)$ (where D is the target schema). Since neither patients nor ailments are registered, this registration is seemingly safe. However, an attacker can

still learn from it some (small amount of) information about the secret. Indeed, if the registered list of doctors is empty, then the source relation must be empty as well, so no patient can suffer from any ailment. If however there is even one doctor in the registered list, then there is a non-zero probability of a certain patient suffering from some disease. Clearly, the attacker has "learned" something about the secret upon observing the list of doctors, and the idealized guarantee is violated. At the same time, ruling out this registration boils down to asking the source owner to not register any data, even if it avoids the attributes involved in the secret query.

Since the absolute Guarantees 1 and 2 are too weak and the idealized guarantee considered above is too strong, we consider a more pragmatic class of *relative* guarantees. These assume that the data owner is willing to live with the current exposure of the secret or source, but wants to make sure that changing the constraints of the information system will not lead to further exposure.

There are two strong relative guarantees the defender can provide. The first applies in case the defender knows that the attacker has no external knowledge about possible sources (but may have external knowledge about possible secrets):

Guarantee 3. If the attacker has no external knowledge about the possible sources, *then secret $Q_Z(S_0)$ has not been further exposed in going from the information system $\mathcal{I} := (\sigma_S, \sigma_T, \Sigma, S_0)$ to the information system $\mathcal{I}' := (\sigma_S, \sigma_T, \Sigma', S_0)$. That is, the set of* possible secrets *under Σ is the same as the set of possible secrets under Σ'.*

The second guarantee applies when the defender cannot safely assume that the attacker will not distinguish among sources.

Guarantee 4. *The secret $Q_Z(S_0)$ has not been further exposed in going from the information system $\mathcal{I} := (\sigma_S, \sigma_T, \Sigma, S_0)$ to the information system $\mathcal{I}' := (\sigma_S, \sigma_T, \Sigma', S_0)$. That is, the set of* possible sources *under Σ is the same as the set of possible sources under Σ'.*

Example 2 in the introduction illustrates a case when Guarantee 1 fails, as there is a client query which fully reveals the source. Therefore, for any secret query Q_Z, Guarantee 2 fails as well, since the attacker can retrieve the full secret by running Q_Z on the exposed source. There are cases when the underlying source is not fully exposed (Guarantee 1 holds), but the secret is (Guarantee 2 fails). For lack of space, we illustrate such a scenario in the extended version [19], where we also show a scenario where Guarantee 3 holds but Guarantee 4 fails.

Source-independent guarantees. Guarantees 1, 2, 3, and 4 are all given in terms of a specific source S_0. For each such Guarantee i, we can define a corresponding Guarantee i' which has the same statement, but instead of referring to *some* source S_0, is quantified over *all* sources. These source-independent guarantee flavors are of significant interest as they do not require re-checking after each update on the sources.

5 Algorithms

In this section we outline algorithms for checking Guarantees 1 through 4. These algorithms are based on reduction to the problem of checking implication of constraints.

The implication problem for constraints is to determine whether, given a set of constraints Σ and a constraint ϕ, Σ implies ϕ, written $\Sigma \models \phi$. $\Sigma \models \phi$ holds if every database that satisfies Σ also satisfies ϕ. In general, checking implication of $\mathrm{IC}(\mathrm{UCQ}^{=})$ constraints is undecidable, as this class includes functional and inclusion dependencies, for which the implication problem is undecidable [1]. However, our reduction yields constraints which we call *convergent* for which checking implication and equivalence of two sets of constraints is in $\mathbf{\Pi_2^P}$ (Theorem 2). Checking whether $\Sigma \models \phi$ holds for convergent constraints can be done by a well-known procedure known as the *chase* [1]. We do not describe this procedure here; instead we assume we have a procedure IMPLIES to check whether $\Sigma \models \phi$ holds. We say that Σ and Σ' are equivalent (which we write $\Sigma \equiv \Sigma'$) in case Σ implies every constraint in Σ' and conversely.

We now reduce the problem of checking guarantees to the implication problem. For instance, to check Guarantee 1 and Guarantee 2, the idea is to find a set of constraints Δ_1 which axiomatize the possible sources (respectively, possible secrets) and a set of constraints Δ_2 which axiomatize the actual source (respectively, the actual secret) and to check whether Δ_1 implies Δ_2. Guarantee 1 (respectively, Guarantee 2) holds if and only iff $\Delta_1 \not\models \Delta_2$. Since as it turns out Δ_1 and Δ_2 are convergent sets, the latter implication is decidable.

The constraints are obtained by the following procedures: $\textsc{AxInstance}(D)$ returns constraints which axiomatize the database D. That is, $D' \models \textsc{AxInstance}(D)$ iff $D = D'$. AXSOURCES yields constraints which axiomatize the possible sources. AXSECRETS returns constraints which axiomatize the possible secrets. Before detailing the procedures, we show how they yield an algorithm for checking the various guarantees. The algorithm is inspired by the following corollary of Theorem 1 below.

Corollary 2 (of Theorem 1)

1. *Guarantee 1 holds iff* $\textsc{AxSources}(\mathcal{I}) \not\models \textsc{AxInstance}(S_0)$.
2. *Guarantee 2 holds iff* $\textsc{AxSecrets}(\mathcal{I}, Q_Z) \not\models \textsc{AxInstance}(Q_Z(S_0))$.
3. *Guarantee 3 holds iff* $\textsc{AxSecrets}(\mathcal{I}, Q_Z) \equiv \textsc{AxSecrets}(\mathcal{I}', Q_Z)$.
4. *Guarantee 4 holds iff* $\textsc{AxSources}(\mathcal{I}) \equiv \textsc{AxSources}(\mathcal{I}')$.

For instance by Corollary 2 we can use the procedures IMPLIES, AXINSTANCE and AXSOURCES to check Guarantee 1 as follows (Guarantees 2 through 4 are checked similarly):

Procedure $\textsc{GuaranteeOneHolds}(\mathcal{I})$
Set $\Delta_1 := \textsc{AxSources}(\mathcal{I})$. Set $\Delta_2 := \textsc{AxInstance}(S_0)$. Return not $\textsc{Implies}(\Delta_1, \Delta_2)$.

We define our procedures next.

In Algorithm 1, R^D denotes the extent of relation R in database D, and $\bar{c} \in R^D$ ranges over all tuples in R^D. In Algorithm 2, given a constraint $\phi \in \mathrm{IC}(\mathrm{UCQ}^{=})$ we define Q_ϕ to be the $\mathrm{UCQ}^{=}$ query whose body is the conclusion of ϕ and whose head is $Q_\phi(\bar{x})$ where $\bar{x}$ are the free variables in the conclusion of ϕ. We define P_ϕ similarly as the query obtained from the premise of ϕ.

Notice that the procedure which issues queries against the integration system is AXSOURCES, and that these queries are precisely those corresponding to the conclusions of the source-to-target constraints. The auxiliary procedure AXONE used within AXSOURCES gives constraints which are satisfied precisely by the sources that agree with

Algorithm 1. AXINSTANCE(D)

returns constraints which are satisfied precisely by database D.
That is, $D' \models$ AXINSTANCE(D) iff $D = D'$.

for every relation R in the signature $\sigma(D)$ of D **do**
  Set $\delta_R^1 := R(\bar{x}) \rightarrow \bigvee_{\bar{c} \in R^D} \bar{x} = \bar{c}$.
  Set $\delta_R^2 := R(\bar{x}) \leftarrow \bigvee_{\bar{c} \in R^D} \bar{x} = \bar{c}$.
end for
return $\{\delta_R^i : R \in \sigma(D), i \in \{1, 2\}\}$.

Algorithm 2. AXSOURCES($\mathcal{I}$)

returns constraints axiomatizing the possible sources.
That is, $S \models$ AXSOURCES($\mathcal{I}$) iff S is a possible source.

for every $\phi \in \Sigma$ **do**
  Issue the query Q_ϕ against $\mathcal{I}$ to obtain $A_{Q_\phi} := \mathrm{cert}_\Sigma^{Q_\phi}(S_0)$.
end for
return $\bigcup_{\phi \in \Sigma}$ AXONE($\Sigma, Q_\phi, A_{Q_\phi}$).

Algorithm 3. AXONE($\Sigma, Q_\phi, A_{Q_\phi}$)

returns constraints which are satisfied precisely by the sources which agree with S_0 on the result A_{Q_ϕ} of query Q_ϕ (which is the conclusion of constraint ϕ).

Set $R_\phi :=$ REWRITE(Σ, Q_ϕ)
// Set Σ_ϕ^r to the set of constraints over schema $\sigma_S \cup \{Q_\phi\}$ which capture R_ϕ:
let R_ϕ be the UCQ$^=$ query $R_\phi(\bar{x}) :- \bigvee_i B_i(\bar{x})$
Set $\Sigma_\phi^r = \{\forall \bar{x} \bigvee_i B_i(\bar{x}) \rightarrow Q_\phi(\bar{x}), \forall \bar{x}\, Q_\phi(\bar{x}) \rightarrow \bigvee_i B_i(\bar{x})\}$
Set $\Sigma_\phi^e :=$ AXINSTANCE(A_{Q_ϕ})
Set $\Sigma_\phi := \Sigma_\phi^r \cup \Sigma_\phi^e$.
return Σ_ϕ.

Algorithm 4. AXSECRETS($\mathcal{I}, Q_Z$)

returns constraints which axiomatize the possible secrets.

Set $\Phi_1 :=$ AXSOURCES($\mathcal{I}$).
// Set Φ_2 to the set of constraints over schema $\sigma_S \cup \{Q_Z\}$ which capture Q_Z:
Let Q_Z be the UCQ$^=$ query $Q_Z(\bar{x}) :- \bigvee_i B_i(\bar{x})$
Set $\Phi_2 := \{\forall \bar{x} \bigvee_i B_i(\bar{x}) \rightarrow Q_Z(\bar{x}), \forall \bar{x}\, Q_Z(\bar{x}) \rightarrow \bigvee_i B_i(\bar{x})\}$
return $\Phi_1 \cup \Phi_2$.

S_0 on the query Q_ϕ (the conclusion of constraint ϕ). AXONE employs the auxiliary procedure REWRITE(Σ, Q) which produces a rewriting R of Q in terms σ_S satisfying $R(S) = \mathrm{cert}_\Sigma^Q(S)$ for any S. Such an algorithm was provided in [9] for the case where Q is a Datalog program and $\Sigma \subseteq \mathrm{IC}(\mathrm{UCQ}^=)$ gives a local-as-view mapping. The extension to source-to-target constraints $\Sigma \subseteq \mathrm{IC}(\mathrm{UCQ}^=)$ is straightforward (see e.g., [23]). For the purposes of procedure AXSOURCES defined below, it is sufficient to have R axiomatizable by $\mathrm{IC}(\mathrm{UCQ}^=)$. However, to ensure decidability of implication on the result of AXSOURCES, we need R to be a UCQ$^=$ query. It is known from [9] that when Σ is source-to-target R is a UCQ$^=$ query.

Due to space limitations, we leave a detailed illustration of the algorithm for the extended version [19]. Here we only remark that in Example 1, the identity queries against the target schema turned out to be useless to the attacker, in contrast to the case of database publishing where the identity queries are the first step required to reveal the extent of the views. This is now explainable by our results: the identity queries are not the conclusions of the source-target constraints. On the other hand, query Q_ϕ in Example 2 constitutes the optimal attack. Our results also imply that when the integration system conforms to a global-as-view case, identity queries against the relations in the target schema lead to optimal attack strategies. We illustrate such an attack in Example 4 below (more examples can be found in [19]).

Example 4. Source S now conforms to schema $\{H(ssn, patient, doctor, ailment)\}$ where H is a history relation listing the social security number and name of patients, as well as the doctor who treated them for an ailment. The registration is given by constraint $\phi_1 = \forall s, p, d, a\ H(s, p, d, a) \rightarrow PD(p, d)$ which exports the projection of H on patient and doctor into PD. Note that this specification corresponds to the standard global-as-view registration given by view $PD(p, d) :- H(s, p, d, a)$. Since the projection of H is all that the source exports, the best an attacker can hope for is to retrieve its exact extent. But how should she query the system to this end? It is easy to show that the projection of H on patients and doctors coincides with the certain answers $\mathrm{cert}_\Sigma^{Q_{\phi_1}}(S)$ to the identity query Q_{ϕ_1} on table PD ($Q_{\phi_1}(p, d) :- PD(p, d)$). This is precisely the conclusion of ϕ_1.

5.1 Correctness

In this section we show (in Theorem 1) that AXSOURCES($\mathcal{I}$) axiomatizes precisely the set of possible sources (indistinguishable modulo *all* queries) and that AXSECRETS($\mathcal{I}$) axiomatizes precisely the set of possible secrets for an information integration system $\mathcal{I} := (\sigma_S, \sigma_T, \Sigma, S_0)$. In particular, the finite set $\mathcal{Q}_0$ of $\mathrm{UCQ}^=$ queries issued by procedure AXSOURCES and consisting of the conclusions of all constraints in Σ suffices to obtain as much information about the source and about the secret as is possible to obtain by querying $\mathcal{I}$. Therefore, among the attacks following the general strategy outlined in Section 3, the optimal algorithms OPTATTACKSOURCE and OPTATTACKSECRET are obtained from ATTACKSOURCE and ATTACKSECRET by replacing lines 1 and 2 with calls to respectively, AXSOURCES and AXSECRETS.

We define the *equivalence class* of S under the mapping given by Σ to be $[S]_\Sigma := \{S' : \forall T\ (S, T) \models \Sigma \text{ iff } (S', T) \models \Sigma\}$. That is, $[S]_\Sigma$ is the set of all sources which have the same possible targets as S. Clearly, given $\mathcal{I} := (\sigma_S, \sigma_T, \Sigma, S_0)$, the members of $[S_0]_\Sigma$ cannot be distinguished from the actual source S_0 or from each other by querying $\mathcal{I}$. Indeed, for any query Q and any $S \in [S_0]_\Sigma$, $\mathrm{cert}_\Sigma^Q(S) = \bigcap_{(S,T)\models\Sigma} Q(T) = \bigcap_{(S_0,T)\models\Sigma} Q(T) = \mathrm{cert}_\Sigma^Q(S_0)$. The following theorem shows that AXSOURCES($\mathcal{I}$) axiomatizes $[S_0]_\Sigma$ which is hence precisely the set of possible sources. It also shows that $\{Q_Z(S) : S \in [S_0]_\Sigma)\}$ is the set of possible secrets, axiomatized by AXSECRETS($\mathcal{I}, Q_Z$).

Theorem 1. *Given an information system* $\mathcal{I} := (\sigma_S, \sigma_T, \Sigma, S_0)$*:*

1. *The equivalence class* $[S_0]_\Sigma$ *is axiomatized by* $\text{AXSOURCES}(\mathcal{I})$ *and* $\text{AXSOURCES}(\mathcal{I}) \subseteq \text{IC}(\text{UCQ}^=)$.
2. *For any secret query* Q_Z*, the set* $\{Q_Z(S) : S \in [S_0]_\Sigma)\}$ *is axiomatized by* $\text{AXSECRETS}(\mathcal{I}, Q_Z)$ *and* $\text{AXSECRETS}(\mathcal{I}, Q_Z) \subseteq \text{IC}(\text{UCQ}^=)$.

This fundamental theorem allows us to (a) state the guarantees independently of the class of queries which the attacker is allowed to issue (we assume that the attacker can issue at least conjunctive queries), (b) outline an optimal attack strategy, and (c) provide algorithms for checking the guarantees.

As an immediate implication of Theorem 1, we obtain some interesting results for pure LAV and pure GAV integration, prefigurated by the discussion preceding Section 5.1: The source is always completely exposed in LAV information integration systems, since the optimal query is the view definition itself, for which the certain answers are exactly the tuples in the source. That is, Guarantee 1 always fails. Moreover, in this case the identity queries are useless, since they always return the empty set if the view registration contains at least one existential variable. The only queries required by an optimal attack against a GAV information integration system are the identity queries.

5.2 Complexity

We call a finite set of constraints Σ *convergent* if there exists a polynomial p such that for every $Q \in \text{UCQ}^=$, the result Q^Σ of chasing Q with Σ is the union of conjunctive queries $Q_1, \ldots, Q_k \in \text{CQ}^=$ satisfying $|Q_i| \leq p(|Q|)$ for $i \in \{1, \ldots, k\}$.

Theorem 2. *If* $\Sigma, \Sigma' \subseteq \text{IC}(\text{UCQ}^=)$ *are finite sets of convergent constraints and* $\phi \in \text{IC}(\text{UCQ}^=)$*, then checking whether* $\Sigma \not\models \phi$ *is decidable in* $\mathbf{\Pi_2^P}$ *in the combined size of* Σ *and* ϕ *and checking whether* $\Sigma \models \Sigma'$ *or* $\Sigma \equiv \Sigma'$ *is decidable in* $\mathbf{\Pi_2^P}$ *in the combined size of* Σ *and* Σ'*. Furthermore, if* ϕ *or* Σ' *have a single model, then the complexity is* $\mathbf{coNP}$.

Theorem 3. $\text{AXSOURCES}(\mathcal{I})$ *and* $\text{AXSECRETS}(\mathcal{I}, Q_Z)$ *each yield a set of convergent constraints, in time polynomial in the combined size of* S_0 *and of* $\text{REWRITE}(\Sigma, Q_\phi)$ *for every* $\phi \in \Sigma$.

Corollary 3. *Checking whether Guarantees 1 and 2 hold is in* $\mathbf{NP}$ *in the combined size of* S_0*,* $\text{REWRITE}(\Sigma, Q_\phi)$ *for every* $\phi \in \Sigma$*, and in the case of Guarantee 2,* Q_Z.

Corollary 4. *Checking whether Guarantees 3 and 4 hold is* $\mathbf{\Pi_2^P}$ *in the combined size of* S_0*,* $\text{REWRITE}(\Sigma, Q_\phi)$ *for every* $\phi \in \Sigma$*,* $\text{REWRITE}(\Sigma', Q_\phi)$ *for every* $\phi \in \Sigma'$*, and in the case of Guarantee 3,* Q_Z.

5.3 An Important Tractable Case

Our algorithms for checking guarantees are in general prohibitively expensive, as the $\mathbf{NP}$ and $\mathbf{\Pi_2^P}$ upper bounds (in Corollaries 3 and 4) include the size of the instance S_0. In this section we show that a practically relevant integration setting, which we call *tagged-union* integration, admits polynomial-time guarantee checking.

Definition 1. An integration is said to have *tagged-unions* if each target relation R has some attribute a such for each constraint $\phi \in \Sigma$ and each R-atom occurring in the conclusion Q_ϕ, a constant c_ϕ occurs in the attribute a such that $c_\phi \neq c'_\phi$ for all distinct $\phi, \phi' \in \Sigma$. No constant is needed if R appears in only one constraint.

All of our examples have tagged-union, since relation names are not shared across conclusions. The tagged-union restriction is quite realistic. While in a car dealership portal there will be many local dealers exporting their car ads into the same target ad relation, each dealer would likely tag the ad with the dealership name, address or phone number. Similarly for scenarios integrating any large community of vendors. Even for our medical example, one would expect various wards or hospitals to tag the published patient or doctor names with their affiliation. For example, consider a Honda and a Toyota dealer who integrate their private data into a brokerage portal (of target schema *deals*), using the tagged-union constraints ϕ_H, respectively ϕ_T:
$(\phi_H) \forall \bar{x}$ *myhondas*$(\bar{x}) \rightarrow$ *deals*("Honda", $\bar{x}$) $\quad$ $(\phi_T) \forall \bar{x}$ *mytoyotas*$(\bar{x}) \rightarrow$ *deals*("Toyota", $\bar{x}$).

Theorem 4. *In tagged-union integration systems, Guarantees 1 through 4 are decidable in polynomial time in the size of the source instance S_0.*

The $\mathbf{NP}$ and $\mathbf{\Pi_2^P}$ upper bounds of Corollaries 3 and 4 are now confined to the combined size of the constraints in Σ and the size of each REWRITE result, but these are data-independent.

5.4 Undecidability of Source-Independent Guarantees

We use the following undecidability results in our proofs below. A view V determines a query Q iff for all databases D_1, D_2, if $V(D_1) = V(D_2)$, then $Q(D_1) = Q(D_2)$. Checking whether V determines Q when $V, Q \in \mathrm{UCQ}$ is undecidable ([21]).

Theorem 5. *Checking Guarantee 2′ is undecidable.*

Theorem 6. *Checking Guarantee 4′ is undecidable.*

Since Guarantee 4′ is a particular case of Guarantee 3′ (for Q_Z the identity query over σ_S), we obtain the following corollary:

Corollary 5. *Checking Guarantee 3′ is undecidable.*

The decidability of Guarantee 1′ remains an open problem.

6 Related Work

One line of prior research focused on implementing access control in data publishing, i.e. allowing clients to see only those published views which they are authorized to. The techniques are based on cryptographically encoding the data (see [17] and references within). Our work is orthogonal to work on access control, as it helps data owners design the views (and more generally, mappings) such that attackers cannot breach privacy using only *authorized* accesses.

[2] introduces c-tables, a compact formalism for finitely representing large (and potentially infinite) sets of possible worlds, and shows $\mathbf{\Pi_2^P}$-complete data complexity for

checking that the sets of possible sources represented by two c-tables are the same. c-tables are not sufficiently expressive to model the set of possible sources corresponding to a materialized view instance. [13] introduces *database templates* to this end and shows how to compute them using the chase, but does not address the comparison of the sets of possible sources. We describe possible sources by a different formalism, namely a finite axiomatization.

[10] focuses on limiting privacy breaches in a scenario in which the aggregation of a set of private client data items is computed at the server. [3] takes aggregation into account and shows that exposing the result of counting queries allows the retrieval of an isomorphic copy of the structure of the database.

[20] takes a dual approach to ours (though in a closed world). While we use queries to specify the secret, [20] uses conjunctive query views to specify what may be seen by outsiders. In this setting, conjunctive client queries asked against the proprietary database are answered only if they have a rewriting using the allowable views.

Perfect Privacy. [18] addresses privacy in database publishing, i.e. in a closed-world, GAV scenario. The work pioneers the idea of specifying the secret as a conjunctive query over the base schema and checking the so-called *perfect privacy* guarantee. This consists in checking that a newly exported view does not modify the attacker's a priori belief about the secret. The attacker's belief is modeled as a probability distribution on the set of possible sources, with the simplifying assumption that the tuples in the secret answer are *independent events*. [8] adopts the notion of perfect privacy from [18] (still in a publishing, not integration scenario), but provides a more general formalization of attacker's beliefs by lifting the independence assumption on secret tuples. With this formalization, perfect privacy is shown in [8] to reduce to the preservation of the set of possible sources. Consequently, Guarantee 4 in this paper is the natural adaptation of perfect privacy from data publishing (in the flavor of [8]) to a data integration scenario.

Probabilistic Databases. One could envision quantitative privacy guarantees, e.g. by requiring a particular secret tuple to appear in no more than a fraction of the possible sources. Such approaches face the challenge of the set of possible sources being potentially infinite, in which case "counting" it must be defined carefully (see [6,7] for pioneering work in this direction, though in a database publishing setting).

7 Discussion

Privacy-preserving Updates. We can express guarantees corresponding to Guarantees 3 and 4, in which the mapping does not change (that is, $\Sigma = \Sigma'$), but the extent of the source does (S_0 is replaced by S_0'). The new guarantees would check that the possible sources, respectively secrets, do not change when S_0 is updated.

Conceptually, we can straightforwardly adapt our algorithms for checking Guarantees 3 and 4 to this new situation. All we need to do is call AXSOURCES (AXSECRETS) on the information system before and after the update (in the case of AXSECRETS, using the same Q_Z). Then we check that the obtained source (secret) axiomatizations imply each other. However, as such a test would have to be performed at run time, further work on efficient run-time algorithms is required towards a practical tool.

Target Constraints. We have modularized our privacy algorithms to work in the presence of arbitrary constraints on the target schema, provided that (i) the integration system can return the certain answers in this case, and (ii) there exists an algorithm REWRITE(Σ, Q) which produces a rewriting of Q in terms of σ_S returning the certain answers of Q on any source, and (iii) REWRITE(Σ, Q) returns a UCQ$^=$ query. It is known from [9] that when there are no target constraints, REWRITE(Σ, Q) returns a UCQ$^=$ query, but returns a recursive Datalog program when the target constraints are full dependencies. In this case, Theorem 1 still holds but the obtained constraints are not convergent and therefore Theorem 3 does not apply so we can not make any claims on the complexity of checking these guarantees. [9] provides no rewriting for more general target constraints.

References

1. S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison-Wesley, 1995.
2. S. Abiteboul, P. Kanellakis, and G. Grahne. On the representation and querying of sets of possible worlds. *Theoretical Computer Science*, 78:159–187, 1991.
3. M. Bielecki and J. V. den Bussche. Database interrogation using conjunctive queries. In *ICDT*, pages 259–269, 2003.
4. A. Cali, D. Calvanese, G. D. Giacomo, and M. Lenzerini. Data integration under integrity constraints. In *CAiSE*, 2002.
5. A. Cali, G. D. Giacomo, and M. Lenzerini. Models of information integration: Turning local-as-view into global-as-view. In *FMII*, 2001.
6. N. N. Dalvi, G. Miklau, and D. Suciu. Asymptotic conditional probabilities for conjunctive queries. In *ICDT*, 2005.
7. N. Dalvi, D. Suciu. Answering queries from statistics and probabilistic views. *VLDB*, 2005.
8. A. Deutsch and Y. Papakonstantinou. Privacy in database publishing. In *ICDT*, 2005.
9. O. Duschka, M. Genesereth, and A. Levy. Recursive query plans for data integration. *Journal of Logic Programming*, 43(1):49–73, 2000.
10. A. Evfimievski, J. Gehrke, and R. Srikant. Limiting privacy breaches in privacy preserving data mining. In *PODS*, 2003.
11. R. Fagin, P. Kolaitis, R. Miller, and L. Popa. Data exchange: Semantics and query answering. In *ICDT*, 2003.
12. M. Friedman, A. Levy, and T. Millstein. Navigational plans for data integration. In 16^{th} *National Conference on Artificial Intelligence (AAAI),*, 1999.
13. G. Grahne and A. O. Mendelzon. Tableau techniques for querying information sources through global schemas. In *ICDT*, 1999.
14. A. Halevy. Logic-based techniques in data integration. In *Logic Based Artificial Intelligence*, 2000.
15. C. Koch. Query rewriting with symmetric constraints. In *FoIKS*, 2002.
16. M. Lenzerini. Data integration: A theoretical perspective. In *PODS*, 2002.
17. G. Miklau, D. Suciu. Controlling access to published data using cryptography. *VLDB*, 2003.
18. G. Miklau and D. Suciu. A formal analysis of information disclosure in data exchange. In *SIGMOD Conference*, 2004.
19. A. Nash and A. Deutsch. Privacy in GLAV information integration. Technical Report CS2006-0869, University of California San Diego, 2006. http://db.ucsd.edu/people/alin.
20. S. Rizvi, A. O. Mendelzon, S. Sudarshan, and P. Roy. Extending query rewriting techniques for fine-grained access control. In *SIGMOD Conference*, 2004.
21. L. Segoufin and V. Vianu. Views and queries: Determinacy and rewriting. In *PODS*, 2005.
22. J. D. Ullman. Information integration using logical views. In *ICDT*, 1997.
23. C. Yu, L. Popa. Constraint-based XML query rewriting for data integration. *SIGMOD*, 2004.