Towards Fine-Grained Data Access Control Through Active Peer Probing

Yael Amsterdamer
Bar-Ilan University
first.last@biu.ac.il

Osnat Drien
Bar-Ilan University
first.last@live.biu.ac.il

Abstract

Data is routinely being shared online by peers, for instance in business transactions, social activities and others. This data, in turn, is often transferred, processed and combined through complex querying and analytics. This raises questions such as the following: who owns the derived data? With whom and for what purpose may it be published? If consent is required for its dissemination, whose consent should be obtained?

The related topics of data sharing, privacy and access control have been extensively studied, but uniquely our focus here is not on data management with known policies but rather on the active probing of peers to ask for their consent. Active probing has the potential to allow finer-grained access control, where it is unreasonable to expect data owners to publish their full policies, defined for all possible sharing scenarios. They may not even have a clear view of their own policies, before asked whether they are willing to share data with a specific third party.

This short paper informally introduces and motivates this new problem. It further identifies interesting connections to two distinct areas: data provenance, which captures the way output data are derived from inputs, and Boolean evaluation, which focuses on effective strategies to probe hidden Boolean values for evaluating a formula. As we shall demonstrate, the composition of these two areas in the context of this problem yields intriguing avenues for further research.

1 Introduction

When peers share data – on social networks, for event planning or in business collaborations – access control is often a concern. Data may be re-shared and used in analysis that combines input from multiple sources, thereby making it difficult to correctly decide access permissions. For example, the data of Alice's recruitment agency may consist of personal data of job seekers, confidential data on companies and internal information on collaborators. Now assume Alice wants to re-share parts of this data with a collaborating agency. In common data sharing platforms, peers that have originally contributed data to Alice either have no control over the re-sharing of their data, or give a broad consent to re-sharing within some group (e.g., for non-commercial purposes), or disallow re-sharing altogether [1]. However, a finer-grained approach may allow re-sharing more data without compromising the preferences of data owners. For instance, Bob, a collaborating agent, may not have agreed that his data is shared with

every third party; but if Alice actively asks for his permission to share data with Carol, a specific mutual collaborator, he may agree. Next, Alice may compute statistics based on combining and analyzing the data provided from many sources. May she share the derived data (analysis results) with a third party? Who should be probed for permissions in this case?

The related topics of data sharing, privacy and access control have been extensively studied (see a brief discussion below), but uniquely our focus here is not on data management with known policies but rather on the *active probing of peers* to ask for their consent, to achieve fine-grained access control. The peer answers may either be given manually (by one of Bob's employees), or (semi)-automatically (by Bob's servers). In either case, probing requires resources and reveals parts of Bob's proprietary policy, and therefore should be minimized. As we shall demonstrate, this active setting, coupled with reasoning performed over the data, leads to novel computational problems. Informally, we introduce the following high-level problem:

We are given a database whose tuples have been contributed by multiple peers, and a query (in some language) over the database. Access control policies with respect to the input *tuples are (completely or partially) unknown, but we may* probe *data owners to ask about them. Our goal is to decide whether* the query output *(or a subset thereof) may be shared with a third party, while minimizing the number of probes (or otherwise optimizing with respect to a related target).*

To continue our above example, assume that an agency owned by Carol seeks information on companies in Pennsylvania in which positions were successfully found for graduates of environmental studies. The agency of Alice, a collaborator of Carol, may have such data at hand, and the information needs of Carol may be captured by an SPJU query on this data – but may the query results be shared? For instance, an output value "PennSolarExperts Inc." may be the result of joining several tuples involving data on (1) the PennSolarExperts company; (2) three environmental studies graduates who found positions in this company, and (3) the agency owned by Bob who collaborated with Alice in finding positions for some of the candidates (and then projecting on the company name). In this case, to share the query result with Carol, Alice may need the consent of the company, at least one of the assigned workers, and Bob. Importantly, Alice may not know in advance whether the company, workers or Bob agree that computation results based on their data are shared with Carol; they could have shared their data with Alice without giving her permissions to share it with a third party. In this case, Alice would need to actively probe the peers, but what is her best "strategy" for probing (i.e. who should she probe and in what order)?

In this short paper we introduce and motivate this new problem in a high-level manner, through an example. We further outline a promising approach for a solution, based on two seemingly disjoint areas of research: Data Provenance and Boolean Evaluation. We believe that the problem, and our line of attack, are worthy of in-depth investigation.

Related Work We conclude this Introduction with an overview of related work. The theory and practice of Access control have been extensively studied in different contexts, including social networks (e.g., [3, 4, 5, 6, 7, 8, 1]), distributed systems (e.g., [9, 10]) cloud services (e.g., [11, 12, 13]), Web applications

Companies

cid	name
11	PennSolarExperts Ltd.

Vacancies

vid	cid	position	amount
111	11	analyst	3
112	11	supervisor	1

JobSeekers

sid	name	education	agency
1	David	Env. studies	Bob
2	Ellen	Env. studies	Bob
3	Frank	Env. studies	Alice
4	Georgia	Env. studies	Bob

Assignment

sid	vid	status	agency
1	111	hired	Bob
2	112	rejected	Alice
2	111	hired	Bob
3	111	rejected	Alice
4	112	hired	Alice

Table 1: Example database of Alice's recruitment agency.

(e.g., [14, 15]) , databases (e.g., [16, 17, 18]), and many other areas. With respect to these works, our novelty is in focusing on a setting where access policies may be unknown or undetermined in advance, which requires active probing of involved peers to obtain permissions. Fine-grained access control policies may be too large or complex to be specified by a client, if, e.g., the permission for every peer and action must be specified. To assist clients in specifying policies, previous work has considered (semi-)automatic computation of access control policies. This includes the computation of policies based on example permissions [19]; evaluating the credibility of peers [6]; mining or interactively defining user roles [20, 21]; using semantically-rich languages to compactly capture real-life factors on policy definition [22]; and using game-theoretic considerations in defining policies with respect to risk minimization [23]. These works are complementary to ours, in the sense that we probe peers as a "black box": peer answers may be obtained manually or predefined using methods such as above.

2 Model, via an Example

We next outline a preliminary model for the problem, illustrated informally via an example.

We are given a Relational Database[1] where each tuple is annotated with a label which we refer to as *concept*, taken from a set of concepts $\mathcal{C}$. Each concept "belongs" to a single owner out of a set of peers $\mathcal{P}$. We will refer to such annotated database as a *shared database*.

Example 2.1 *Table 1 outlines a simple DB for the recruitment agency example described above, consisting of details of companies and job vacancies in these companies, with the type of position and the number of open positions; job seekers with their name, education and agency to which they have applied;*

[1] For brevity, we demonstrate the problem and our approach in a relational setting and for tuple-level access control; it applies similarly, with some extensions, to semi-structured data and value-level access control.

and the assignment of seekers to vacancies, including the status of the assignment and the agency responsible for matching. The annotations here can be set to reflect the row (table+key) and the owner of the data. In this case, we assume for simplicity that company and vacancy data is owned by the company, and job seeker and vacancy data is owned by the relevant agency as the seekers' representatives. The annotation for the first row in Companies could for instance be `Companies11PennSolar` *and the first row of Assignment could be* `Assignment1-111Bob`.

The premise is that there is a hidden truth value to whether or not we are allowed to share each concept with a specific third party (or publish it in public, etc.); this truth value is known only to the concept owner.

Example 2.2 *Recall the database in Table 1, and assume that Alice wishes to share the* **JobSeekers** *table with Carol. In this case, Alice's agency is the owner of the third tuple, and thus can check whether she can share the data - e.g., if Frank agreed in his contract with the agency to share data with third parties. The yes/no answer would translate to a valuation of* true/false *respectively to the Boolean variable captured by the annotation* `JobSeekers3Alice`. *The other tuples correspond to job seekers recruited by Bob's agency, hence we assume that when asked by Alice, Bob's agency can answer yes/no to the sharing request, again translated to a Boolean valuation.*

Some concepts may be associated with a semantic interpretation, in which case the hidden truth values are constrained. For instance, in the database from Figure 1, we can assume that access permission to a row in **Vacancies** implies access permission to the relevant company's row in **Companies**. To capture such constraints, we use taxonomies.

Next, instead of sharing the data as-is, we consider a *query* executed on the database to perform some analytics and the sharing of its results. We consider "query" as a broad term here, and variants of the problem will focus on different query languages (e.g., relational SPJU, Datalog, etc.).

```
SELECT DISTINCT c.name
FROM   Companies c,
       JobSeekers j,
       Vacancies v,
       Assignments a
WHERE  c.cid = v.cid AND
       v.vid = a.vid AND
       a.status = 'hired' AND
       a.sid = s.sid AND
       s.education = 'Env. studies'
```

Figure 1: Query over the example database

Example 2.3 *Recall our running example and now assume that Alice wishes to share with Carol the names of companies where environmental studies graduates have successfully found jobs. To this end, she runs the query in Figure 1 on the*

database in Table 1. In this simplified example the answer is the single company in the database - "PennSolarExperts Ltd.", where David, Ellen and Georgia have been hired.

The question is then: given an (annotated) database and a query, are we allowed to share the result?

Example 2.4 *Sharing the single result value returned by the example query, "PennSolarExperts Ltd.", requires the company's consent, as the owner of the relevant tuple. Furthermore, sharing the result may reveal information about other tuples participating in the derivation. For instance, if there are few environmental studies graduates who work in PennSolarExperts, sharing the result with Carol would reveal personal information about them, including that at least one of them was recruited by Alice's agency or her collaborators (and in turn, that this person belongs to the type of job applications at which Alice and her collaborators specialize, e.g., interns, part-time positions, etc.). Beyond our example, peers can ask queries, e.g. Boolean, whose result does not contain any tuple cell, and yet may reveal the existence of other tuples. We shall therefore consider to* which *tuples used in the derivation permissions are needed. In our case, intuitively, it is sufficient to have permissions to the data involved in* one *relevant job assignment, since the existence of additional assignments does not change the result. To share e.g., Georgia's assignment details, Alice needs permission to share all the tuples jointly involved in it in addition to the company's tuple – tuple 4 in* **JobSeekers***, vacancy 112 and the relevant assignment. To obtain these permissions, Alice needs to probe Bob Bob, the owner of the relevant* **JobSeekers** *tuple, and PennSolarExperts for the* **Companies** *and* **Vacancies** *tuples. The owner of the assignment itself is Alice's agency, which means she has the information of whether this tuple can be shared or not. If only one of the four aforementioned tuples cannot be shared, Georgia's assignment cannot be shared.*

Since access control policies with respect to the individual concepts are "hidden", namely known only to owners, the tool that we have for deciding whether or not a result may be shared is to pose questions or *probes* to the owners of relevant data items. The goal is then to optimally select probes in order to discover whether sharing is permitted.

Example 2.5 *In our running example, 9 tuples contributed in some way (to be formalized below through the notion of provenance) to the result of the example query. If the only tuple owned by Alice (the assignment of job seeker 4) can be shared, we are left with 8 tuples. If we ask PennSolarExperts whether their company's details can be shared and get a negative answer, we know that the data cannot be shared and there is no need to ask further questions. As another example, recall that we assumed that vacancies data can only be shared if the company's data can be shared. Then assume we get PennSolarSystem's permission to share data about vacancy 112 (and hence also the company details) and Bob's permissions to share Georgia's details – we obtain that the query result can be shared having used only 2 probes.*

Naturally, the number of questions that will be asked in practice depends on the answers received, which are unknown in advance. The goal is to design a

strategy for choosing which questions to pose and in what order, where multiple variants of the problem could be of interest. These variants may be based on axes such as the query language expressiveness (e.g. Conjunctive Queries, Datalog, etc.); the optimization goal (e.g. minimizing the number of questions or maximizing the number of shareable results for a given "budget" of questions); optimizing for the worst or expected case (with respect to the peer answers); selecting probes in advance or incrementally; and restricting the per-peer probes or optimizing the overall number of probes.

3 Towards a Solution

Having informally introduced the problem, we next outline a preliminary approach for a solution, combining multiple areas of previous work.

Provenance. We are interested in whether or not we may share *derived* data, whereas (hidden) access control policies are defined with respect to the original, *atomic* data items. The propagation of meta-data from atomic data items to the query results related to them has been studied under the prism of *provenance* [16, 24, 17] (and previously, c-tables [25]). In our case, we may use provenance to compute expressions capturing the access control of derived data, in terms of the concepts annotating the input.

Example 3.1 *Recall the query in our running example from Figure 1. Using c-tables [25] (or alternatively Boolean provenance [16, 24]), we can compute a Boolean expression reflecting the dependencies of access control credentials to the output on permissions to view relevant input tuples.*

$$\begin{aligned}&\texttt{Companies11PennSolar}\wedge\\ &(\texttt{Vacancies111PennSolar}\wedge((\texttt{Assignment1-111Bob}\wedge\texttt{JobSeekers1Bob})\\ &\qquad\qquad\vee(\texttt{Assignment2-111Bob}\wedge\texttt{JobSeekers2Bob}))\\ &\vee(\texttt{Vacancies112PennSolar}\wedge\texttt{Assignment4-112Alice}\wedge\texttt{JobSeekers4Bob}))\end{aligned}$$

The formula uses the access control concepts of the relevant tuples as variables. Indeed, it matches the intuition of Example 2.5 on how probe answers may affect the final decision: if PennSolarExperts refuse sharing their company details with Carol, `Companies11PennSolar` *will be evaluated to* false*, and the truth value of the entire formula will be* false*. Alternatively, if we know that* `Vacancies112PennSolar`, `Companies11PennSolar`, `Assignment4-112Alice` *and* `JobSeekers4Bob` *evaluate to* true*, the entire expression evaluates to* true*.*

(Boolean) provenance constructions such as the one exemplified above have been developed for different query languages and formalisms, and the shape of the resulting Boolean expression depends on the formalism for which provenance is tracked, which in turn may affect the probe selection process. For instance, if we restrict attention to Union of Conjunctive Queries, then provenance of each output tuple may be represented in Disjunctive Normal Form of polynomial size with respect to the input database size [25]; negation is needed for queries with relational difference [26]; for Datalog, a polynomial size representation is possible in the worst case only if we resort to Boolean circuits [27]; etc.

(Incremental) Boolean Evaluation. Given Boolean provenance formulas over data items, had we known whether they are authorized for publication, we could simply assign true/false to the corresponding variables in the formulas and decide whether the derived data could be shared. However, policies of peers may be unknown or undetermined, therefore we probe peers to obtain them. We now consider the optimization problem of selecting the best variables to observe next.

To illustrate, we next outline our preliminary solution for the following setting:

- Relational SPJUD queries (select-project-join-union-difference),
- Minimizing the number of questions
- Optimizing the expected case
- Selecting questions incrementally
- Considering either the number of questions overall or per peer
- Assuming an equal cost for all the questions/peers and given answer prior probabilities.

In this case, as explained above, output tuples would be annotated by Boolean expressions (computed via [25]). We then explore results on Boolean evaluation for such expression, and may leverage them to show that the problem is NP-hard, via [28, 29]. In contrast, previous work has studied optimal solutions for restricted cases, heuristics and approximations (see, e.g., [28, 30] for a survey). In particular, in [31] we have adapted and extended an approximate solution by [32], as we next briefly outline.

Denote the set of output tuple annotations by $\mathbf{E}$. For each Boolean formula $e_i \in \mathbf{E}$ we define two *utility* functions $g_0^i, g_1^i : \{1, 0, *\}^{|\mathcal{C}|} \to \mathbb{R}^+$, where $\mathcal{C}$ is the set of variables in the Boolean expressions and each entry represents either a value assignment to a variable or no assignment ($*$). $g_0^i(\vec{c})$ and $g_1^i(\vec{c})$ are respectively the number of terms (conjunctions) set to 0 in the DNF form of e_i, and the number of clauses (disjunctions) set to 1 in the CNF form of e_i by the partial assignment $\vec{c}$. Denote by m_i and l_i the number of terms and clauses in e_i respectively. The utility function $g^i(\vec{c}) = m_i l_i - (m_i - g_0^i(\vec{c}))(l_i - g_1^i(\vec{c}))$ reaches its maximum, $m_i l_i$, when $g_0^i(\vec{c}) = m_i$ or $g_1^i(\vec{c}) = l_i$, i.e., e_i is evaluated.

To minimize the *overall* number of questions a greedy algorithm is then used. The algorithm repeatedly selects the unknown variable c_j whose probe maximizes the expected value (over the possible answers) of g^i. Some properties of g^i (monotonicity, submodularity) guarantee that the expected number of questions is within a factor of $\ln(m_i l_i) + 1$ from the optimum.

Next, we need to simultaneously evaluate all the formulas in $\mathbf{E}$, which may be done by defining $g(\vec{c}) = \sum_{e_i \in E} g^i(\vec{c})$. g is a function that reaches its maximum, $\sum_{e_i \in \mathbf{E}} m_i l_i$, when all the expressions are evaluated. By similar analysis to that of g^i, we get that the solution is a $\ln(\sum_{e_i \in \mathbf{E}} m_i l_i) + 1$ approximation of the optimum.

The above adaptation is an example of a rather direct combination of results from the provenance and Boolean evaluation literature; our setting suggest further novel algorithmic developments. For instance, in [31] we have extended

the solution to compute batches of probes rather than single probes and to account for constraints imposed by a taxonomy over the items (i.e., implication constraints between access rights), showing that we still achieve the same approximation bound. We also study an incremental setting where the client is allowed to terminate the process at any point and share partial results via "safe views" – views of the results that are known to be safe for sharing.

Another example for a variant that is not accounted for by previous results is one that targets the minimization of the probes *per peer*, in this case, minimizing. For that, we keep track of the number of probes per peer $p \in \mathcal{P}$ (denoted by probes(p)). At each step, let $\mathcal{C}^*$ be the set of variables still unknown. When selecting the next probe, we consider only variables in $\{c \in \mathcal{C}^* \mid \text{probes}(\text{owns}(c)) = \min_{p' \in \mathcal{P}^*} \text{probes}(p') \wedge \mathrm{E}[g(\vec{c})] > 0\}$, i.e., that are owned by least-probed peers and whose utility is non-zero.

Our preliminary results are encouraging and suggest that the combination of provenance constructions and Boolean evaluation methods is a promising direction towards studying our problem complexity and designing efficient algorithms. There are still many challenges to be overcome in this space, both theoretical and practical: for instance, can we achieve better guarantees by restricting the query language? What guarantees can we obtain for more expressive languages? For instance, what if we have aggregates, or recursion and then Boolean circuits rather than formulas? What if we have a "budget" for the number of probes to be used? How can we estimate the probe answer probabilities, in light of constraints defined by the taxonomy, peer trust, and accumulated probe answers? The interplay between the choice of query language/provenance model, optimization goal and constraints leads to many intriguing computational questions which will be central to research on this problem.

4 Conclusion

This paper has advocated the study of access control management in a setting where peers are actively probed to ask for their permissions. Our main insight is that the problem may be addressed by computing Boolean provenance for query results, and treating the Boolean expression as input to active Boolean evaluation algorithms. We believe that this high-level approach paves the way to exciting research possibilities at the intersection of these two seemingly unrelated areas.

Acknowledgements

This work was funded in part by the BIU Center for Research in Applied Cryptography and Cyber Security in conjunction with the Israel National Cyber Bureau in the Prime Ministers Office, and by the Israel Science Foundation (grant No. 1157/16).

References

[1] L. Yu, S. M. Motipalli, D. Lee, P. Liu, H. Xu, Q. Liu, J. Tan, and B. Luo, "My friend leaks my privacy: Modeling and analyzing privacy in social

networks," in *SACMAT*, 2018.

[2] T. Abdessalem and I. B. Dhia, "A reachability-based access control model for online social networks," in *DBSocial*, 2011.

[3] A. K. Abdulla and S. Bakiras, "HITC: data privacy in online social networks with fine-grained access control," in *SACMAT*, 2019.

[4] Y. Cheng, J. Park, and R. S. Sandhu, "An access control model for online social networks using user-to-user relationships," *IEEE Trans. Dependable Sec. Comput.*, vol. 13, no. 4, 2016.

[5] M. Cramer, J. Pang, and Y. Zhang, "A logical approach to restricting access in online social networks," in *SACMAT*, 2015.

[6] E. Gudes and N. Voloch, "An information-flow control model for online social networks based on user-attribute credibility and connection-strength factors," in *CSCML*, 2018.

[7] P. Ilia, B. Carminati, E. Ferrari, P. Fragopoulou, and S. Ioannidis, "SAMPAC: socially-aware collaborative multi-party access control," in *CODASPY*, 2017.

[8] N. C. Rathore, P. Shaw, and S. Tripathy, "Collaborative access control mechanism for online social networks," in *ICDCIT*, 2016, pp. 142–147.

[9] S. Abiteboul, P. Bourhis, and V. Vianu, "A formal study of collaborative access control in distributed datalog," in *ICDT*, 2016.

[10] V. Z. Moffitt, J. Stoyanovich, S. Abiteboul, and G. Miklau, "Collaborative access control in webdamlog," in *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, Melbourne, Victoria, Australia, May 31 - June 4, 2015*, 2015.

[11] A. Bates, B. Mood, M. Valafar, and K. R. B. Butler, "Towards secure provenance-based access control in cloud environments," in *(CODASPY)*, 2013.

[12] P. Derbeko, S. Dolev, E. Gudes, and S. Sharma, "Security and privacy aspects in mapreduce on clouds: A survey," *Computer Science Review*, vol. 20, pp. 1–28, 2016. [Online]. Available: https://doi.org/10.1016/j.cosrev.2016.05.001

[13] M. G. Solomon, V. S. Sunderam, and L. Xiong, "Towards secure cloud database with fine-grained access control," in *DBSec*, 2014.

[14] H. Qunoo and M. Ryan, "Modelling dynamic access control policies for web-based collaborative systems," in *DBSec*, 2010.

[15] G. S. Tuncay, S. Demetriou, and C. A. Gunter, "Draco: A system for uniform and fine-grained access control for web code on android," in *CCS*, 2016.

[16] J. N. Foster, T. J. Green, and V. Tannen, "Annotated XML: queries and provenance," in *PODS*, 2008.

[17] G. Karvounarakis, Z. G. Ives, and V. Tannen, "Querying data provenance," in *SIGMOD*, 2010.

[18] A. Roichman and E. Gudes, "Fine-grained access control to web databases," in *SACMAT*, 2007.

[19] G. P. Cheek and M. Shehab, "Policy-by-example for online social networks," in *SACMAT*, 2012.

[20] N. Gal-Oz, Y. Gonen, and E. Gudes, "Mining meaningful and rare roles from web application usage patterns," *Computers & Security*, vol. 82, 2019.

[21] T. Wang, M. Srivatsa, and L. Liu, "Fine-grained access control of personal data," in *SACMAT*, 2012.

[22] A. Masoumzadeh and J. B. D. Joshi, "OSNAC: an ontology-based access control model for social networking systems," in *PASSAT*, 2010.

[23] H. Hu, G. Ahn, Z. Zhao, and D. Yang, "Game theoretic analysis of multi-party access control in online social networks," in *SACMAT*, 2014.

[24] T. J. Green, G. Karvounarakis, and V. Tannen, "Provenance semirings," in *PODS*, 2007.

[25] T. Imielinski and W. L. Jr., "Incomplete information in relational databases," *J. ACM*, vol. 31, no. 4, 1984.

[26] Y. Amsterdamer, D. Deutch, and V. Tannen, "On the limitations of provenance for queries with difference," in *TaPP'11*, 2011.

[27] D. Deutch, T. Milo, S. Roy, and V. Tannen, "Circuits for datalog provenance," in *ICDT*, 2014.

[28] S. R. Allen, L. Hellerstein, D. Kletenik, and T. Ünlüyurt, "Evaluation of monotone DNF formulas," *Algorithmica*, vol. 77, no. 3, 2017.

[29] L. A. Cox, Q. Yuping, and W. Kuehner, "Heuristic least-cost computation of discrete classification functions with uncertain argument values," *Annals of Operations research*, vol. 21, no. 1, 1989.

[30] T. Ünlüyurt, "Sequential testing of complex systems: a review," *Discrete Applied Mathematics*, vol. 142, no. 1-3, 2004.

[31] Y. Amsterdamer and O. Drein, "PePPer: Fine-grained personal access control via peer probing (demo)," in *ICDE*, 2019.

[32] A. Deshpande, L. Hellerstein, and D. Kletenik, "Approximation algorithms for stochastic Boolean function evaluation and stochastic submodular set cover," in *SODA*, 2014.