Towards a Distributed Inference Detection System in a Multi-Database Context

Sad Rafik, Paul Lachat, Nadia Bennani, Veronika Rehn-Sonigo

Towards a Distributed Inference Detection System in a Multi-Database Context

Sad Rafik[1], Paul Lachat[2,3], Nadia Bennani[2], Veronika Rehn-Sonigo[4]

1 Università degli Studi di Milano, Milano, Italy
2 LIRIS, INSA Lyon, Villeurbanne, France.
3 DIMIS, University of Passau, Passau, Germany.
4 FEMTO-ST Institue, University of Bourgogne Franche-Comté, Besançon, France.

Abstract

The omnipresence of services offered by diverse applications leads customers to share more and more personal data, among which some are sensitive. Dishonest entities perform inference attacks by querying non-sensitive data in order to deduce the stored sensitive data. Detecting those attacks is still an open problem in a setting where a dishonest entity has access to distinct data controllers' databases containing data collected from the same customer. This problem has been addressed considering a centralized detection system. However, this approach is limited because of this centralized nature where the system protects the customers' privacy at the expense of the data controllers' privacy. Hence, we propose in this article the description of a distributed architecture to detect inference attacks in a multi-database context, while preserving the privacy of both the applications and the customers.

1 Introduction

Nowadays, individuals are used to share both non-sensitive and, more carefully, sensitive personal data with distinct applications, in order to benefit from a multitude of services. The data controllers are collecting, storing, and sharing a part of this information with external entities based on the individuals' consent [Ger+18]. Hence, access control systems are leveraged in order to protect the sensitive data against unauthorized direct access. Yet, dishonest entities perform inference attacks on sensitive data by querying non-sensitive data, for which they have an authorized access, to deduce the sensitive data stored in the databases [FJ02]. Inference detection systems (InfDSs) have been proposed in the literature with the aim of protecting a single personal database against inference attacks [CC08]; [GMB17]; [TFE10]; [CM03]. However, this issue is still open in a setting where a dishonest entity has access to databases managed by distinct controllers containing data collected from the same customers.

To illustrate this threat, let us consider an example in which a customer creates an account on two distinct applications: *Train* and *Flight* allow to buy train and plane tickets, respectively. At each account creation, the customer is invited to share some of their personal data (e.g., name, age, sex, and so on). Locally, the services protect the customers' privacy thanks to both an access control system and an InfDS. While respecting the consent of customers, the services share the non-sensitive data with authorized entities. In this setting, when a dishonest but authorized entity tries to gather a subset of non-sensitive data related to a targeted customer in a single application, the InfDS of this application will detect and prevent any queries leading to an inference attack. However, by leveraging the fact that the targeted customer has subscribed to the two services, and that the two services are asking for a common subset of personal data, the dishonest entity queries a subset of non-sensitive data from each application so that an inference attack becomes possible once combining the two subsets of personal data. In this case, the inference attack is not detected locally by the InfDSs since the queried subsets are not harmful when considered separately. This example illustrates the threat of inference attacks exploiting the *distributed dependency strategy* (DDS) [WB10].

In order to cope with this kind of inference attacks an InfDS must: (i) keep track of the information that each entity obtains from all the collaborating applications; (ii) be able to identify the semantic similarities between differently formatted data managed by distinct applications. This observation has motivated a previous work by Lachat et al. [LRB20] to address this problem by making the applications collaborate with a centralized InfDS. The work presented in [LRB20] is based on the two knowledge representations proposed by Chen et al. [CC08]: the *Semantic Inference Model* (SIM) which represents the probabilistic dependencies of a database at schema level, and the *Semantic Instance Graph* (SIG) which reflects the dependencies expressed by the SIM at instance level. We proposed the *Global Instance Graph* (GIG) which is composed of all the SIGs of the collaborating applications, extended with a new kind of links (i.e., the semantic links) representing semantic similarities between attributes of distinct databases. In order to build the GIG, the centralized detection system requires to have access to the schema of every database in order to identify the similarities, as well as to the instances from distinct databases in order to incorporate those similarities in between the SIGs.

The solution lacks privacy for both individuals' data and databases schema shared with the centralized InfDS, which discourages the applications to collaborate with such a system. Therefore, to increase the acceptance of such a detection system, it should be able to detect inference attacks without centralizing and having access to the schema and the instances in clear. Hence, a distributed InfDS must satisfy the following requirements in this multi-database context: (i) The GIG has to be build in a distributed way preventing any schema and data disclosure. (ii) The knowledge that an entity obtains by issuing a query has to be propagated to all the collaborating applications to correctly keep track of the queried information, while limiting the impact on the query answer time. Thus, such a system will provide the required warranties in order to increase the

motivation of applications to participate.

The main contribution of this paper is the proposition of a first solution towards the distributed detection of inference attacks, which protects both customers' and applications' privacy based on Chen et al. solution.

This paper is organized as follows: Section 2 presents the limitation of inference detection systems w.r.t the described issue. Section 3 presents the SIM and the SIG, the two models on which is based Chen et al. solution. Section 4 describes our solution for the distributed detection of inference attacks. Section 5 presents the conclusion and the future research directions.

2 Related works

In this section, we review the related work in the field of inference detection and raise their limitations to overcome the issues listed in Section 1. Most of the proposed solutions focus on inference attacks on a single database.

Chen et al. [CC08] and Guarnieri et al. [GMB17] propose to tackle inference attacks by building a model which represents probabilistic inference channels. Guarnieri et al. propose a system where one module acts as a policy decision point whereas the other checks inference attempts. The two solutions address inference attack detection with the assumption of protecting a single static personal database. The solution of Guarnieri et al. works under the assumption that only closed queries are issued to the database.

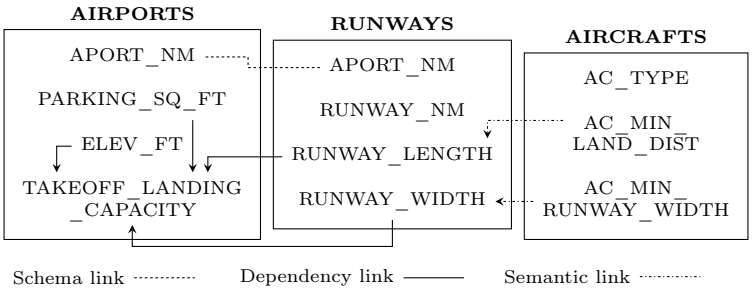
The system presented by Toland et al. [TFE10] models the Functional Dependencies (FDs) within a database to compute the disclosed knowledge each time a query is issued. In their work the FDs are limited to logical dependencies. This work is the only one that considers database updates (i.e., tuple updates, deletion or insertion), by storing the most recent updates in the query history log. This solution however focuses on protecting a single personal database.

Similarly to Chen et al., Chang et al. [CM03] propose a mechanism which reasons on the probabilistic dependencies among attributes, but for a distributed personal database. The authors are focusing on a distributed database, the solution is not suitable to model the semantic similarities between databases owned by distinct applications, thus having different schemas. Lachat et al. [LRB20] extend the work of Chen et al. in order to detect inference attacks exploiting multi-database inference channels using data linkage techniques. While Lachat et al. is the only work which, to the best of our knowledge, focuses on protecting multiple databases, it leads to the limits described in Section 1.

3 Detection in presence of a single database

The contribution presented in this article relies on two models proposed by Chen et al. [CC08]. In this section, we provide a short description of the *Semantic Inference Model* (SIM) and the *Semantic Instance Graph* (SIG). The SIM represents existing probabilistic dependencies at the schema level. This

(a) Semantic Inference Model (SIM).



(b) Semantic Instance Graph (SIG). The sensitive attribute is in orange (LAX_TAKEOFF...) and the queried attributes are in red (C5_MIN_LAND..., C5_MIN_RW..., LAX_ELEV...).

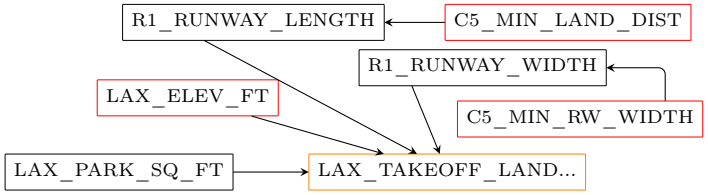


Figure 1: Example of a SIM and a SIG. Partial reproduction of Fig.1 in [LRB20].

model is based on the *Probabilistic Relational Model* (PRM), which itself relies on Bayesian networks. According to [Get+01], the PRM "[...] allows the properties of an object to depend probabilistically both on other properties of that object and on properties of *related* objects". This model is composed of a skeleton and parameters. The skeleton describes the direct influences between *parents* attributes and some *child* attribute (i.e., a DAG). The parameters define the *Conditional Probability Distribution* (CPD) between a child attribute and its parents. The CPDs represent the probabilistic distribution of a given snapshot of the stored data. Chen et al. define three kinds of links for the SIM: (i) dependency link: which corresponds to an edge in the skeleton of the PRM, (ii) schema link: which models relationships between primary keys and foreign keys, and (iii) semantic link: which represents an influence between attributes which can be manually specified based on domain knowledge for instance. In order to reason on queries selecting specific instances of the database, the dependencies of the SIM must be instantiated at the instance-level. This results in a SIG where the nodes of the graph represent an instance attribute value. To detect inference attacks based on the SIG, the system administrator must first define which attribute are considered as sensitive in the SIM as well as defining for each of them an inference threshold. Each queried parent attributes will modify the percentage of confidence an entity has concerning the sensitive child attribute. Hence, when a new query is issued to the protected database, the detection system applies an evidence on each of the selected nodes in the SIG. Then, the

system checks whether the probability of a sensitive attribute becomes greater than its defined inference threshold.

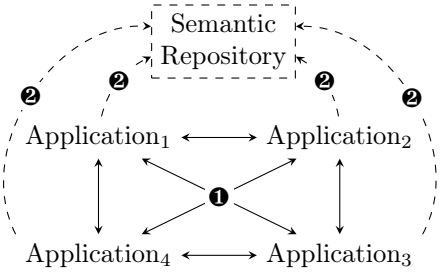
In the following we develop an example, extracted from [CC08], based on the SIM and SIG depicted by Figure 1 in order to illustrate how the solution of Chen et al. [CC08] works. Here, a subset of the SIG makes references to three distinct instances: *LAX*, *R1*, and *C5* which correspond to an airport, a runway, and an aircraft, respectively. Each attribute is prefixed with the name of its related instance to avoid any ambiguity. In this illustration, the sensitive attribute is TAKEOFF_LANDING_CAPACITY (TLC) with an inference threshold equal to 70%. Assuming that an entity knows, e.g., via a previous query, that *C5* is able to land on *R1* from *LAX* and that she has queried C5_MIN_LAND_DIST (= *long*) and C5_MIN_RUNWAY_WIDTH (= *wide*), then the entity is able to infer the value of TLC with a confidence of 58.3%. If she succeeds to query LAX_PARKING_SQ_FT (= *large*), she would be able to infer that TCL is equal to *large* with a percentage of confidence of 71.5%. Since it is above the defined inference threshold, an inference attack is detected for this last query. Therefore, this solution is limited to detecting attacks that issue queries to the protected database only, and not from external sources, e.g., from collaborating data controllers.

4 Distributed detection of inference attacks

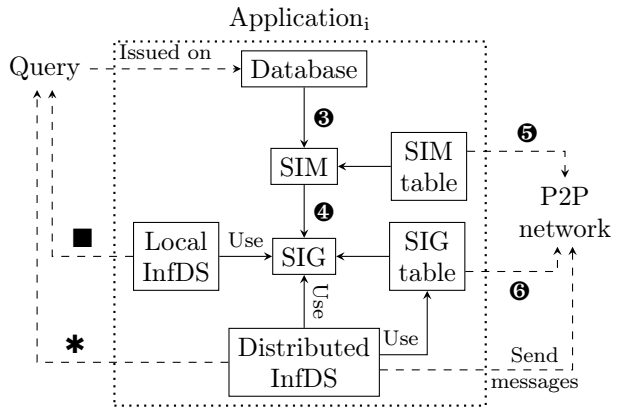
In the following, we propose a solution to the problem of distributively detecting inference attacks leveraging the DDS. As a first step, we define two generic assumptions. We assume that: (i) The applications are collaborating in order to protect the privacy of their customers. (ii) The authorized entities use a single identity when they interact with the applications (i.e., there is no collusion between entities).

As depicted in Figure 2a, the collaborating applications are uniquely identified and form a peer-to-peer (P2P) network to share information for the distributed detection ❶. Each application creates and manages locally its own SIM ❸ and its own SIG ❹, and is able to locally detect inferences thanks to the local InfDS (see Figure 2b). Hence, upon receiving the query of an authorized entity, an application performs two actions: (i) ■ It checks locally if the selected attribute leads or not to an inference on its own database. (ii) ✱ It informs the other applications about the queried information, via the P2P network, in order to let them keep track of the knowledge obtained by the entity and to detect inference attacks leveraging the DDS.

This second action requires that each application knows what the semantic similarities are that exist between its own database and the databases of the other applications. Consequently, the applications must cooperate to build a distributed GIG while protecting their own privacy and the privacy of their customers.



(a) Applications collaborating via a peer-to-peer network.



(b) Architecture of an application in the network.

Figure 2: Distributed architecture enabling the collaboration of applications in order to detect inference attacks using the DDS.

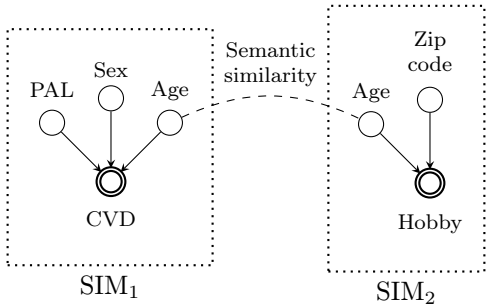


Figure 3: SIMs representing the dependencies within two distinct databases, connected via two similar attributes. PAL: Physical Activity Level and CVD: CardioVascular Disease.

4.1 Distributed computation of the GIG

The applications must compute the GIG without relying on a centralized system, but instead they exchange, via the P2P network, the required information to build the GIG. Now, the risk is to leak private information to the other collaborating applications. Hence, the required information must be shared while providing privacy guarantees to the applications. Thus, the following privacy requirements must be met when building the distributed GIG:

1. In the distributed context, the schema information which appears in a SIM must be shared with other applications to be able to compute the similarity links. However, this information must be protected (e.g., encrypted, hashed, etc.) so that another application cannot access the name of an attribute, while still being able to perform a schema matching technique.

2. In order to compute the semantic similarities between SIGs, the applications must identify which instances in distinct databases are related to the same real world customers [LRB20]. The instances shared among applications must be encoded in a way which enables the computation of a similarity score between instances of distinct databases, while preventing other applications to retrieve attribute values of an instance.

3. If one of the collaborating applications becomes malicious, by design the solution must limit the information that is obtained about other applications. When an application receives information from the P2P network, it should not be able to know from which application it originates. Hence, both the schema and the instances must be protected using techniques to avoid disclosing this identity.

For the first requirement, we assume that at the start of the collaboration, each application has access to a common semantic repository (e.g., using a generic resource such as Wikidata [Erx+14]) and maps each of its SIM nodes to a semantic term from the semantic resource ❷. The usage of such resource enables the data controllers to share information about their database schema, without having to share the schema by itself. They can choose to provide precise semantic terms to improve the accuracy of the identified semantic similarity links, or more generic terms to reduce the schema information indirectly leaked to the other DCs. The issues related to the requirements 2 and 3 are not tackled in this article.

Based on those requirements, we replicate the two stages described in Section 1 to compute the GIG. First, in addition to the local SIM and SIG, each application has a *SIM table*, ❺ in Figure 2b, with three columns: (i) the unique identifier of a node (i.e., an attribute) in the local SIM (ii) the semantic term associated to this attribute (c.f., requirement 1) (iii) one or multiple encrypted SIM node identifiers, which have the same semantic term in other collaborating applications. Those identifiers are stored in an encrypted way to prevent the application from identifying the origin and the name of the attribute. Figure 3 shows an example

where two SIMs are connected, since two attributes are semantically similar in the two distinct databases. The SIM table enables applications to keep track of those links between the SIMs of collaborating applications. Following the same idea, each application has a *SIG table* ❻ with three columns: (i) the unique identifier of a node (i.e., the attribute of an instance) in the local SIG (ii) the identifier of the related attribute in the local SIM (iii) one or multiple encrypted (c.f., requirement 1) SIG node identifiers, related to a similar instance in the other collaborating applications. Thus, the *SIG table* allows the collaborating applications to have a distributed representation of the GIG by keeping track of the similarity links between the SIGs. Note that the way the SIM and the SIG tables are filled in thanks to some protocols among application is an ongoing work and will not be presented in this paper.

4.2 Local detection & Propagation of the queried information

As described above, once receiving a query issued by an authorized entity e, the local InfDS performs the detection in two phases. First, it uses its local SIG to check if there is an inference attack on its own database. If so, then the application prevents the attack by denying the query. Otherwise, it shares the information about the queried data with the other collaborating applications, in order to let them keep track of the knowledge obtained by the entity. The local InfDS sends on the P2P network a message containing the protected identifier of the queried node in its local SIG, as well as the entity's identifier. As soon as this message is received, each application checks that the protected identifier of the node appears in its *SIG table*. If so, it updates its local SIG, indicating that this entity e has queried this information from one of the collaborating applications. Otherwise, it means the application does not have a similarity link with the queried node, hence it ignores the message.

In case an entity issues at the same time multiple queries to distinct applications, it can lead to an unintended disclosure. For instance, if each application first checks locally for an inference, they each conclude that the received query does not lead to an inference and thus answer it. To prevent this threat, the distributed detection system implements a token distribution. In this way, when an entity e queries a data from the database of one of the collaborating applications p, p takes the token of the entity e and releases it once the distributed detection is finished. Hence, any further query by the same entity could not be answered in the mean time. Note that a more intelligent token management system with smaller granularity could be studied to assign tokens only if the data asked in parallel could be a threat on a data at a participating application.

The distributed detection is performed following one of the three strategies listed below:

(i) It starts by propagating the queried information to the network ✱. (a) If another collaborating application says that this information leads to an inference, then the initial application concludes to not answer the query, in

order to avoid the attack. (b) If no inference is detected, the application checks if the query leads to an inference for its local SIG ■.

(ii) It first performs locally the detection ■, propagates the information to the network ✱, then concludes if it can safely answer the query.

(iii) It first performs locally the detection ■, concludes based on the result if it can safely answer the query, and propagates the information to the network ✱.

In the two first choices, the initial application ensures that the query does not lead to an inference, both for itself and for others. However, it increases the query answer time, since it waits for the answer of each other application. In the last choice the application prioritizes the decrease of the query answer time, while still allowing other applications to keep track of the queried knowledge, although the disclosure risk still exists.

A first implementation of the proposed distributed system is currently in progress. The communication via a peer-to-peer network is based on the *p2pnetwork* library[1]. The local detection of inferences is implemented based on the *pyAgrum* library [DGW20] to leverage the Bayesian network data structure (to represent the SIM & the SIG), as well as algorithms to apply evidences and to propagate probabilities.

5 Conclusion

In this paper, we have highlighted the lack of solutions aiming at distributively detecting inference attacks in a multi-database context, in order to protect both the privacy of applications and the privacy of its customers. We have provided an overview of a distributed architecture in which the detection is performed by the collaborating applications. As a future work, we plan to formalize the three proposed protocols. We plan to finish the ongoing prototype to demonstrate the feasibility of our approach on a case study. Finally, we plan to evaluate and compare the different distributed detection strategies listed in Section 4.2.

References

[CC08] Yu Chen and Wesley W. Chu. "Protection of Database Security via Collaborative Inference Detection". In: *IEEE Transactions on Knowledge and Data Engineering* 20.8 (2008), pp. 1013–1027. ISSN: 1558-2191. DOI: 10.1109/TKDE.2007.190642.

[1] https://github.com/macsnoeren/python-p2p-network

[CM03] LiWu Chang and Ira Moskowitz. "A Study of Inference Problems in Distributed Databases". In: *Research Directions in Data and Applications Security: IFIP TC11 / WG11.3 Sixteenth Annual Conference on Data and Applications Security July 28–31, 2002, Cambridge, UK*. Ed. by Ehud Gudes and Sujeet Shenoi. Boston, MA: Springer US, 2003, pp. 191–204. ISBN: 978-0-387-35697-6. DOI: 10.1007/978-0-387-35697-6_15. URL: https://doi.org/10.1007/978-0-387-35697-6_15 (visited on 04/04/2022).

[DGW20] Gaspard Ducamp, Christophe Gonzales, and Pierre-Henri Wuillemin. "aGrUM/pyAgrum : a toolbox to build models and algorithms for Probabilistic Graphical Models in Python". In: *Proceedings of the 10th International Conference on Probabilistic Graphical Models*. Ed. by Manfred Jaeger and Thomas Dyhre Nielsen. Vol. 138. Proceedings of Machine Learning Research. PMLR, 2020, pp. 609–612. URL: https://proceedings.mlr.press/v138/ducamp20a.html.

[Erx+14] Fredo Erxleben et al. "Introducing Wikidata to the Linked Data Web". In: *The Semantic Web – ISWC 2014*. Ed. by Peter Mika et al. Cham: Springer International Publishing, 2014, pp. 50–65. ISBN: 978-3-319-11964-9.

[FJ02] Csilla Farkas and Sushil Jajodia. "The Inference Problem: A Survey". In: *ACM SIGKDD Explorations Newsletter* 4.2 (2002), pp. 6–11. ISSN: 1931-0145. DOI: 10.1145/772862.772864. URL: https://doi.org/10.1145/772862.772864 (visited on 04/04/2022).

[Ger+18] Armin Gerl et al. "LPL, Towards a GDPR-Compliant Privacy Language: Formal Definition and Usage". In: *Transactions on Large-Scale Data- and Knowledge-Centered Systems XXXVII*. Ed. by Abdelkader Hameurlain and Roland Wagner. Berlin, Heidelberg: Springer Berlin Heidelberg, 2018, pp. 41–80. ISBN: 978-3-662-57932-9. DOI: 10.1007/978-3-662-57932-9_2. URL: https://doi.org/10.1007/978-3-662-57932-9_2.

[Get+01] Lise Getoor et al. "Learning Probabilistic Relational Models". In: *Relational Data Mining*. Ed. by Sašo Džeroski and Nada Lavrač. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 307–335. ISBN: 978-3-662-04599-2. DOI: 10.1007/978-3-662-04599-2_13. URL: https://doi.org/10.1007/978-3-662-04599-2_13.

[GMB17] Marco Guarnieri, Srdjan Marinovic, and David Basin. "Securing Databases from Probabilistic Inference". In: *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*. 2017 IEEE 30th Computer Security Foundations Symposium (CSF). 2017, pp. 343–359. DOI: 10.1109/CSF.2017.30.

[LRB20] Paul Lachat, Veronika Rehn-Sonigo, and Nadia Bennani. "Towards an Inference Detection System Against Multi-database Attacks". In: *New Trends in Databases and Information Systems*. Ed. by Jérôme Darmont, Boris Novikov, and Robert Wrembel. Cham: Springer

International Publishing, 2020, pp. 199–209. ISBN: 978-3-030-54623-6. DOI: 10.1007/978-3-030-54623-6_18.

[TFE10] Tyrone S. Toland, Csilla Farkas, and Caroline M. Eastman. "The Inference Problem: Maintaining Maximal Availability in the Presence of Database Updates". In: *Computers & Security* 29.1 (2010), pp. 88–103. ISSN: 0167-4048. DOI: 10.1016/j.cose.2009.07.004. URL: https://www.sciencedirect.com/science/article/pii/S0167404809000789 (visited on 04/04/2022).

[WB10] Philip Woodall and Pearl Brereton. "A Systematic Literature Review of Inference Strategies". In: *International Journal of Information and Computer Security* 4.2 (2010), pp. 99–117. ISSN: 1744-1765. DOI: 10.1504/IJICS.2010.034813. URL: https://www.inderscienceonline.com/doi/abs/10.1504/IJICS.2010.034813 (visited on 04/04/2022).