Data Privacy in the Semantic Web of Things

Hira Asghar

▶ **To cite this version:**

Hira Asghar. Data Privacy in the Semantic Web of Things. Artificial Intelligence [cs.AI]. Université Grenoble Alpes [2020-..], 2023. English. NNT : 2023GRALM085 . tel-04650289

HAL Id: tel-04650289
https://theses.hal.science/tel-04650289v1

Submitted on 16 Jul 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ GRENOBLE ALPES

École doctorale : MSTII - Mathématiques, Sciences et technologies de l'information, Informatique
Spécialité : Informatique
Unité de recherche : Laboratoire d'Informatique de Grenoble

Confidentialité des Données dans le Web Sémantique des Objets

Data Privacy in the Semantic Web of Things

Présentée par :

Hira ASGHAR

Direction de thèse :

Marie-Christine ROUSSET PROFESSEURE DES UNIVERSITES, UNIVERSITE GRENOBLE ALPES	Directrice de thèse
Christophe BOBINEAU MAITRE DE CONFERENCES, GRENOBLE INP	Co-encadrant de thèse

Rapporteurs :

PASCAL MOLLI
PROFESSEUR DES UNIVERSITES, UNIVERSITE DE NANTES
BENJAMIN NGUYEN
PROFESSEUR DES UNIVERSITES, INSA CENTRE VAL DE LOIRE

Thèse soutenue publiquement le **15 décembre 2023**, devant le jury composé de :

CLAUDIA RONCANCIO, PROFESSEURE DES UNIVERSITES, GRENOBLE INP	Présidente
PASCAL MOLLI, PROFESSEUR DES UNIVERSITES, UNIVERSITE DE NANTES	Rapporteur
BENJAMIN NGUYEN, PROFESSEUR DES UNIVERSITES, INSA CENTRE VAL DE LOIRE	Rapporteur
SILVIU MANIU, PROFESSEUR DES UNIVERSITES, UNIVERSITE GRENOBLE ALPES	Examinateur

Invités :

MARIE-CHRISTINE ROUSSET
PROFESSEURE DES UNIVERSITES, UNIVERSITE GRENOBLE ALPES
CHRISTOPHE BOBINEAU
MAITRE DE CONFERENCES, GRENOBLE INP

Résumé

Les données personnelles sont de plus en plus diffusées sur le web par l'intermédiaire d'appareils mobiles et d'environnements intelligents, et sont exploitées pour développer des services et des applications de plus en plus sophistiqués. Toutes ces avancées s'accompagnent de risques sérieux d'atteintes à la vie privée qui peuvent révéler des informations privées que les producteurs de données souhaitaient ne pas divulguer. Il est donc de la plus haute importance d'aider les producteurs de données à identifier les risques d'atteinte à la vie privée soulevés par les demandes des fournisseurs de services à des fins utilitaires.

Dans cette thèse, nous abordons le problème de la préservation de la vie privée en fonction de l'utilité dans le cadre d'applications où les fournisseurs de services demandent la collecte de données auprès des producteurs de données afin d'effectuer des analyses de données agrégées à des fins d'optimisation ou de recommandation. Tout d'abord, nous traitons l'aspect temporel dans la définition de la vie privée et de l'utilité en exprimant les requêtes de vie privée et d'utilité sous forme de requêtes conjonctives agrégées temporelles. La prise en compte de l'aspect temporel pour la protection de la vie privée est très importante car de nombreuses applications traitent des données dynamiques (par exemple, la consommation d'électricité, les séries chronologiques, les données de mobilité) pour lesquelles les données temporelles sont considérées comme sensibles et les agrégats temporels sont importants pour l'analyse des données. Ensuite, nous formalisons les risques d'atteinte à la vie privée par des demandes exprimées (et gardées secrètes) par chaque producteur de données pour spécifier les données qu'il ne souhaite pas divulguer et nous développons un cadre formel pour détecter les risques d'atteinte à la vie privée. Dans notre cadre formel, nous fournissons la caractérisation des risques pour la vie privée uniquement sur la base des expressions de requête et donc indépendamment des données.

Nous étendons le cadre formel en prenant en compte les connaissances ontologiques, qui aident les producteurs de données à comprendre les risques détectés pour la vie privée grâce aux explications élaborées pour chaque risque détecté. En outre, notre cadre fournit également plusieurs options pour modifier les requêtes d'utilité afin d'éliminer les risques de confidentialité détectés et ces requêtes d'utilité modifiées peuvent être envoyées aux fournisseurs de services comme base de négociation.

Dans cette thèse, nous développons également une interface interactive conviviale au-dessus de la mise en œuvre du cadre formel indépendant des données.Elle aide les producteurs de données à gérer la tension entre les risques pour la vie privée et l'utilité des données qu'ils acceptent de publier.Il fournit un environnement convivial pour détecter et comprendre les risques pour la vie privée et facilite la modification des requêtes d'utilité pour éliminer les risques détectés pour la vie privée. Pour évaluer la facilité d'utilisation pratique et l'efficacité de l'interface, une étude utilisateur est menée sur un scénario de compteur intelligent inspiré d'un cas d'utilisation réel.

Abstract

Personal data are increasingly disseminated over the Web through mobile devices and smart environments, and are exploited for developing more and more sophisticated services and applications. All these advances come with serious risks for privacy breaches that may reveal private information wanted to remain undisclosed by data producers. It is therefore of utmost importance to help the data producers in identifying privacy risks raised by the requests of service providers for utility purposes.

In this thesis, we approach the problem of utility-aware privacy preservation in the setting of applications where service providers request collecting data from data producers in order to perform aggregate data analytics for optimization or recommendation purposes. First, we handle the temporal aspect in the definition of privacy and utility by expressing privacy and utility queries as temporal aggregated conjunctive queries. Taking into account the temporal aspect for privacy protection is very important since many applications handle dynamic data (e.g., electrical consumption, time series, mobility data) for which temporal data are considered sensitive and aggregates on time are important for data analytics. Then we, formalize privacy risks by privacy queries expressed (and kept secret) by each data producer to specify the data they do not want to disclose and develop a formal framework for detecting privacy risks. In our formal framework, we provide the characterization of privacy risks solely based on the query expressions and thus independent of the data.

We extend the formal framework by taking into account ontological knowledge, which helps the data producers in understanding the detected privacy risks through the explanations constructed for each detected privacy risk. Moreover, our framework also provides several options for modifying the utility queries in order to remove the detected privacy risks and these modified utility queries can be sent to

the service providers as a basis for negotiation.

In this thesis, we also develop an interactive user-friendly interface on top of the implementation of the formal data-independent framework. It helps the data producers in managing the tension between the privacy risks and the utility of the data they accept to publish. It provides a user-friendly environment for detecting and understanding the privacy risks and facilitates in modifying the utility queries to remove the detected privacy risks. To evaluate the practical usability and effectiveness of the interface, a user study is conducted that focuses on a smart meter scenario inspired by a real-world use case.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

After proving the efficacy of Semantic Web Technologies in various domains such as finance [CLS09], business [TPC03; Hep08], medicine [Pis04], and e-learning [IMa13] it is now being considered as the future of Internet of Things (IoT). Current research efforts aim to incorporate Semantic Web capabilities into the Web of Things (WoT), resulting in the development of the Semantic Web of Things (SWoT), an enhancement of the WoT that utilizes Semantic Web technologies and principles to improve the IoT [GPa17]. Several authors have contributed to this area, addressing issues such as data integration, data storage, interoperability, data access, scalability, semantic reasoning and interpretation [BWa12; SS15; TSH09; SW16]. In addition, some authors [GP17; CGa10] have also presented the IoT-related issues that can be resolved with the evolution of SWoT.

The Internet of Things (IoT) has brought forth an unprecedented level of technological, medical, and social advancement to our daily lives. IoT applications range from smart objects interacting through embedded sensors in homes (such as smart thermostats, smart locks, and smart televisions), cities (such as smart buildings, smart grids, and smart traffic systems) and wearables (such as smart watches, smart rings and smart glasses). However, as we move towards a smarter world through IoT, we are simultaneously entering an era of immense data collection related to our interactions with sensory devices in our daily lives [BRa17]. One of the major challenges that arises is the integration and interoperability of the data collected from various IoT devices, which can be addressed using Semantic Web technologies such as ontologies, semantic annotation, Linked Open Data and

Semantic Web services. Ontologies provide a shared understanding of a domain of knowledge, enabling humans and machines to communicate [Qas+23]. Currently, over 550 ontology-based projects for IoT have been developed and are available in the Linked Open Vocabularies for the Internet of Things (LOV4IoT) ontology catalogue[1].

The use of Semantic Web technologies in the context of the IoT presents numerous research challenges and while attaining all the challenges, it is imperative to consider the privacy of the data being utilized. Personal data are increasingly disseminated over the internet through mobile devices and smart environments, and are exploited for developing more and more sophisticated services and applications. All these advances come with serious risks for privacy breaches that may reveal private information wanted by users to remain undisclosed. It is therefore of utmost importance to help data producers to keep the control on their data for their privacy protection while preserving the utility of disclosed data for service providers.

The goal of this thesis is to provide a utility-aware privacy preserving framework for detecting privacy risks and to help data producers in understanding and removing privacy risks when exchanging data in the Semantic Web of Things.

In this thesis, to attain the above mentioned goal, we consider the setting of applications where service providers (data consumers) perform data analytics on data concerning their customers for optimization or recommendation purposes. In such settings, data from sensors are gathered, abstracted and transferred through internet protocols from data producers environment (e.g., smart home, smart personal devices) to a centralized data consumer in charge of aggregating data for conducting varied analytic tasks.

Sensitive data leakage can occur at different stages and places due to security vulnerabilities of *(i)* the network, *(ii)* the centralized server used by the data consumer for collecting data outsourced by the different data producers, and *(iii)* the local servers of each data producer.

Following the vision of [All+10], we propose, first, to rely on data encryption to secure data exchange through the network and, second, to avoid the privacy risks of data centralization by keeping the data produced by each data owner decentralized in secure personal data servers.

[1] `https://lov4iot.appspot.com/?p=ontologies/`

The setting that we propose is illustrated in Figure 1.1 and can be summarized as follows:
- The data producers keep the control on their data and have the choice to transmit their data to data consumer according to their own *privacy policy* (a set of privacy queries). The privacy policies are specific to each data producer that specifies the local data they do not want to disclose.
- The data consumer express the data needs by *his/her utility policy* (a set of utility queries) and explain for which task or service s/he requests data from data producers.
- Privacy risks are detected by evaluating the query expressions of privacy and utility queries on the data producer's side. Detected privacy risks come with explanations to help data producers understand the privacy risks associated with their data. Based on the explanations of detected privacy risks, the data producer is provided with several options to remove the detected privacy risks and is guided to propose new utility queries that can be sent to the data consumer as a basis for negotiation.

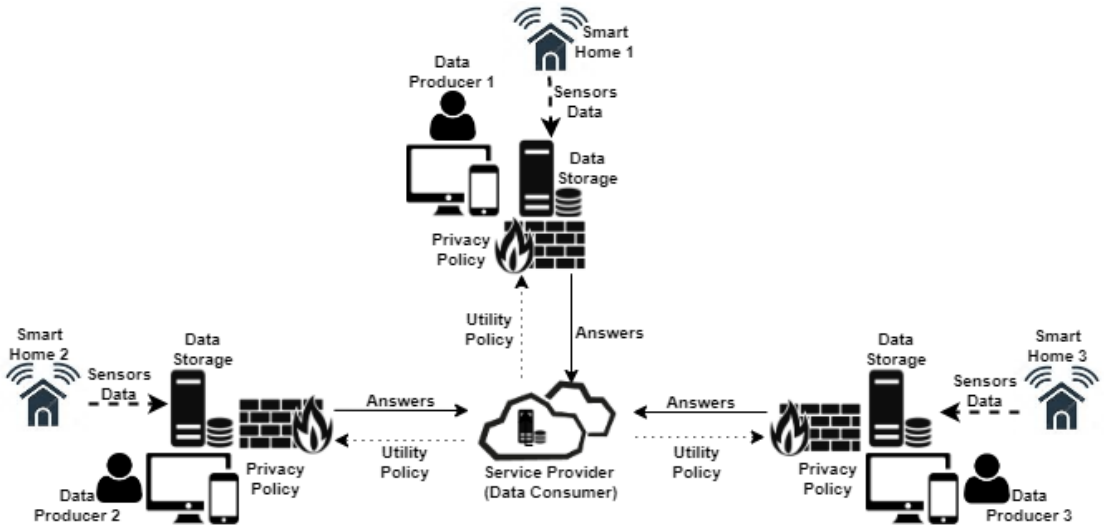


Figure 1.1: General setting

This chapter is structured as follows. Section 1.1 presents the contributions of this thesis. Section 1.2 presents the organization of this thesis manuscript.

1.1 Thesis contributions

The main contributions of this thesis can be summarized as follows:

- For detecting privacy risks raised by the utility policy, our contribution is threefold. First, by extending the framework proposed in [DBa18], we formalize privacy and utility policies as temporal aggregated conjunctive queries. Second, we formally define and characterize privacy risks based on the query expressions of privacy and utility queries and thus independently of the data. Third, considering the characterization of each privacy risk and its proof, we design and implement algorithms for detecting privacy risks. The results of this part of the thesis have been published in the proceedings of the 37th Conference on Data Management- Principles Technologies and Applications (BDA 2021) [ABR21] and in the proceedings of the 23rd International Conference on Web Information Systems Engineering (WISE 2022) [ABR22].
- To help data producers in understanding the privacy risks, factual explanation for each privacy risk is constructed along with several options for negotiating the utility queries in order to remove the privacy risks associated to their data. We also developed *PrivEx*, an interactive user-friendly interface on top of the implementation of the formal results presented in [ABR22]. *PrivEx* provides several types of support to data producers in managing the tension between the privacy risks and the utility of the data they accept to publish. First, it presents in an interpretable form the requests of a service provider for utility purpose. Second, it provides a form-based interface for guiding data producers in construction of privacy queries. Third, it detects the privacy risks and explains each privacy risk through example. Last, it provides several options for negotiating the utility queries to remove the detected privacy risks. The results of this part of the thesis have been published in the demo proceedings of the ESWC 2023 conference [ABR23].

1.2 Thesis organization

This thesis is organized into the following chapters. Chapter 2 presents the existing literature, research, and advancements relevant to the scope of this thesis. Chapter 3 presents the main definitions and standards on which this thesis is based. Chapter 4 presents the characterization and detection of privacy risks by evaluating privacy and utility policies. Chapter 5 presents our approach for helping data

producers in understanding and removing the detected privacy risks. Chapter 6 concludes the results of our thesis and presents future research directions.

Chapter 2

State of the art

Over the past few decades, there has been a notable increase in privacy attacks targeting the extraction of sensitive information from protected data, as documented in a comprehensive survey [Dwo+17]. These privacy attacks present substantial risks to disclosing sensitive information about an individual, even when the data is anonymized or protected. Considering these evolving threats, it becomes evident that there is a need for new data privacy preserving approaches to safeguard individuals' information. In this chapter, we will address certain limitations of existing data privacy preserving approaches to emphasis the importance of our proposed approach.

In this thesis, we introduce a privacy preserving approach that is designed for use in the context of SWoT and is appropriate for the protection of temporal RDF data. First, in Section 2.1, we will focus on notable privacy preserving methods intended to protect personal data within the context of RDF and Linked Data. In Section 2.2, we will focus on notable privacy preserving methods intended to protect personal data within the context of IoT or connected environments such as smart grids, homes and cities. In Section 2.3, we will present the connection between our proposed framework and existing methods. It would be exhaustive to provide a review of all existing data privacy-preserving approaches in the aforementioned context. Therefore, we will primarily focus on discussing some prominent existing methods.

2.1 Privacy preserving methods for RDF and Linked Data

In Section 2.1.1, we will provide an overview of anonymization methods for RDF and Linked Data. In Section 2.1.2, we will focus on access control methods, which play an important role in ensuring data privacy in the context of RDF and Linked Data. In Section 2.1.3, we will present encryption methods for the protection of RDF and Linked Data.

2.1.1 Anonymization methods

To the best of our knowledge, existing methods in the context of RDF data anonymization are limited. In [RGP15], researchers mainly applied generalization and suppression operations. Furhermore, in works such as [HHD17], specific areas or neighborhoods are defined where anonymization properties like k-anonymity are satisfied. In these papers the researchers simplify and anonymize RDF structures by reducing them to micro-data, thereby handling a vast amount of information encompassing heterogeneous nodes and relations. However, when dealing with RDF datasets containing thousands of diverse resources, the current solutions prove inadequate due to the use of greedy algorithms to generate all possible solutions (anonymous RDF) and subsequently evaluate and select the most suitable one. Given that RDF data can sometimes be transformed into structured data resembling databases, database anonymization techniques could also be considered. While these methods can manage smaller RDF datasets effectively, they may result in significant semantic information loss (properties), particularly in the case of big RDF datasets, when simplifying their complexity into structured models.

In [Aro13] and [SLa17], the authors proposed differential privacy based solutions for the preservation of Linked Data. Differential privacy does not align perfectly with Linked Data, as it places a stronger emphasis on preserving statistical integrity rather than ensuring accurate and qualitative query results. The primary utility of Linked Data typically revolves around qualitative query results, especially when accessed through SPARQL endpoints. While differential privacy serves a valuable purpose in scenarios where aggregated data analysis results, such as statistics regarding groups of individuals, can be safely disclosed, its application falls short in Privacy-Preserving Data Publishing (PPDP) contexts. In PPDP, the paramount goal is to safeguard individuals' privacy while still making the published

data practically usable. While the foundational principles of PPDP, particularly anonymization, have been extensively studied for relational data (as exemplified in [FWa10] with a comprehensive survey), the theoretical groundwork for PPDP within the Linked Data context has only been laid out in [GK16] and [GK19]. These works primarily focus on the examination of computational complexity in the context of PPDP for Linked Data.

In [DBa18], a query-based approach for preserving the privacy of RDF data publishing has been presented. The primary focus of this approach is to check the compatibility between a privacy policy and a utility policy, both of which are specified as queries. Additionally, the aim is to create anonymizations that preserve the answers to a set of utility queries (when compatibility is satisfied). However, this approach has a vulnerability when it comes to safeguarding against privacy breaches resulting from linking external datasets, which is a common occurrence in the Linked Open Data (LOD) environment. They developed safe anonymizations of an RDF graph to ensure that linking the anonymized graph with any external RDF graph would not result in privacy breaches. By taking a set of privacy queries as input, they provided a data-independent safety solution along with the necessary sequence of anonymization operations to enforce it. Nevertheless, it's worth noting that this approach has limitations. It cannot support automatic synchronization, which means performing consistent replacements of a data item or an IRI not only in the sensitive triple but also throughout the entire graph. These essential features cannot be achieved using any of the existing SPARQL operations or query forms, such as CONSTRUCT or UPDATE.

Considering the aforementioned limitations of the SPARQL query/update language, the authors presented a language for sanitizing RDF graphs in [RKa14]. It consists of a collection of sanitization operations that modify a graph by hiding sensitive data. These operations are integrated into a novel SPARQL query form known as SANITIZE. They have introduced three sanitization operations SNode, SEdge and SPath to anonymize the nodes, edges and paths present in RDF graphs. In addition to this, they also provided synchronization operation to perform automatic synchronization in the complete RDF graph. This approach is adequate for providing privacy to RDF dataset but the authors did not provide sufficient insight on preserving utility with privacy.

In [DC19], authors introduced the RiAiR framework. Its primary objective is to simplify the RDF structure to streamline the process of expert users classifying

RDF data into categories like identifiers, quasi-identifiers, and more. An intersection process has been proposed to identify the sensitive data that needs to be anonymized whereas basic generalization and suppression mechanism has been used for data anonymization. Unlike other privacy preservation methods this approach mainly focused on what to anonymize rather than on how to anonymize and they provided a mechanism for identifying the information that needs to be anonymized.

2.1.2 Access control methods

An alternative approach for protecting against privacy breaches consists in applying access control methods to RDF data [KMD17]. In case there is a possibility to infer sensitive data from the answers to a query disclosed to a user then in the literature of access control, this problem is known as inference problem [FJ02]. This survey is aimed to be concise and focused to provide a summary of the common access control approaches that either consider inference rules (specified by W3C in [Rec04a]) or not in order to enforce authorizations.

Several access control models have been proposed in relation to RDF data (without inference rules) [Abe+07; Flo+10]. In [Abe+07], authors introduced a query rewriting mechanism that aims to enforce authorizations. The formal semantics of the authorization language are not presented by the authors, and their conflict resolution strategies are implemented in a hard-coded manner. In [Flo+10], authors proposed access control language that utilizes annotations. This language is designed to enable fine-grained authorizations on RDF data. The authors also provide a formal semantics for this language. The definition of authorizations in this paper is evidently influenced by the work of [Flo+10]. The approach employed involved utilizing a predetermined set of conflict resolution strategies, namely *deny/permit takes precedence* and *deny/permit by default*. However, the specific details regarding precedence are not provided.

Alternative methodologies that take into account inference rules and employ propagation techniques to compute authorizations that can be applied to inferred triples [RFJ05; Lop+12; Pap+12]. An access control language that takes update operations into account for RDF stores is proposed by authors in [RFJ05]. They do not offer formal semantics for their language; instead, they define default policies and conflict resolution strategies using meta-rules. In [Lop+12], authors have presented a similar method that is based on provenance and involves labelling each

triple and propagating the labels using an inference procedure and a fixed conflict resolution strategy. A versatile model that defines a triple's access label as an algebraic expression is put forth by the authors in [Pap+12]. The authors limited their analysis to a specific set of RDFS rules, excluding any user-defined rules. In conclusion, label-based strategies in the field of authorization may employ more sophisticated authorization languages or incorporate update mechanisms. However, it is important to note that these techniques rely on underlying base graphs and do not take into account the potential risk of information leakage.

In [JF06; Say+15], the authors focused on addressing the inference problem. In [JF06], authors introduced a label-based propagation technique designed specifically for RDF data. The authors presented an algorithm designed to identify unauthorized inferences, specifically those instances where lower security triples can be used to deduce higher security triples. However, the detection of violations requires the use of a graph, and the strategies for resolving conflicts, as well as the default strategy, are implemented as hard-coded components. On the other hand, in [Say+15], authors proposed a static analysis algorithm for writing the authorization policy that does not rely on graph knowledge, hence enabling the implementation of more adaptable conflict resolution strategies.

2.1.3 Encryption methods

When it comes to the privacy of Linked data, studies have either proposed mechanisms for accessing encrypted RDF data [KS13] or partial encryption of RDF graphs [Ger08; Gie05; Gie] by utilizing eXtensible Markup Language (XML) encryption methods. In [Gie05; Gie], authors illustrate the application of XML-based encryption methods for the purpose of encrypting sensitive data within an RDF-graph, while keeping all non-sensitive data in its original plaintext form. In [Ger08], authors extended their previous research by ananlyzing the utilization of encryption methods for encrypting RDF subgraphs and RDF elements, with an objective to reduce the overhead associated with the encryption approach. In [KS13], authors proposed a method for running user-defined SPARQL queries on encrypted graph data. Access to the graph data is restricted to users who are authorized to run queries. The approach relied on eight distinct query types that correspond to the various binding options within a single SPARQL triple pattern. The owner of the data graph may further limit the allowed queries by pre-defining a certain predicate or object. However, the scalability of this proposal is compromised due to the encryption of each triple multiple times, depending on the level of access

restriction for the subject, predicate, and/or object.

In [Fer+17], authors employed the predicate-based encryption [KSW08] to facilitate controlled access to encrypted RDF data. This approach allows data providers to generate query keys based on specific patterns, such as triple-patterns. Consequently, a single decryption key can be used to decrypt all triples that correspond to its associated triple pattern. The aforementioned research do not examine compression and encryption together, which can be useful for efficient storage and exchange of sensitive data. In [Ver+20], authors proposed a framework that combines the notion of compression and encryption of RDF data. It uses *HDT* (a compression technique for reducing the RDF data storage space) along with encryption in which different users have different access rights and can only access particular subgraphs of RDF dataset.

2.2 Data privacy preserving methods for connected environments

In Section 2.2.1, we will present privacy preserving models for data anonymization and differential privacy techniques that have been widely adopted. In Section 2.2.2, we will provide an overview of homomorphic encryption schemes and their various categories. In Section 2.2.3, we will discuss Secure Multi-Party Computations (SMPC) and highlight its relevance in Privacy-Preserving Data-Aggregation (PPDA) compared to resource-intensive homomorphic encryption schemes.

2.2.1 Anonymization methods

Anonymization involves the process of anonymizing the microdata. Microdata refers to unprocessed data that includes user information, consisting of various attributes or columns [Sam01]. To identify a user, there are three different types of identifiers that classify the attributes within microdata. The first type is explicit identifiers, that are referred to as unique identifiers such as a passport number. The second type is quasi-identifiers, which can identify a user when combined together. Examples of quasi-identifier include age and gender. The third type is sensitive attributes, which are attributes that require protection such as salary. The initial stage of the anonymization process involves eliminating the explicit identifier.

K-anonymity [Swe02] is one of the oldest approaches introduced for the data pri-

vacy preservation. It involves the anonymization of a dataset in a manner that ensures each record (or row) cannot be distinguished from at least k-1 other records, specifically in relation to the quasi-identifier properties. Anonymization is accomplished by the utilization of generalization and suppression techniques. The primary objective of k-anonymity is to mitigate the risk of linking attacks, wherein an adversary is unable to uniquely identify an individual by associating their quasi-identifier attributes (such as birth date, zip code, and gender) with other datasets. The use of k-anonymity is appropriate for non-interactive data publishing in cases where either there is absence of sensitive attribute or when the distribution of the sensitive attribute is sparse.

K-anonymity provides protection against linking attacks, which aim to identify individual records and compromise privacy. However, it is important to note that k-anonymity is vulnerable to two other types of attacks: homogeneity attacks and background knowledge attacks. In a homogeneity attack, an adversary can identify the value of a sensitive attribute if all the sensitive attributes in a group of k records are the same. In the context of a background attack, an adversary leverages their existing knowledge to discern the identities of specific individuals. In order to overcome the limitation of k-anonymity, an extension called l-diversity [Mac+06] was introduced. This approach mandates that each record within a group must possess a minimum of diverse values for the sensitive attribute. L-diversity is a viable option for non-interactive data publishing scenarios where the data publisher intends to release an anonymized dataset without the need to respond to individual queries. In contrast to k-anonymity, l-diversity is employed when the anonymized dataset necessitates each record within a group to possess a minimum of l diverse values for the sensitive attribute. In [LLV07], authors presented that l-diversity does not provide complete protection against the homogeneity attack. Two types of attacks, namely skewness attack and similarity attack, were employed to illustrate the constraints of the l-diversity approach. The skewness attack occurs when the anonymized dataset exhibits a skewed distribution of the sensitive attribute within equivalence groups. In this scenario, the l-diversity mechanism proves ineffective in preventing the attack due to the disparity between the distribution of the sensitive attribute and the dataset. In a similarity attack, the anonymized dataset contains distinct values of a sensitive attribute that are organized into equivalence groups, but these values are semantically similar. The failure of l-diversity in preventing the attack can be attributed to the adversary's ability to estimate the value of a sensitive attribute by linking it to another sensitive attribute.

The next development following the l-diversity is t-closeness [LLV07], which places emphasis to ensure that even after values have been removed or generalized, the resulting distribution of values for a specific attribute remains similar (within the threshold t) to the original distribution. This is important because attackers may be able to deduce sensitive values if they possess prior knowledge about their distribution and if the anonymization process is excessively deterministic. This trade-off results in a reduction in utility, while enhancing privacy protection, especially in specific attack scenarios.

The term differential privacy was initially introduced by Dwork [Dwo06]. It is defined as a property of a mechanism where the output remains relatively unaffected by the inclusion or exclusion of a single record from the dataset. The privacy of users can be safeguarded when sharing the dataset with an untrusted entity through the application of data perturbation techniques. This solution addresses the limitations of anonymization techniques, particularly tackling the challenges posed by high dimensionality [Sal19].

2.2.2 Homomorphic encryption

Homomorphic encryption is a cryptographic methodology that enables the execution of computations directly on encrypted data. This feature enables the preservation of data confidentiality by allowing sensitive data to remain encrypted during processing. The initial introduction of the concept of homomorphic encryption was proposed by authors in [RAD78b], who referred to it as Privacy Homomorphisms [RAD78b]. Homomorphic encryption schemes can be classified into three distinct categories:

- Partially homomorphic encryption schemes exclusively facilitate a singular operation, such as multiplication or addition, on ciphertexts. The RSA cryptosystem, as described in [RAD78a], exhibits partial homomorphism with respect to multiplication.
- Somewhat homomorphic encryption schemes offers the capability to perform both addition and multiplication operations on ciphertexts, but with a restricted number of iterations. For example, some encryption schemes support unlimited additions and a single multiplication operation.
- Fully homomorphic encryption schemes [Dij+10] allow for an unrestricted number of operations involving both addition and multiplication.

Unfortunately, the operational constraints of partially and somewhat homomorphic

encryption schemes make them unsuitable for generic computations. In the context of fully homomorphic encryption schemes, the primary focus revolves around optimizing performance and ensuring scalability. Despite significant efforts to reduce the impact, the fully homomorphic encryption schemes have limitations that hinder their practicality in real environments [Mar+22].

2.2.3 Secure multi-party computation

Secure Multi-Party Computations (SMPC) [Yao82] are algorithmic processes in which multiple parties (or individuals), each possessing different fragments of sensitive information perform a joint computation on their data to compute a specific result by utilizing the algorithms. By collectively leveraging their respective inputs, these entities can unveil hidden information, authenticate a message, or authorize a transaction. It is noteworthy to mention that SMPC accomplishes this objective while maintaining the confidentiality of the information possessed by each user, without disclosing any specific specifics.

In recent years along with Privacy-Preserving Data Publication(PPDP), Privacy-Preserving Data-Aggregation (PPDA) has also gained attention from researchers [DA21; YKD21]. Most of the current PPDA approaches depend on homomorphic encryption, which requires a lot of computing power and do not work well with real systems that are limited in resources. SMPC-based strategies, on the other hand, try to find a collaborative answer for PPDA by relying less on computation but heavily on communication and data sharing between the entities. Consequently, various researchers like in [GS22] are making efforts to explore and enhance SMPC-based approaches.

2.3 Position of this thesis

A query-based logical framework for RDF data has been introduced in [GK16; GK19], where sensitive information is expressed as a privacy policy in the form of SPARQL query whose results must not disclose sensitive information of individual. It has been extended to handling utility queries in [DBa18], they proposed theoretical criteria for finding the compatibility between utility policy and privacy policy, where both policies are specified as conjunctive queries. When a privacy query turns out to be incompatible with utility queries, they have used anonymization as a safeguard measure (see Section 2.1.1). The both approaches however are

restricted to simple conjunctive queries. They do not consider aggregated queries and they cannot be applied to temporal RDF data. We have extended their approach by formalizing the utility and privacy policies in the form of temporal aggregated conjunctive queries. We proposed a (data-independent) query-based framework that is suitable for protecting temporal RDF data and it comes with explanations of privacy risks and builds several options for negotiating the utility queries for removing the privacy risks. In our negotiation process, one of the options for removing privacy risks is the generalization of some properties in the ontology, for which rules of generalization have been user-defined, to ensure that data to be kept secret by the data producer is not disclosed.

In contrast to all approaches that are based on changing the exposed data either by adding noise to the data or by applying generalization operations to sensitive data, our data-independent approach is complementary and should be used beforehand for detecting privacy risks. We also discussed an alternative approach for protecting against privacy risks by applying access control methods to RDF data. However, these approaches do not handle utility policy.

Chapter 3

Preliminaries

In this chapter, we introduce the main definitions and standards on which this thesis is based. In Sections 3.1 and 3.2, we summarize the RDF and RDFS standards for describing data and ontologies on the Semantic Web and we present the extension that we consider to capture temporal data and dynamic properties. In Section 3.3, we define temporal aggregated conjunctive queries used in our approach and we provide their semantics when evaluated over temporal RDF graphs. We illustrate the different notions through examples that are built using the ISSDA dataset, a real world power grid dataset provided by the *Irish Social Science Data Archive (ISSDA) Commission for Energy Regulation (CER)*[1].

3.1 Temporal RDF graphs

Temporal RDF graphs are an extension of the graph data model standardized for the Semantic Web by the Resource Description Framework (RDF) [Rec14a]. In RDF, web resources are described by statements, where each RDF statement is a triple consisting of a subject, a property and an object. A subject can only be an Internationalized Resource Identifier (IRI) or a blank node. A property can only be an IRI. An object can be a IRI, a blank node or a literal. Literals and IRIs are called constants.

For example, "The occupier has a yearly income whose value is 75000" can be represented as the RDF statement: (_:o1031 :yearlyIncome "75000"), where *_:o1031* is

[1] `https://www.ucd.ie/issda/data/commissionforenergyregulationcer/`

the subject (an occupier is represented by a blank node), *:yearlyIncome* represents the property and *"75000"* is a literal corresponding to the value.

Definition 1 (RDF statement)**.** *Let I, L, and B be countably infinite pairwise disjoint sets representing respectively IRIs, literals and blank nodes. An RDF statement is a triple $(s\ p\ o)$, where $(s\ p\ o) \in (I \cup B) \times I \times (I \cup L \cup B)$.*

Definition 2 (RDF graph)**.** *An RDF graph is a finite set of RDF statements.*

In our approach, we consider temporal RDF graphs in which all statements are associated with timestamps. Each temporal statement will be represented by a pair where the first element in a pair correspond to an RDF statement and the second element is its timestamp taken in the subset TS in the set of L of literals.

Definition 3 (RDF temporal statement)**.** *An RDF temporal statement is a pair $(s\ p\ o,\ t)$ where $(s\ p\ o) \in (I \cup B) \times I \times (I \cup L \cup B)$ and $t \in TS$.*

Definition 4 (RDF temporal graphs)**.** *An RDF temporal graph is a set of RDF temporal statements.*

By convention and for homogeneity purpose we use *any* as a special timestamp when the corresponding statements holds at any time. We call *static* the properties having *any* as timestamps. We call *dynamic* the properties having actual timestamps.

Example 1 illustrates a temporal RDF graph that we have built from the real-world ISSDA dataset. The original data is tabular, and includes both personal data and temporal smart meter data related to different house owners. This example shows some data of a given house owner described using properties `:associatedOccupier`, `:yearlyIncome`, `:numberOfPersons` (that are static) and `:consumption` (that is dynamic).

Example 1. *The associated occupier, yearly income and number of persons are expressed as temporal statements with special timestamp any whereas energy consumption statements are timestamped.*

Temporal RDF graph of house owner $sm1031$

$(:sm1031\ :associatedOccupier\ \_:o1031, \text{“}any\text{”})$
$(\_:o1031\ :yearlyIncome\ \text{“}75000\text{”}, \text{“}any\text{”})$
$(\_:o1031\ :numberOfPersons\ \text{“}6\text{”}, \text{“}any\text{”})$
$(\_:o1031\ :familySize\ \text{“}Large\text{”}, \text{“}any\text{”})$
$(:sm1031\ :consumption\ \text{“}78\text{”}, \text{“}2009\text{-}07\text{-}14T00:00:00\text{”})$
$(:sm1031\ :consumption\ \text{“}60\text{”}, \text{“}2009\text{-}07\text{-}14T00:30:00\text{”})$

(: $sm1031$: $consumption$ “143”, “2009 -07 -14T01 :00 :00”)
(: $sm1031$: $consumption$ “34”, “2009 -07 -14T01 :30 :00”)

3.2 RDFS ontologies extended with rules

An ontology is a formal representation of a shared vocabulary of a domain enabling humans and machines to communicate and reason about the domain. In particular, an ontology can serve as a schema constraining the description of data in that domain.

Ontologies play a critical role in the SWoT in facilitating interoperability between different IoT devices and personalizing IoT applications and services [Qas+23].

Ontologies can be represented in various forms, including Web Ontology Language (OWL) and RDF Schema (RDFS).

In our approach, we consider that data producers and data consumers understand each other through a common ontology that is designed using RDFS in which we enable the declaration of dynamic properties, as explained in Section 3.2.1. We also extended RDFS with property generalization rules, as explained in Section 3.2.2.

3.2.1 RDFS

RDF Schema (RDFS) [Rec14c] is part of the RDF 1.1 specification [Rec14a]. It provides two namespaces `rdf:` and `rdfs:` with predefined properties to state relationships between instances, classes and properties:

- `rdfs:Class` is a meta-class for grouping all the classes in RDFS.
- `rdf:type` is used to express that a resource identified by an IRI is an instance of a class (also identified by an IRI).
- `rdf:Property` is the class, the instances of which are properties (rdf:Property is an instance of rdfs:Class).
- `rdf:Statement` is the class, the instances of which are RDF statements (rdf:Statement is an instance of rdfs:Class).
- `rdf:subject` is an instance of rdf:Property used to relate a statement to its subject.

- `rdf:predicate` is an instance of rdf:Property used to relate a statement to its property.
- `rdf:object` is an instance of rdf:Property used to relate a statement to its object.
- `rdfs:subClassOf` is used to specify subsumption relationships between classes, i.e., that a class is a subclass of another.
- `rdfs:subPropertyOf` is used to denote that a property is a subproperty (specialization) of another.
- `rdfs:domain` relates a property to a class to express that the subjects of the property are instances of the class.
- `rdfs:range` relates a property to a class to express that the objects of the property are instances of the class.
- `rdfs:label` associates a human-readable name to an IRI identifying an RDF resource.
- `rdfs:comment` associates a human-readable description to an IRI identifying an RDF resource.

Statements in which the property is an RDFS property `rdfs:subClassOf`, `rdfs:subPropertyOf`, `rdfs:domain` or `rdfs:range` are called *schema statements* (also called *RDFS statement*), as they express semantic constraints on the classes and properties used to describe RDF data of a given domain.

Within an RDF graph, we will distinguish schema statements from data statements. Though they are all denoted as RDF statements, the former ones convey a rule-based semantics provided by 13 RDFS entailment rules [Rec04b].

Table 3.1 and Table 3.2 provide subsets of these rules respectively called RDFS assertion and constraint rules in [Bur20].

RDFS rule	body ⇒ head
`rdfs2`	p rdfs:domain c , $s\ p\ o \Rightarrow s$ rdf:type c
`rdfs3`	p rdfs:range c , $s\ p\ o \Rightarrow o$ rdf:type c
`rdfs7`	p rdfs:subPropertyOf q , $s\ p\ o \Rightarrow s\ q\ o$
`rdfs9`	c rdfs:subClassOf d , s rdf:type $c \Rightarrow s$ rdf:type d

Table 3.1: RDFS assertion rules

RDFS rule	body ⇒ head
`rdfs5`	p_1 rdfs:subPropertyOf p_2 , p_2 rdfs:subPropertyOf p_3 ⇒ p_1 rdfs:subPropertyOf p_3
`rdfs11`	c_1 rdfs:subClassOf c_2 , c_2 rdfs:subClassOf c_3 ⇒ c_1 rdfs:subClassOf c_3
`ext1`	p rdfs:domain c_1 , c_1 rdfs:subClassOf c_2 ⇒ p rdfs:domain c_2
`ext2`	p rdfs:range c_1 , c_1 rdfs:subClassOf c_2 ⇒ p rdfs:range c_2
`ext3`	p rdfs:subPropertyOf p_1 , p_1 rdfs:domain c ⇒ p rdfs:domain c
`ext4`	p rdfs:subPropertyOf p_1 , p_1 rdfs:range c ⇒ p rdfs:range$c$

Table 3.2: RDFS constraint rules

For example, if *:electricOfficeEquipment* is a subproperty of *:electricEquipment*, *:pb1* is an instance of the class *:ProfessionalBuilding* and we have the RDF statement: (:pb1 :electricOfficeEquipment "true"), then the RDFS assertion rule (rdfs7) allow us to infer that the statement (:pb1 :electricEquipment"true") is also true.

In Section 3.2.1.1, we present the generic classes and generic properties introduced for the declaration of dynamic properties and explained them using an ontology. In Section 3.2.1.2, we present the ways for the implementation of temporal statements.

3.2.1.1 Temporal extension of RDFS

We have introduced generic classes named as *:DynamicProperty* and *:TemporalStatement* and generic properties named as *:onPredicate* and *:timestamp* for the modeling of dynamic properties. Below, we specify the generic classes, generic properties, and their relationships using RDFS vocabulary.

Specification of generic classes and generic properties using RDFS vocabulary

```
:DynamicProperty rdfs:subClassOf rdf:Property
:TemporalStatement rdfs:subClassOf rdf:Statement
:onPredicate rdfs:subPropertyOf rdf:predicate
:onPredicate rdfs:domain :TemporalStatement
:onPredicate rdfs:range :DynamicProperty
:timestamp rdf:type rdf:Property
:timestamp rdfs:domain :TemporalStatement
:timestamp rdfs:range xsd:dateTime
```

A domain ontology involving dynamic properties can be declared by relating domain specific classes and properties in addition to the RDFS classes and properties.

Figure 3.1 shows an extract of the RDFS ontology[2] that we have built to model the ISSDA dataset. The aforementioned generic classes and generic properties introduced for modeling of dynamic properties are highlighted in red. The dynamic properties are defined as instances of :DynamicProperty.

The *:yearlyIncome*, *:numberOfPersons*, *:associatedOccupier*, *:owns* and *:surface* are static properties. They are modeled in the conventional way as instances of the meta-class *rdf:Property* . The only dynamic property is *:consumption*, it is declared as an instance of *:DynamicProperty*.

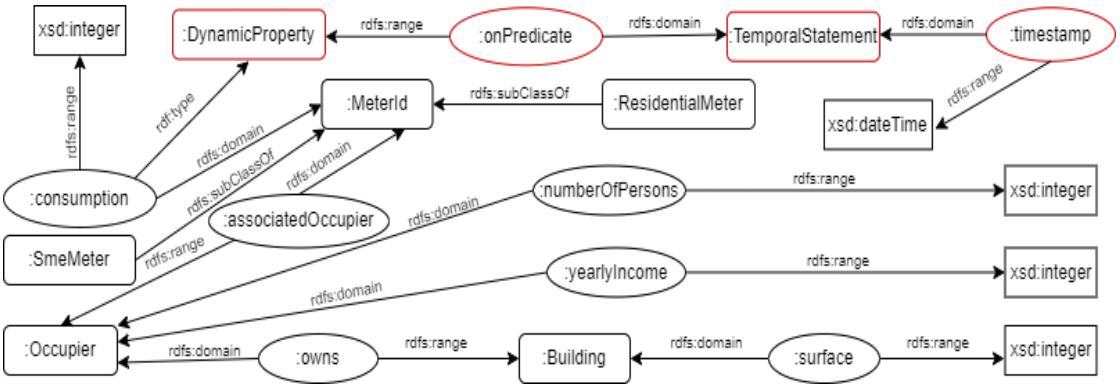


Figure 3.1: Extract of the RDFS ontology modeling ISSDA dataset

3.2.1.2 Implementation of temporal statements

Temporal statements can be implemented in RDF by using RDF reification [Rec14b]. RDF reification allows the creation of new RDF resources that represent statements, which enables the use of other properties to describe statements.

In Example 2, we show how the RDF reification method can be used to express "The smart meter number 'sm1031' had a consumption of 34 on 2009-07-14T01:30:00": first, we create a new resource *:statement1* that represents the statement: (:sm1031 :consumption "34") that is created as an instance of the property *:TemporalStatement*. Then we assert the subject, property and object for the resource. Finally, we assert the property *:timestamp* for the resource.

[2]Available at `https://raw.githubusercontent.com/repository-code/PrivEx/main/issda_schema.ttl`

Example 2. *This example illustrates the RDF reification of a statement (:sm1031 :consumption "34") and adds another property* :timestamp *to a resource of the statement.*

Adding the property :timestamp to a resource using RDF reification

```
:statement1 rdf:type :TemporalStatement
:statement1 rdf:subject :sm1031
:statement1 :onPredicate :consumption
:statement1 rdf:object "34"
:statement1 :timestamp "2009-07-14T01:30:00"
```

The temporal statements can also be implemented using RDF-star [Arn+22] extension of RDF, which allows statements to be used as the subject or object of other statements, enabling more complex and expressive representation of knowledge graphs and other data. In Example 3, we present the same temporal statement as in Example 2 implemented using RDF-star.

Example 3. *For implementing the temporal statement, a statement with the dynamic property i.e.; (:sm1031 :consumption "34") is used as a subject, :timestamp as a property and timestamp value as an object.*

Temporal statement representation in RDF-star

```
(: sm1031 : consumption "34" {|: timestamp "2009−07−14T01:30:00"|}
```

3.2.2 Property generalization rules

We have introduced schema statements of the form p **:isGeneralizedBy** q to declare that a property p is generalized by another property q.

In contrast to the generic rule *rdfs7*, the way the values of the property p are abstracted into qualitative values of q is specified by user-defined rules.
For the ISSDA dataset, we have declared three such generalization statements [3].
Table 3.3 presents the generalization rules relating the property *:numberOfPersons* to its generalization *:familySize*.

[3] Available at `https://cloud.univ-grenoble-alpes.fr/s/omsaDAQHkMtJWi9`

Rule number	Generalization rules for *:numberOfPersons* into *:familySize*
r1	s :numberOfPersons o, o<3 ⇒ s :familySize “Small”
r2	s :numberOfPersons o, 3≤o≤6⇒ s :familySize “Medium”
r3	s :numberOfPersons o, o>6 ⇒ s :familySize “Large”

Table 3.3: Generalization rules defining *:numberOfPersons :isGeneralizedBy :familySize*

3.3 Temporal aggregated conjunctive queries

Data analytics on temporal data requires the computation of aggregates on time intervals, also called time windows. Therefore, the queries that we consider in this thesis must enable to compute aggregate functions such as the sum, maximum or average of groups of selected values in the temporal data.

In Section 3.3.1, we define temporal aggregate conjunctive queries (TACQ) with a SPARQL-like syntax extended with time windows for capturing aggregates on time. In Section 3.3.2, we provide their semantics that defines the answers set of TACQS when evaluated over temporal graph patterns . In Section 3.3.3, we introduce useful notions for comparing TACQs. In Section 3.3.4, we present ways to implement TACQs as iterations on time intervals of the evaluation of SPARQL or SPARQL-star queries.

3.3.1 SPARQL-like syntax for TACQs

The SPARQL Protocol and RDF Query Language (SPARQL) [Rec13] is a standard query language for retrieving and manipulating data stored in RDF format but it cannot query or manipulate temporal data. Temporal Aggregated Conjunctive Queries (TACQs) extend the SPARQL syntax and semantics for capturing aggregate on time.

First, we will define *temporal graph pattern* that is a core part of TACQ. *Temporal graph pattern* are extensions of standard graph patterns in SPARQL in which we allow to associate *timestamp variables* to patterns involving *dynamic* properties.

Definition 5 (Temporal graph pattern)**.** *Let Var be a set of variables in which each variable starts with a special character ?. A temporal graph pattern is a finite set of temporal patterns, where a temporal pattern is a pair $(s\ p\ o, ?ts)$ such that*

$(s\ p\ o) \in (I \cup B \cup Var) \times (I \cup Var) \times (I \cup L \cup B \cup Var)$, *and* $?ts \in Var$.

Aggregated conjunctive queries allow the grouping of data and compute values over the groups. Common aggregate functions [Rec09] supported in SPARQL are as follows:

- COUNT: counts the number of times a given variable or an expression has a value within the group.
- SUM: computes the sum of numeric values for a variable or an expression over a group.
- AVG: computes the average of numeric values for a variable or an expression over a group.
- MIN: returns the minimum value for a variable or an expression from a group.
- MAX: returns the maximum value for a variable or an expression from a group.

We will only consider aggregate queries with a single aggregate term as in most cases, queries with several aggregate terms are equivalent to the unions of queries with same body and a single aggregate [Coh05]. In particular, AVG can be computed by the union of two queries, one for computing SUM and the other one for computing $COUNT$.

Defining a time window allows the evaluation of a TACQ over a sequence of time windows (of a given size) shifted by a certain time duration (known as step).

Definition 6 (Temporal aggregated conjunctive query)**.** *A TACQ is defined as:*
SELECT $\bar{x}$, $agg(y)$
WHERE $\{TGP\ .\ FILTER\}$
GROUP BY $\bar{x}$
TIMEWINDOW (Size, Step)
where:
- TGP *is a temporal graph pattern;*
- $FILTER$ *is a (possibly empty) conjunction of atomic comparisons of the form* $t\ \theta\ t'$ *where* t *and* t' *are variables of* GP *or literals (numbers, strings or dates) and* $\theta \in \{<>, <, <=, =, >=, >\}$*;*
- $\bar{x}$ *is a tuple of variables called the output (or grouping) variables;*
- when the aggregate term $agg(y)$ *is present,* y *(called the aggregate variable) is not*

in $\bar{x}$ and agg is an aggregate function that produces a single value when applied to a set of values assigned to y;
- Size and Step are time durations (i.e. differences between timestamps). A size expresses the duration of each time window, and a step expresses the time duration separating consecutive time windows.
- TIMEWINDOW(∞, 0) is a notation for the single time window covering all the timestamps in the data set.

The general syntax can be simplified as follows for capturing particular cases:

- When either $\bar{x}$ is empty or there is no aggregate term, we can omit the GROUP BY clause.
- When the query contains only static properties, we can omit the TIMEWINDOW clause.
- When the query contains the notation TIMEWINDOW(∞, 0), we can omit the TIMEWINDOW clause.
- The $FILTER$ clause can be omitted when the corresponding boolean expression is empty (called empty $FILTER$). Note however, that when TIMEWINDOW is specified, FILTER always implicitly contains the following constraints for each timestamp variable $?ts$:
  $?ts > ?timeWindowsEnd - Size \wedge ?ts \leq ?timeWindowEnd$
  where $?timeWindowEnd$ is a specific timestamp variable that will be mapped successively to the upper bound of each time window computed from the timestamp at which the query is executed.
- For static properties, the timestamp variables can be omitted from temporal graph pattern and the corresponding temporal patterns can be simplified into standard patterns.

In our approach, apart from a $TACQ$ in its general form, we will consider $TACQs$ that are without aggregate terms named as *conjunctive queries* and without time window definition named as aggregated conjunctive queries. They are defined in Definitions 7 and 8.

Definition 7 (Conjunctive query)**.** *A conjunctive query is a query without an aggregate term and can have an empty FILTER. It is defined as:*
SELECT $\bar{x}$
WHERE $\{TGP\ .\ FILTER\}$

Definition 8 (Aggregated conjunctive query)**.** *An aggregated conjunctive query is a query without a time window and can have an empty FILTER. It is defined as:*
SELECT $\bar{x}$ *,* $agg(y)$
WHERE $\{TGP\ .\ FILTER\}$
GROUP BY $\bar{x}$

In Example 4, we illustrate several cases of $TACQ$ by examples, beginning with the simplest case of a $TACQ$ to the most general case of a $TACQ$. The examples are built on the same RDF schema used for describing the data in Example 1.

Example 4.
$TACQ_1$ *is an example of a conjunctive query that presents the simplest case of a* $TACQ$*. It asks for the smart meter's number and the yearly income of each occupier.*

$TACQ_1$: a conjunctive query (without FILTER expression)

```
SELECT ?sm ?y
WHERE {?sm :associatedOccupier ?o . ?o :yearlyIncome ?y}
```

$TACQ_2$ *is an example of conjunctive query, the results are returned only if the given FILTER expression is satisfied. It asks for the smart meter's number and the yearly income of each occupier only if it is more than 50000.*

$TACQ_2$: a conjunctive query

```
SELECT ?sm ?y
WHERE {?sm :associatedOccupier ?o . ?o :yearlyIncome ?y .
FILTER (?y > 50000)}
```

$TACQ_3$ *is an example of an aggregated conjunctive query. It asks for the maximum number of persons per home with smart meters installed, grouped by yearly income.*

$TACQ_3$: an aggregated conjunctive query

```
SELECT ?y  MAX(?n)
WHERE {?sm :associatedOccupier ?o . ?o :numberOfPersons ?n .
?o :yearlyIncome ?y}
GROUP BY ?y
```

$TACQ_4$ *presents a case of a* sliding *time window of 6 hours. It asks for the sum of of energy consumption, computed every hour over the measurements of the previous 6 hours and grouped by smart meter's number, and the time window end.*

$TACQ_4$: a simple TACQ with sliding time window

```
SELECT ?sm ?timeWindowEnd SUM(?consumption)
WHERE {(?sm :consumption ?consumption, ?ts)}
GROUP BY ?sm ?timeWindowEnd
TIMEWINDOW (6h, 1h)
```

$TACQ_5$ presents a case of a tumbling *time window of 1 hour. It asks for the maximum energy consumption, computed over intervals of every hour and grouped by smart meter's number, yearly income and the time window end.*

$TACQ_5$: a general TACQ with tumbling time window

```
SELECT ?sm ?y ?timeWindowEnd MAX(?consumption)
WHERE {?sm :associatedOccupier ?o . ?o :yearlyIncome ?y .
(?sm :consumption ?consumption, ?ts)}
GROUP BY ?sm ?y ?timeWindowEnd
TIMEWINDOW (1h, 1h)
```

In our approach, we will often extract parts from a $TACQ$ and evaluate them independently to capture evaluation at different levels and to show their contribution to the evaluation of the query being considered. They are named as *conjunctive part* and *aggregated conjunctive part* and are defined as:

Definition 9 (Conjunctive part and aggregated conjunctive part)**.** *Let Q be a TACQ of the form:*
Q: SELECT $\bar{x}$ $agg(y)$
WHERE $\{TGP\ .\ FILTER\}$
GROUP BY $\bar{x}$
TIMEWINDOW (Size, Step)
The conjunctive part of Q, noted $Conj(Q)$, is defined as:
$Conj(Q)$: SELECT $\bar{x}$ y
WHERE $\{TGP\ .\ FILTER\}$
The aggregated conjunctive part Q, noted $AConj(Q)$, is an aggregated conjunctive query:
$AConj(Q)$: SELECT $\bar{x}$ $agg(y)$
WHERE $\{TGP\ .\ FILTER\}$
GROUP BY $\bar{x}$

3.3.2 Semantics of TACQs

The evaluation of a $TACQ$ over a temporal RDF graph TG is defined in terms of homomorphisms (Definition 10) and filtered homomorphisms (Definition 11).

The answers to a general $TACQ$ are obtained by iteratively evaluating the aggregated conjunctive part of the query over each time interval, which is computed from the values of $Size$ and $Step$ specified in the time window definition.

Definition 10 (Graph homomorphisms)**.** *Let H and H' be temporal RDF graphs or temporal graph patterns. An homomorphism from H' to H is an application $h : (I \cup L \cup B \cup Var) \rightarrow (I \cup L \cup B \cup Var)$ such that $h(c) = c$ for $c \in L \cup I$ and $h(H') \subseteq H$ where:*

$$h((s\,p\,o, t)) = (h(s)\,h(p)\,h(o), h(t))$$

Notation:
For a tuple of variables $\bar{x}$, $\mu(\bar{x})$ is the tuple obtained by replacing each variable x by its value $\mu(x)$.

For a FILTER, $\mu(FILTER)$ is obtained by replacing each variable x appearing in FILTER by its value $\mu(x)$. For example, if $FILTER = (x > 2)$ and $\mu(x) = 3$, then $\mu(FILTER) = (3 > 2)$, which is evaluated to True.

Definition 11 (Filtered homomorphisms)**.** *Let TGP and $FILTER$ be respectively a temporal graph pattern and a FILTER expression, and let TG a temporal graph. The set of filtered homomorphisms is the subset of homomorphisms μ from TGP to TG such that $\mu(FILTER) = True$.*

Remark 1. *When FILTER is empty, $\mu(FILTER) = True$ for each homomorphism and the set of filtered homomorphisms is thus equal to the full set of homomorphisms from TGP to any temporal graph TG.*

An answer to a conjunctive query against a data set is a tuple of values obtained by projecting the output variables of the query to values in the data for each (filtered) homomorphism mapping the temporal graph pattern in the query body to a subgraph in the data. We formalize this in Definition 12.

Definition 12 (Answers and support for a conjunctive query)**.** *Let CQ a conjunctive query: SELECT $\bar{x}$ WHERE $\{TGP\ .\ FILTER\}$.*
The answer set of CQ over a temporal graph TG, denoted Ans(CQ, TG), is defined as:

$Ans(CQ, TG) = \{\mu(\bar{x}) \mid \mu$ is a (filtered) homomorphism from TGP to $TG\}$
Let $\bar{a}$ an answer in $Ans(CQ, TG)$. Its (filtered) homomorphism support, denoted $Hsupport(\bar{a})$, is defined as:
$Hsupport(\bar{a}) = \{\mu \mid \mu$ is a (filtered) homomorphism from TGP to TG such that $\mu(\bar{x}) = \bar{a}\}$

For aggregated conjunctive queries, the computation of aggregate over the groups plays a vital role in obtaining an answer set. In the case of *aggregated conjunctive queries*, an answer set is obtained by using the complete set of (filtered) homomorphisms from temporal graph pattern to temporal RDF graph which is then partitioned into subgroups that correspond to the same assignment of the grouping variables and then in each subgroup, the aggregate function is applied to the set of data values corresponding to the aggregate variable. As formalized in Definition 13, for each group, an answer to an *aggregated conjunctive query*, is a concatenation of an answer obtained from its conjunctive part and a result obtained by applying an aggregated function to the data values in a group.

Definition 13 (Answers and support for an aggregated conjunctive query)**.** *Let ACQ an aggregated conjunctive query: SELECT $\bar{x}$ $agg(y)$ WHERE $\{$TGP. FILTER$\}$ GROUP BY $\bar{x}$.*
Let $Conj(ACQ)$ be the conjunctive part of ACQ. For each answer $\bar{a}$ of $Conj(ACQ)$ over a temporal graph TG, $Group(\bar{a})$ is defined as:
$Group(\bar{a}) = \{\mu(y) \mid \mu$ is a (filtered) homomorphism from TGP to TG, $\mu(\bar{x}) = \bar{a}\}$
The answer set of ACQ over a temporal graph TG, denoted Ans(ACQ, TG), is defined as:
$Ans(ACQ, TG) = \{(\bar{a}, agg(Group(\bar{a})) \mid \bar{a} \in Ans(Conj(ACQ))\}$
Let $(\bar{a}, r)$ an answer in $Ans(ACQ, TG)$. Its (filtered) homomorphism support, denoted $Hsupport\ (\bar{a}, r)$, is defined as:
$Hsupport(\bar{a}) = \{\mu \mid \mu$ is a (filtered) homomorphism from TGP to TG such that $\mu(\bar{x}) = \bar{a}\}$

The computation of the answer set of a $TACQ$ over a dataset TG requires the iteration of the evaluation of aggregated conjunctive part of $TACQ$ over as many time intervals of the form: $]now - k \times Step - Size, now - k \times Step]$, covering all the timestamps in the data set, where *now* denotes the timestamp when query is evaluated, and k denotes successive integers ranging from 0 to K_{max}, which is the minimal value of an integer k such that $now - k \times Step - Size$ is smaller than the smallest timestamp T_{min} in the dataset, i.e.: $K_{max} = \left\lceil \frac{now - T_{min} - Size}{Step} \right\rceil$.

In other words, the answer set of a $TACQ$ is the union of the answer sets of the *aggregated conjunctive part* of $TACQ$ evaluated over each time interval of the form: $]now - k \times Step - Size, now - k \times Step]$. Therefore, there are as many groups as (filtered) homomorphisms allowing to match the tuple of output variables of the query with tuple of values in the data multiplied by the number of time intervals. We formalize this in Definition 14.

Definition 14 (Answers and support for a temporal aggregated conjunctive query)**.** *Let $TACQ$ a temporal aggregated conjunctive query: SELECT $\bar{x}$ $agg(y)$ WHERE $\{TGP.\ FILTER\}$ GROUP BY $\bar{x}$ TIMEWINDOW (Size, Step).*
Let TG be a temporal graph and T_{min} the smallest timestamp in it. Let now be the timestamp at which the query is evaluated over TG and let $K_{max} = \left\lceil \frac{now - T_{min} - Size}{Step} \right\rceil$.
For each answer $\bar{a}$ of the conjunctive part $Conj(TACQ)$ of $TACQ$ over TG, for each integer k in $[0, K_{max}]$, $Group_k(\bar{a})$ is defined as:
$Group_k(\bar{a}) = \{\mu(y) \mid \mu$ *is a (filtered) homomorphism from TGP to TG,* $\mu(\bar{x}) = \bar{a}$, *and for each timestamp variable* $\mu(?ts) \in]now - k \times Step - Size, now - k \times Step]$ *and* $\mu(?timeWindowEnd) = now - k \times Step\}$
The answer set of $TACQ$ over TG, denoted $Ans(TACQ, TG)$, is defined as:
$Ans(TACQ, TG) = \{(\bar{a}, agg(Group_k(\bar{a})) \mid \bar{a} \in Ans(Conj(TACQ))\}$
Let $(\bar{a}, r)$ an answer in $Ans(TACQ, TG)$. Its (filtered) homomorphism support, denoted $Hsupport\ (\bar{a}, r)$, is defined as:
$Hsupport(\bar{a}, r) = \{\mu \mid \mu$ *is a (filtered) homomorphism from TGP to TG such that* $\mu(\bar{x}) = \bar{a}\}$

Example 5 shows the answers sets of $TACQ_1$ and $TACQ_5$ of Example 4 over the temporal RDF graph TG of Example 1.

Example 5.
The answer set of $TACQ_1$ is:

Answer set of $TACQ_1$ over TG

$Ans(TACQ_1, TG) =$ `{(`$:sm1031$`, 75000)}`

In case of $TACQ_1$ the homomorphism support of its single answer is restricted to the single homomorphism as shown below.

Homomorphism support of an answer of $TACQ_1$ over TG

$Hsupport($`:sm1031`$, 75000) = \{?sm/:sm1031, ?o/\_:o1031, ?y/75000\}$

$TACQ_5$ is evaluated at $2009-07-14T01:40:00$ *over temporal RDF graph TG of Example 1, its answer set is:*

Answer set of $TACQ_5$ over TG

$Ans(TACQ_5, TG) =$ {(:sm1031, 75000, 2009−07−14T01:40:00, 143)
(:sm1031, 75000, 2009−07−14T00:40:00, 78)}

The homomorphism support of the answer (: $sm1031$, 75000, $2009-07-14T01:40:00$, 143*) is made of two homomorphisms which differ in the result computed for the variable ?consumption over time interval of 1 hour.*

Homomorphism support of an answer of $TACQ_5$ over TG

$Hsupport($:sm1031, 75000, 2009-07-14T01:40:00, 143) =
{?sm/:sm1031, ?o/_:o1031, ?y/75000,
?timeWindowEnd/2009−07−14T01:40:00, ?consumption/34,
?ts/2009−07−14T01:30:00},
{?sm/:sm1031, ?o/_:o1031, ?y/75000,
?timeWindowEnd/2009−07−14T01:40:00, ?consumption/143,
?ts/2009−07−14T01:00:00}

3.3.3 Comparison between temporal graph patterns

Definitions 16, 17 and 18 are introduced for comparing temporal graph patterns that will be used in theorems and proofs presented in Section 4.3. First, we will define *unifiable temporal graph patterns* that serve as a basis for Definitions 16 and 17.

Definition 15 (Unifiable temporal graph patterns)**.** *Let TGP_1 and TGP_2 two temporal graph patterns. TGP_1 and TGP_2 are unifiable if there exists a function s replacing variables from TGP_1 and TGP_2 by constants or by variables of TGP_1, such that $s(TGP_1) = s(TGP_2)$.*

Definition 16 (Overlapping temporal graph patterns)**.** *Let TGP_1 and TGP_2 two temporal graph patterns. TGP_1 and TGP_2 are overlapping if they contain subgraphs that are unifiable.*

Example 6. *The following two listings show two overlapping temporal graph patterns, where the unifiable subgraphs are indicated in bold.*

Temporal graph pattern TGP_1

?**sm** : **associatedOccupier** ?**o** . ?**o** : **yearlyIncome y** . ?**o** : **numberOfPersons** ?**n**

Temporal graph pattern TGP_2

```
(?sm1 :consumption ?c1, ?ts1) . ?sm1 : associatedOccupier ?o1 .
?o1 : yearlyIncome ?y1 . ?o1 : numberOfPersons ?n1 . ?o1 :owns ?s1
```

Definition 17 (Disjoint temporal graph patterns)**.** *Let TGP_1 and TGP_2 two temporal graph patterns. TGP_1 and TGP_2 are disjoint if they are not overlapping.*

Example 7. *The following two listings show two disjoint temporal graph patterns.*

Temporal graph pattern TGP_1

```
?sm :associatedOccupier ?o . ?o  :yearlyIncome  ?y .
?o :numberOfPersons ?n
```

Temporal graph pattern TGP_2

```
(?sm1 :consumption ?c1, ?ts1) . ?o1 :owns  ?s1
```

Definition 18 (Isomorphic temporal graph patterns)**.** *Let TGP_1 and TGP_2 two temporal graph patterns. TGP_1 and TGP_2 are isomorphic if there is a homomorphism h from TGP_1 to TGP_2 that is bijective.*

Example 8. *The following two listings show two isomorphic temporal graph patterns.*

Temporal graph pattern TGP_1

```
(?sm :consumption ?c, ?ts) . ?sm :associatedBuilding ?b
```

Temporal graph pattern TGP_2

```
(?sm1 :consumption ?c1, ?ts1) . ?sm1 :associatedBuilding ?b1
```

3.3.4 Implementation of TACQs

The specificity of $TACQs$ is that their answer sets depend on the time at which they are evaluated, which defines the time intervals on which the temporal data should be grouped based on the *Size* and *Step* specified in the time window definition (as formalized in Definition 14).

For example, for $TACQ_5$, evaluated over the temporal RDF graph TG of Example 1 at the timestamp 2009-07-14T01:40:00, given the value of Size = 1h and Step = 1h, there will be two time intervals (2009-07-14T00:39:00 to 2009-07-14T01:40:00

and 2009-07-13T23:39:00 to 2009-07-14T00:40:00), on which its *aggregated conjunctive part* $AConj(TACQ_5)$ (presented in following Example 9) will be evaluated.

Example 9. *This example presents the aggregated conjunctive part of* $TACQ_5$ *of Example 4.*

$AConj(TACQ_5)$: the aggregated conjunctive part of $TACQ_5$

```
SELECT ?sm ?y MAX(?consumption)
WHERE {?sm :associatedOccupier ?o . ?o :yearlyIncome ?y .
(?sm :consumption ?consumption, ?ts)}
GROUP BY ?sm ?y
```

The evaluation over each time interval of the *aggregated conjunctive part* of $TACQ$ can be implemented in SPARQL over reified data or by SPARQL-star [Arn+22] over RDF-star data, by mimicking each time interval with corresponding FILTER expressions. In Examples 10 and 11, the $AConj(TACQ_5)$ is extended with FILTER expressions in SPARQL and SPARQL-star, to capture the evaluation over same time intervals when $TACQ_5$ is evaluated at timestamp $2009-07-14T01:40:00$ over temporal RDF graph TG of Example 1.

Example 10. *The answer sets are obtained by evaluating each SPARQL query over temporal RDF graph TG of Example 1.*

$AConj(TACQ_5)$ extended with FILTER expressions in SPARQL

```
SPARQL query for covering first time interval:
SELECT ?sm ?y (2009-07-14T01:40:00 AS ?timeWindowEnd)
       (MAX(?consumption) AS ?c)
WHERE { ?sm :associatedOccupier ?o .
        o :yearlyIncome ?y .
        ?r1 rdf:type :TemporalStatement;
            rdf:subject ?sm ;
            :onPredicate :consumption ;
            rdf:object ?consumption ;
            :timestamp ?ts .
        FILTER (?ts > "2009-07-14T00:39:00" &&
        ?ts <= "2009-07-14T01:40:00")}
GROUP BY ?sm ?y ?imeWindowEnd

Evaluating the query for first time interval, we get:
{:sm1031, 75000, 2009-07-14T01:40:00, 143}
```

```
SPARQL query for covering second time interval:
SELECT ?sm ?y (2009-07-14T01:40:00 AS ?timeWindowEnd)
       (MAX(?consumption) AS ?c)
WHERE { ?sm :associatedOccupier ?o .
        o :yearlyIncome ?y .
        ?r1 rdf:type :TemporalStatement;
            rdf:subject ?sm ;
            :onPredicate :consumption ;
            rdf:object ?consumption ;
            :timestamp ?ts .
        FILTER (?ts > "2009-07-13T23:39:00" &&
        ?ts <= "2009-07-14T00:40:00") }
GROUP BY ?sm ?y ?imeWindowEnd

Evaluating the query for second time interval, we get:
{:sm1031, 75000, 2009-07-14T00:40:00, 78}
```

Example 11. *The answer sets are obtained by evaluating each SPARQL-star query over temporal RDF graph TG of Example 1.*

$AConj(TACQ_5)$ extended with FILTER expressions in SPARQL-star

```
SPARQL-star query for covering first time interval:
SELECT ?sm ?y (2009-07-14T01:40:00 AS ?timeWindowEnd)
       (MAX(?consumption) AS ?c)
WHERE { ?sm :associatedOcuppier ?o;
            :associatedOcuppier/:yearlyIncome ?y;
            :consumption ?consumption {| :timestamp ?ts |}
        FILTER (?ts > "2009-07-14T00:39:00" &&
        ?ts <= "2009-07-14T01:40:00") }
GROUP BY ?sm ?y

Evaluating the query for first time interval, we get:
{:sm1031, 75000, 2009-07-14T01:40:00, 143}

SPARQL-star query for covering second time interval:
SELECT ?sm ?y (2009-07-14T00:40:00 AS ?timeWindowEnd)
       (MAX(?consumption) AS ?c)
WHERE { ?sm :associatedOcuppier ?o;
            :associatedOcuppier/:yearlyIncome ?y ;
            :consumption ?consumption {| :timestamp ?ts |}
        FILTER (?ts > "2009-07-13T23:39:00" &&
        ?ts <= "2009-07-14T00:40:00") }
```

```
GROUP BY ?sm ?y

Evaluating the query for second time interval, we get:
{:sm1031, 75000, 2009-07-14T00:40:00, 78}
```

Chapter 4

Formal specification and verification of privacy risks

In the context of the Semantic Web of Things, the data of data producers transmitted to service providers in exchange for some services may reveal private information wanted by data producers to remain undisclosed. It is therefore of utmost importance to help data producers to keep the control on their data for their privacy protection while preserving the utility of disclosed data for service providers.

We approach the problem of utility-aware privacy preservation in the setting of applications where service providers (data consumers) request collecting data from data producers in order to perform aggregate data analytics for optimization or recommendation purposes.

The approach that we promote to face the privacy versus utility dilemma in this setting is summarized as follows:

- Data producers keep the control on the data they accept to transmit to the data consumer according to their own *privacy policy* (a set of privacy queries).
- The data consumer makes explicit *her/his utility policy* (a set of utility queries) and explain for which task or service s/he requests data from data producers.
- On the side of the data producer, we evaluate the privacy and utility policies

to detect privacy risks. If privacy policy is violated by utility policy, then the data producer gets an explanation that can be exploited later by data producer to find an acceptable privacy-utility trade-off.

In this chapter, we focus on the characterization and detection of privacy risks by evaluating privacy and utility policies. The chapter is organized as follows. In Section 4.1, we provide an illustrative scenario of our approach. In section 4.2, we provide query-based specification of privacy and utility policies and formal definition of privacy risk. In Section 4.3, we present the formal characterization of privacy risks. In Section 4.4, we present the algorithms designed and implemented to detect the privacy risks.

4.1 Illustrative scenario

We consider a use-case related to smart power grids, in which the data producers are customers with smart meters in their home. A service provider has a catalog of energy efficiency products (including energy efficient insulation, windows, appliances) and requests collecting data from all the customers to adapt the proposed services to the profile of each of them based on some personal data.

We assume that the service provider and the customers understand each other through a common vocabulary using the same ontology presented in Section 3.2.1.1. This shared vocabulary allows service providers to specify their data needs in a precise way through a set of *utility queries*, that can then be compared to a set of *privacy queries* that are defined and kept secret by each data producer to state the data that they do not want to disclose directly or indirectly.

Let us suppose that the service provider has the following data needs:
(1) for each identifier of customers, their smart meter number and number of persons at home;
(2) for each identifier of customers that are owners of their home, their yearly income if it is more than 75000;
(3) for each smart meter number, the sum of consumptions computed every hour over the meter readings of the previous 3 hours.
These needs can be translated into the utility queries shown below by using SPARQL-like query language.

The utility queries into SPARQL-like query language

```
UQ1: SELECT ?sm ?o ?n
     WHERE { ?sm :associatedOccupier ?o . ?o :numberOfPersons ?n }
UQ2: SELECT ?o ?y
     WHERE { ?o :yearlyIncome ?y . ?o issda:owns ?s.
     FILTER(?y > 75000) }
UQ3: SELECT ?sm ?timeWindowEnd SUM(?c)
     WHERE { (?sm :consumption ?c, ?ts) }
     GROUP BY ?sm ?timeWindowEnd
     TIMEWINDOW (3h, 1h)
```

Now, suppose that a given customer, possibly with the help of privacy officer or tool, states that, among the data they accept to transmit, they want to prevent:
(1) the association between their smart meter number and their yearly income;
(2) the disclosure of their energy consumption measurements aggregated over intervals of 6 hours.
This can be translated into the following privacy queries for which no answer should be transmitted or inferred by any external data consumer.

The privacy queries of a given customer

```
PQ1: SELECT ?sm ?y
     WHERE { ?sm :associatedOccupier ?o . ?o :yearlyIncome ?y }
PQ2: SELECT ?timeWindowEnd SUM(?c)
     WHERE { (?sm :consumption ?c , ?ts) }
     GROUP BY ?timeWindowEnd
     TIMEWINDOW (6h, 6h)
```

With our approach, as it will be explained in Sections 4.3 and 4.4, we can detect privacy risks raised by the above utility queries, and provide the following privacy diagnosis:
(1) the first privacy risk is due to the possible violation of the privacy query PQ1 by the combination of answers to the utility queries UQ1 and UQ2.
(2) the second privacy risk is due to the possible violation of the privacy query PQ2 by answers to the utility query UQ3 because:
(a) PQ2 and UQ3 compute the same aggregate under the same conditions;
(b) groups of UQ3 are partitions of groups of PQ2;
(c) and finally, all time windows of PQ2 can be obtained by disjoint union of some time windows of UQ3.

On the basis of above reasoning, the data producer can try to negotiate with the

service provider by:
(1) refusing to provide an answer for the required customer's identifier in one of the utility queries UQ1 or UQ2;
(2) accepting to answer UQ3 if the time window is modified, for example by changing the step between each consumption computation from 1 hour to 2 hours.

4.2 Query-based specification of utility and privacy

The utility and privacy policies are defined as TACQs expressed in a common vocabulary or ontology, but their *fulfillment* has different semantics made explicit in the following definitions.

Definition 19 (Utility policy)**.** *A utility policy is a set of TACQ queries, called utility queries. A utility policy, issued by a data consumer, is satisfied by a data producer if s/he accepts to answer all of the utility queries on any of her/his local data set.*

Definition 20 (Privacy policy)**.** *A privacy policy is a set of TACQ queries, called privacy queries. A privacy policy, specific to each data producer, is satisfied if there is no risk that any answer of any privacy query over any local data set is disclosed through query answering.*

Forbidding answering privacy queries is not enough to guarantee that a privacy policy is satisfied because answers to a privacy query can be inferred from answers to other queries (such as utility queries). We focus on detecting such privacy risks that are formally defined in Definition 23 as the possibility of inferring an answer of a privacy query from answers to some utility queries, based on the query expressions only.

The inference problem of an answer (to a query Q) from a set of answers (to other queries) can be formalized as a logical inference problem based on the notion of logical signature of an answer that is defined as the logical formula characterizing all the (unknown) temporal data graphs leading to an answer for Q by interpreting temporal graph pattern as the logical conjunction of atomic formulas. For defining the logical signature of an answer of Q, we consider a partial instantiation of variables appearing in the logical formula, focusing solely on assigning the output variables of Q to the corresponding constants in the given answer within the logical

formula. In Definitions 21 and 22, we define the logical signatures of an answer of a query when it is a conjunctive query or a (temporal) aggregated conjunctive query.

Definition 21 (Logical signature of an answer of a conjunctive query)**.** *Let CQ be a conjunctive query: SELECT $\bar{x}$ WHERE $\{TGP$. $FILTER\}$.*
For an answer $\bar{a}$ to a query CQ, let $\phi_{\bar{a}}$ be the partial instantiation that assigns each output variable x in $\bar{x}$ to the corresponding constant a in $\bar{a}$.
The logical signature of an answer of CQ, denoted $\sigma(\bar{a}, CQ)$, is the formula:
$(\exists \bar{z}\, \phi_{\bar{a}}(TGP) \wedge \phi_{\bar{a}}(FILTER))$
where $\bar{z}$ is the (possibly empty) subset of variables in TGP not including the output variables $\bar{x}$.

Definition 22 (Logical signature of an answer of a (temporal) aggregated conjunctive query)**.** *Let Q be a (temporal) aggregated conjunctive query: SELECT $\bar{x}$ $agg(y)$ WHERE $\{TGP$. $FILTER\}$ GROUP BY $\bar{x}$ TIMEWINDOW(Size, Step).*
For an answer $(\bar{a}, r)$ to a query Q, let $\phi_{\bar{a}}$ be the partial instantiation that assigns each output variable x in $\bar{x}$ to the corresponding constant a in $\bar{a}$.
The logical signature of an answer of Q, denoted $\sigma((\bar{a}, r), Q)$, is the formula:
$(\exists y \exists \bar{z}\, \phi_{\bar{a}}(TGP) \wedge \phi_{\bar{a}}(FILTER)) \wedge agg(\{y | \exists \bar{z},\, \phi_{\bar{a}}(TGP) \wedge \phi_{\bar{a}}(FILTER)\}) = r$
where $\bar{z}$ is the (possibly empty) subset of variables in TGP not including the output variables $\bar{x}$ and aggregate variable y.

Definition 23 formalizes privacy risk as the possibility of inferring answers of a privacy query from the answers of utility queries on the same data graph *without knowing it.*

Definition 23 (Privacy risk)**.** *A utility policy raises a privacy risk for a privacy policy if the logical signature of an answer to a privacy query is entailed by the conjunction of logical signatures of answers of utility queries.*

The logical signatures of the respective answers `(:sm1031, 75000)` and `(:sm1031, 75000, 2009-07-14T01:40:00, 143)` to the $TACQ_1$ and the $TACQ_5$ of Example 4 are given below. The logical signature of the answer of $TACQ_5$ logically entails the logical signature of the answer of $TACQ_1$ and this entailment is highlighted in bold.

Logical signature of the answer (:sm1031, 75000) of $TACQ_1$

σ`((:sm1031, 75000), `$TACQ_1$`):`
$\exists$?**o** : **sm1031** : **associatedOccupier** ?**o** $\wedge$?**o** : **yearlyIncome** **75000**

Logical signature of the answer (: $sm1031$, 75000, 2009-07-14T01:40:00, 143) of $TACQ_5$

σ((:sm1031, 75000, 2009-07-14T01:40:00, 143), $TACQ_5$):
$\exists ?consumption\ \exists ?ts\ \exists ?\mathbf{o}$: **sm1031** : **associtaedOccupier** ?**o**
$\wedge$?**o** : **yearlyIncome 75000** $\wedge$ (:sm1031 :consumption ?*consumption*, ?*ts*)
$\wedge$?*ts* $\leq$ 2009-07-14T01:40:00 $\wedge$?*ts* $>$ 2009-07-14T00:40:00
$\wedge$ MAX {?*consumption* | $\exists ?ts\ \exists ?o$, :sm1031 :associatedOccupier ?*o*
$\wedge$?*o* :yearlyIncome 75000 $\wedge$ (:sm1031 :consumption ?*consumption*, ?*ts*)
$\wedge$?*ts* $\leq$ 2009-07-14T01:40:00 $\wedge$?*ts* > 2009-07-14T00:40:00} = 143

4.3 Formal characterization of privacy risks

We characterize privacy risks by independently evaluating each privacy query of a given privacy policy against the given set of utility queries (defining the utility policy). In our approach, privacy risks are characterized by distinguishing the cases when a privacy query is a conjunctive query or an aggregated conjunctive query or a temporal aggregated conjunctive query. The characterization of privacy risks in all cases is illustrated with the help of examples and in each example privacy and utility queries are built using the same ontology presented in Section 3.2.1.1.

This section is structured as follows. In section 4.3.1, we provide the characterization of privacy risk when a privacy query is a conjunctive query. In section 4.3.2, we provide the characterization of privacy risk when a privacy query is an aggregated conjunctive query. In Section 4.3.3, we provide the characterization of privacy risk when a privacy query is a temporal aggregated conjunctive privacy query.

4.3.1 Characterizing privacy risk for a conjunctive privacy query

In this section, we provide the full characterization of privacy risk when a privacy query Q_p is a conjunctive query. In this case, we characterize privacy risk for Q_p by evaluating it against the given set of utility queries.

Without loss of generality, by renaming variables within each query, we consider that queries have no variable in common. We will use the following notations for a conjunctive privacy query:

Q_p: SELECT $\bar{x}_p$
WHERE $\{TGP_p\ .\ FILTER_p\}$

We will use the following notations for the utility queries:

Q_{u_i}: SELECT $\bar{x}_{u_i}\ agg_{u_i}(y_{u_i})$
WHERE $\{TGP_{u_i}\ .\ FILTER_{u_i}\}$
GROUP BY $\bar{x}_{u_i}$
TIMEWINDOW ($Size_{u_i}$, $Step_{u_i}$)

Theorem 4.3.1 relies on evaluating the conjunctive privacy query Q_p on all the (small) temporal data graphs that are representative of the different ways of joining answers of utility queries. Each of these data graph is obtained by *freezing* (Definition 24) the variables in the union of temporal graph patterns in the utility queries, in a way that allows to replace distinct output variables of utility queries with a same constant (in order to mimic possible joins between output variables coming from different utility queries).

Definition 24 (Freezing of graph patterns)**.** *Let TGP a temporal graph pattern. A freezing of variables in TGP, denoted $freeze(TGP)$, is a temporal graph obtained from TGP by replacing each variable by a constant, such that every variable that is not an output variable is replaced by a distinct constant.*

Theorem 4.3.1 also verifies the entailment of the FILTER conditions of Q_p (if FILTER is not empty) by the FILTER conditions of utility queries. When conjunction of FILTER conditions of the privacy queries and utility queries is just satisfiable (Definition 25), we obtain a characterization of weak privacy risk (Definition 27). The weak privacy risk prevents the possibility to get an answer to a privacy query among the answers inferred by variants of utility queries. A variant of a query (Definition 26) differs from the original query only by the FILTER condition, while preserving compatibility with a new FILTER variant.

Definition 25 (Satisfiable Boolean expression)**.** *A Boolean expression Exp is satisfiable if there exists at least an assignment of variables in Exp that makes it* `TRUE`*.*

Definition 26 (Variant of a query)**.** *Let Q be a (temporal aggregated) conjunctive query: SELECT $\bar{x}$ $agg(y)$ WHERE $\{TGP\ .\ FILTER\}$ GROUP BY $\bar{x}$ TIMEWINDOW (Size, Step).*
A query Q' is a variant of Q if the two queries differ only on their FILTER part and $FILTER_Q \wedge FILTER_{Q'}$ is satisfiable.

Definition 27 (Weak privacy risk)**.** *A utility policy raises weak privacy risk for a privacy policy if a privacy risk is raised by replacing some utility queries with one of their variants.*

Theorem 4.3.1 (Characterizing privacy risk for a conjunctive privacy query raised by utility queries)**.** *A set of utility queries Q_{u_1}, ..., Q_{u_n} can raise* (1) a privacy risk *or* (2) a weak privacy risk *for a conjunctive privacy query Q_p* if and only if *there exists a freezing of the variables in $\bigcup_{i\in[1..n]} TGP_{u_i}$, and an answer $\bar{c} = h(\bar{x}_p)$ of Q_p over $freeze(\bigcup_{i\in[1..n]} TGP_{u_i})$ and* if Q_p has a FILTER condition*:*
1) $freeze(\bigwedge_{i\in[1..n]} FILTER_{u_i}) \models h(FILTER_p)$
or
2) $freeze(\bigwedge_{i\in[1..n]} FILTER_{u_i}) \wedge h(FILTER_p)$ *is satisfiable.*

Proof. **We first prove the case (1) of the theorem (privacy risk):**
If utility queries raise a privacy risk for Q_p, it means by Definition 21 that there exists tuples of constants $\bar{a}$, $\bar{a}_1, ..., \bar{a}_n$ and partial instantiations $\phi_{\bar{a}}$, $\phi_{\bar{a}_1}$,...,$\phi_{\bar{a}_n}$ that assigns each output variable in $\bar{x}_p$, $\bar{x}_{u_1}$,...,$\bar{x}_{u_n}$ to the corresponding constant in $\bar{a}$, $\bar{a}_1, ..., \bar{a}_n$ such that $\exists \bar{z}_1 ... \exists \bar{z}_n\, \phi_{\bar{a}_1}(TGP_{u_1}) \wedge \phi_{\bar{a}_1}(FILTER_{u_1}) \wedge ... \wedge \phi_{\bar{a}_n}(TGP_{u_n}) \wedge \phi_{\bar{a}_n}(FILTER_{u_n}) \models \exists \bar{z}_p\, \phi_{\bar{a}}(TGP_p) \wedge \phi_{\bar{a}}(FILTER_p)$.
Since the sets of variables in each query are pairwise disjoint, the entailment is only possible if there exists a filtered homomorphism h from the variables in $\bar{z}_p$ to the variables or constants in the left hand side so that all the atoms in $h(\phi_{\bar{a}}(TGP_p))$ appear in the union of the atoms in $\phi_{\bar{a}_1}(TGP_{u_1}) \wedge ... \wedge \phi_{\bar{a}_n}(TGP_{u_n})$, and $h(\phi_{\bar{a}}(FILTER_p))$ is entailed by $\phi_{\bar{a}_1}(FILTER_{u_1}) \wedge ... \wedge \phi_{\bar{a}_n}(FILTER_{u_n})$.
Let $Frozen$ be the result on $\bigcup_{i\in[1..n]} TGP_{u_i}$ of the freezing that replaces each output variable x_{u_i} by $\phi_{\bar{a}_i}(x_{u_i})$. The filtered homomorphism $h \cup \phi_{\bar{a}}$ from the temporal graph pattern TGP_p to $Frozen$ allows to show that $\bar{a}$ is an answer of Q_p when evaluated over $Frozen$, and: $freeze(\bigwedge_{i\in[1..n]} FILTER_{u_i}) \models h \cup \phi_{\bar{a}}(FILTER_p)$.
For the converse way of the proof, let us consider $Frozen$ the result on $\bigcup_{i\in[1..n]} TGP_{u_i}$ of a freezing $freeze$ of the output variables such that there exists an answer $\bar{c}$ for Q_p when evaluated over $Frozen$ with a filtered homomorphism support h such that $h(\bar{x}_p) = \bar{c}$ and $freeze(\bigwedge_{i\in[1..n]} FILTER_{u_i}) \models h(FILTER_p)$.
The filtered homomorphism h allows to show the entailment between the formulas δ_1: $\exists \bar{z}_u\, Frozen \wedge freeze(\bigwedge_{i\in[1..n]} FILTER_{u_i})$ and δ_2: $\exists \bar{z}_p\, h_{\bar{c}}(TGP_p) \wedge h_{\bar{c}}(FILTER_p)$ where TGP_p and $Frozen$ are interpreted as the conjunction of their respective temporal patterns seen as logical atoms, and $h_{\bar{c}}$ is the restriction of h to the output variables of Q_p.
In fact, δ_1 and δ_2 are respectively the conjunction of logical signatures of the an-

swers $freeze(\bar{x}_{u_i})$ of each Q_{u_i}, and the logical signature of the answer $\bar{c}$ of Q_p. Therefore, the utility queries raise privacy risk for privacy query Q_p.

We now prove the case (2) of the theorem (weak privacy risk):
If the set of utility queries raise a weak privacy risk for Q_p, a set of variants Q'_{u_i} of the utility queries Q_{u_i} raise privacy risk for Q_p. By applying the case (1) of the theorem, there exists a freezing $freeze$ of the output variables in $\bigcup_{i\in[1..n]} TGP'_{u_i}$, and an answer $\bar{c}$ of Q_p over $Frozen = freeze(\bigcup_{i\in[1..n]} TGP'_{u_i})$ with a filtered homomorphism support h such that:
$freeze(\bigwedge_{i\in[1..n]} FILTER'_{u_i}) \models h(FILTER_p)$
This means that every variable assignment satisfying $freeze(\bigwedge_{i\in[1..n]} FILTER'_{u_i})$ satisfies $h(FILTER_p)$ too.
By definition of the variants (Definition 26), there exists a variable assignment satisfying both $freeze(\bigwedge_{i\in[1..n]} FILTER'_{u_i})$ and $freeze(\bigwedge_{i\in[1..n]} FILTER_{u_i})$.
This assignment satisfies $h(FILTER_p)$ too. Thus $freeze(\bigwedge_{i\in[1..n]} FILTER_{u_i}) \wedge h(FILTER_p)$ is satisfiable.
For the converse way, let us suppose that there exists a freezing $freeze$ of the output variables in $\bigcup_{i\in[1..n]} TGP_{u_i}$, and an answer $\bar{c}$ of Q_p over $Frozen = freeze(\bigcup_{i\in[1..n]} TGP_{u_i})$ with a filtered homomorphism support h such that:
$freeze(\bigwedge_{i\in[1..n]} FILTER_{u_i}) \wedge h(FILTER_p)$ is satisfiable.
The goal is to build variants Q'_{u_i} of utility queries by adding to the FILTER constraints $FILTER_{u_i}$ some constraints making $h(FILTER_p)$ true.
For doing so, first we remark that each freezing satisfying the conditions of the theorem can be constrained by equating freezing constants for getting a *connected* freezing satisfying also the conditions of the theorem. A freezing is connected if each single TGP_{u_i} has a fresh constant in common with the freezing of at least another TGP_{u_j}. Then:
- for each atomic comparison t comp t' in $h(FILTER_p)$ such that t and t' are either numbers or terms in the freezing of a single TGP_{u_i}, we add to $FILTER_{u_i}$ the atomic constraint obtained by defreezing the constants possibly involved in t *comp* t'.
- for each atomic comparison t *comp* t' in $h(FILTER_p)$ such that t and t' are not in the freezing of single TGP_{u_i}, we can build a chain of comparisons t_0 *comp* t_1, ...,t_{k-1} *comp* t_k where $t_0 = t$ and $t_k = t'$ where each pair t_j, t_{j+1} are terms appearing in the freezing of single TGP_{u_j}. We just have to add to each $FILTER_{u_j}$ the atomic constraint obtained by defreezing the constants possibly involved in t_j *comp* t_{j+1}. □

Example 12. *Let us consider the following privacy and utility queries $PQ1$, $UQ1$ and $UQ2$, corresponding to the first privacy query and the first two utility queries (up to variable renaming) of the scenario illustrated in Section 4.1:*
$PQ1$: SELECT ?sm ?y
WHERE { ?sm :associatedOccupier ?o . ?o :yearlyIncome ?y }
$UQ1$: SELECT ?x1 ?y1 ?n
WHERE { ?x1 :associatedOccupier ?y1 . ?y1 :numberOfPersons ?n }
$UQ2$: SELECT ?x2 ?y2
WHERE { ?x2 :yearlyIncome ?y2 .
x2 :owns ?z1 . FILTER (?y2 > 75000) }
The following $Frozen$ and $Frozen'$ are different freezing of the variables in the union of the temporal graph patterns of utility queries, where the constants corresponding to the freezing of output variables are denoted by constants oc_i:
$Frozen = \{oc_1$:associatedOccupier oc_2 . oc_2 :numberOfPersons oc_3 .
oc_4 :yearlyIncome oc_5 . oc_4 :owns $c_6\}$
obtained by the freezing: { ?x1/oc_1, ?y1/oc_2, ?n/oc_3, ?x2/oc_4, ?y2/oc_5, ?z1/c_6 }
$Frozen' = \{oc_1$:associatedOccupier oc_2 . oc_2 :numberOfPersons oc_3 .
oc_2 :yearlyIncome oc_4 . oc_2 :owns $c_5\}$
obtained by a freezing: { ?x1/oc_1, ?y1/oc_2, ?n/oc_3, ?x2/oc_2, ?y2/oc_4, ?z1/c_5 } in which ?y1 and ?x2 that are output variables in each of the utility queries are frozen to the same constant oc_2.
$Ans(PQ1, Frozen)$ is empty but $Ans(PQ1, Frozen') = \{(oc_1, oc_4)\}$.
The conditions for case 1 of the Theorem 4.3.1 are satisfied. Thus, the privacy risk for $PQ1$ is raised by the two utility queries $UQ1$ and $UQ2$.
It is important to note that when each of the utility query is considered in isolation with $PQ1$ then no privacy risk is raised.

Example 13. *Let us consider the same queries $PQ1$, $UQ1$ and $UQ2$ of previous example 12 by adding a FILTER condition to the $PQ1$ as follows:*
$PQ1'$: SELECT ?sm ?y
WHERE { ?sm :associatedOccupier ?o . ?o :yearlyIncome ?y .
FILTER (?y > 80000)}
As $Ans(PQ1', Frozen') = \{(oc_1, oc_4)\}$. Now we also need to check FILTER conditions. The output variable ?y2 of $UQ2$ is frozen to oc_4. So we get:
freeze(?y2 > 75000) = (oc_4 > 75000)
The homomorphism support h of the answer of $PQ1'$ over $Frozen'$ we get:

h(?y > 80000)= (oc_4 > 80000)
Checking (oc_4 > 80000) is entailed by (oc_4 > 75000) is enough to prove that a privacy risk is raised for $PQ1'$ by the two utility queries $UQ1$ and $UQ2$ and the conditions for case 1 of the Theorem 4.3.1 are satisfied.
Now let us modify the FILTER condition of the utility query $UQ2$ with (?y2 < 85000).
Checking (oc_4 > 80000) is satisfiable with (oc_4 < 85000) is enough to prove that a weak privacy risk for $PQ1'$ is raised by the two utility queries $UQ1$ and variant of $UQ2$ and the conditions for case 2 of the Theorem 4.3.1 are satisfied.
Now let us consider $PQ1$, $UQ2$ and the following $UQ1'$ as privacy and utility queries. $UQ1'$ is obtained by reducing one of the output variable ?y1 in $UQ1$.
$UQ1'$: SELECT ?x1 ?n
WHERE { ?x1 :associatedOccupier ?y1 . ?y1 :numberOfPersons ?n }
In this case no freezing of the output variables ?x1 and ?n of $UQ1'$ combined with a freezing of the output variables of $UQ2$ can lead to an answer of $PQ1$ evaluated over the resulting temporal graph which is of the form:
{oc_1 :associatedOccupier c_2. c_2 :numberOfPersons oc_3 . oc_4 :yearlyIncome oc_5 . oc_4 :owns c_6}
Therefore no privacy risk for $PQ1$ is raised by the two utility queries $UQ1'$ and $UQ2$.

Worst-case complexity: In the worst case, detecting a privacy risk using Theorem 4.3.1 requires to evaluate a conjunctive privacy query (without FILTER) Q_p over the temporal graph $Frozen$ resulting from all the possible freezing of the output variables of the utility queries. The evaluation of Q_p over $Frozen$ is polynomial in the size of the utility queries but the number of possible freezing is 2^{OV_u} where OV_u is the number of output variables of the utility queries.

In practice: In fact, each freezing can be obtained from the initial most general freezing, which assigns each output variable to a distinct fresh constant and by equating subsets of these constants. The choice of constants to equate is constrained by the join variables within the conjunctive privacy query (without FILTER) to obtain an answer.

4.3.2 Characterizing privacy risk for an aggregated conjunctive privacy query

In this section, we provide the characterization of a (weak) privacy risk when a privacy query Q_p is an aggregated conjunctive query. In this case, we separately consider the utility queries without time window definitions from the utility queries with time window definitions and characterize a (weak) privacy risk for Q_p by evaluating it against a set of utility queries without time window definitions. For the general case of TACQs addressed in Section 4.3.3, the following Theorem 4.3.2 is applied only to evaluate the aggregated conjunctive part of a temporal aggregated conjunctive privacy query against the set of aggregated conjunctive parts of all utility queries for characterizing a (weak) privacy risk .

Without loss of generality, by renaming variables within each query, we consider that queries have no variable in common. We will use the following notations for an aggregated conjunctive privacy query:

Q_p: SELECT $\bar{x}_p$ $agg_p(y_p)$
WHERE $\{TGP_p$. $FILTER_p\}$
GROUP BY $\bar{x}_p$

We will use the following notations for the utility queries:

Q_{u_i}: SELECT $\bar{x}_{u_i}$ $agg_{u_i}(y_{u_i})$
WHERE $\{TGP_{u_i}$. $FILTER_{u_i}\}$
GROUP BY $\bar{x}_{u_i}$

In the case of conjunctive queries, an answer can be inferred if there exists a (filtered) homormorphism assigning variables in the temporal graph pattern to values in the temporal data graph whereas in the case of aggregated conjunctive queries, the computation of an answer necessitates the construction of the complete set of (filtered) homomorphisms from the temporal graph pattern to values in the temporal data graph which is then partitioned into subgroups that correspond to the same assignment of the grouping variables. Finally, within each subgroup, the aggregate function is applied to the set of data values corresponding to the aggregate variable. Therefore, to infer an answer of an aggregated query Q_p from the answers of utility queries, it is necessary that a subset S of the utility queries computes a equivalent set of (filtered) homomorphisms than Q_p for this answer. According to [Coh05], it is necessary that the existence of an *isomorphism* can be enforced between the temporal graph pattern of the privacy query and the

union of the temporal graph patterns from S. This is the core of the privacy risk characterization stated in Theorem 4.3.2.

Theorem 4.3.2 (Characterizing privacy risk for an aggregated conjunctive privacy query)**.** *Let Q_p be an aggregated conjunctive privacy query. There exists a (weak) privacy risk for Q_p* if and only if *there exists a subset S of utility queries such that:*
1) *the set of conjunctive parts of all utility queries in S raises a (weak) privacy risk for the conjunctive part of Q_p,*
and
2) *the union of the temporal graph patterns of the utility queries in S is isomorphic to TGP_p (through an isomorphism I), or can be made isomorphic to TGP_p (through an isomorphism I) by replacing some output variables by constants and/or by equating some output variables of some utility queries in S,*
and either
3.1) $I(\bar{x}_p \cup \{y_p\}) \subseteq \bigcup_{Q_{u_i} \in S} \bar{x}_{u_i}$,
or
3.2) *the subset S contains at least one aggregated conjunctive query Q_{u_i} where $I(\bar{x}_p) \subseteq \bar{x}_{u_i}$ and $agg_p = agg_{u_i}$ and $I(y_p) = y_{u_i}$.*

Proof. Knowing the way aggregates are computed, S raises a (weak) privacy risk for Q_p if it first computes the same set of filtered homomorphisms as Q_p. This means that the set of conjunctive parts of queries in S raises a (weak) privacy risk for the conjunctive part of Q_p, thus satisfying the condition 1 of the theorem.
In this case, there exists at least one way to join the temporal graph patterns of the utility queries in S using only their output variables such as there exists an homomorphism between this joining graph J and the temporal graph pattern TGP_p of Q_p.
Moreover, [Coh05] has showed that utility queries in S joined according to J can produce the same set of filtered homomorphisms if and only if J and TGP_p are isomorphic or can be made isomorphic by replacing some of the output variables in J by constants, thus satisfying condition 2 of the theorem.
There are only two ways to build an aggregate over a group of Q_p from results of the utility queries in S: either they produce directly the set of (non aggregated) needed values as results to compute the aggregate, or they produce partial aggregated values corresponding to the aggregate computation of Q_p but over subgroups of Q_p.
In the first case, $I(\bar{x}_p \cup \{y_p\}) \subseteq \bigcup_{Q_{u_i} \in S} \bar{x}_{u_i}$ (condition 3.1 of the theorem).

In the second case, there exist at least one aggregated conjunctive utility query Q_{u_i} in S computing the same aggregate ($agg_p = agg_{u_i}$) over the variable corresponding to y_p ($I(y_p) = y_{u_i}$) over subgroups defined by at least one supplementary output variable from $\bar{x}_p$ ($I(\bar{x}_p) \subseteq \bar{x}_{u_i}$), thus satisfying condition 3.2 of the theorem. □

Example 14. *Let us consider the following privacy query Q_p and utility queries Q_{u_1} and Q_{u_2}.*

Q_p: SELECT ?y MAX(?n)
WHERE { ?sm :associatedBuilding ?b . ?b rdf:type :Apartment .
?sm :associatedOccupier ?o . ?o :yearlyIncome ?y .
?o :numberOfpersons ?n }
GROUP BY ?y

Q_{u_1}: SELECT ?sm1 ?o1 ?y1 ?a1
WHERE { ?sm1 :associatedBuilding ?b1 . ?b1 rdf:type ?a1 .
?sm1 :associatedOccupier ?o1 . ?o1 :yearlyIncome ?y1 }

Q_{u_2}: SELECT ?o2 ?n1
WHERE { ?o2 :numberOfPersons ?n1 }

According to Theorem 4.3.1, the conjunctive parts of Q_{u_1} and Q_{u_2} raise privacy risk for the conjunctive part of Q_p, thus condition 1 of Theorem 4.3.2 is satisfied. By replacing the output variable ?a1 with the constant :Apartment and by equating the output variables ?o1 and ?o2, we enforce the isomorphism I between the temporal graph pattern of Q_p and the union of temporal graph patterns of Q_{u_1} and Q_{u_2}, we get:

$Join(Q_{u_1}, Q_{u_2})$: { ?sm1 :associatedBuilding ?b1 .
?b1 rdf:type :Apartment . ?sm1 :associatedOccupier ?o1
. ?o1 :yearlyIncome ?y1 . ?o1 :numberOfPersons ?n1 }

The condition 2 of the Theorem 4.3.2 is also satisfied.

$\bigcup_{Q_{u_i}} \bar{x}_{u_i}$= ?sm1 ?o1 ?y1 ?a1 ?n1

$I(\bar{x}_p \cup \{y_p\})$= ?y1 ?n1

$I(\bar{x}_p \cup \{y_p\}) \subset \bigcup_{Q_{u_i}} \bar{x}_{u_i}$, thus condition 3.1 of the Theorem 4.3.2 is also satisfied. Therefore, the privacy risk for Q_p is raised by the two utility queries Q_{u_1} and Q_{u_2}. Now let us consider Q_{u_2} and the following privacy query Q'_p and conjunctive utility query Q'_{u_1}.

Q'_p: SELECT ?y MAX(?n)
WHERE { ?sm :associatedBuilding ?b . ?b rdf:type :Apartment .
?sm :associatedOccupier ?o . ?o :yearlyIncome ?y .
?o :numberOfpersons ?n . FILTER (?y > 60000)

GROUP BY ?y

Q'_{u_1}: *SELECT ?sm1 ?o1 ?y1 ?a1*
WHERE { ?sm1 :associatedBuilding ?b1 . ?b1 rdf:type ?a1 .
?sm1 :associatedOccupier ?o1 . ?o1 :yearlyIncome ?y1 .
FILTER(?y1 < 80000) }

Q'_p *and* Q'_{u_1} *only differ from* Q_p *and* Q_{u_1} *by the FILTER conditions.*
By considering the same join as in the previous case and by replacing the FILTER condition of Q'_{u_1} *by the FILTER condition of* Q'_p*, we enforce the conditions of the Theorem 4.3.2. Condition 1 is satisfied and ensures that the replacement of the FILTER condition in* Q'_{u_1} *leads to a (compatible) variant of* Q'_{u_1}*.*
This proves that a weak privacy risk for Q_p *is raised by the two utility queries* Q'_{u_1} *and* Q_{u_2}*.*

Example 15. *Let us consider the same* Q_p *of the previous Example 14 and the following aggregated conjunctive utility query* Q_{u_3}*.*

Q_{u_3}: *SELECT ?o1 ?y1 ?a1 MAX(?n1)*
WHERE { ?sm1 :associatedBuilding ?b1 . ?b1 rdf:type ?a1 .
?sm1 :associatedOccupier ?o1 . ?o1 :yearlyIncome ?y1 .
?o1 :numberOfpersons ?n1 }
GROUP BY ?o1 ?y1 ?a1

According to Theorem 4.3.1, the conjunctive part of Q_{u_3} *raises privacy risk for the conjunctive part of* Q_p*, thus condition 1 of Theorem 4.3.2 is satisfied.*
By replacing the output variable ?a1 *in* TGP_{u_3} *with the constant :Apartment, we enforce an isomorphism (I) between the temporal graph pattern of* Q_{u_3} *and the temporal graph pattern of* Q_p*. Condition 2 of the Theorem 4.3.2 is satisfied.*
Both queries Q_p *and* Q_{u_3} *compute the same aggregate* MAX*.*
Through an isomorphism I, we get:
$I(y_p)$=*?n1 and* $I(\bar{x}_p)$= *?y1.*
As y_{u_3} = *?n1 and* $\bar{x}_{u_3}$= *?o1 ?y1 ?a1, so* $I(y_p) = y_{u_3}$ *and* $I(\bar{x}_p) \subseteq \bar{x}_{u_3}$*, thus condition 3.2 of the Theorem 4.3.2 is also satisfied.*
Therefore, the privacy risk for Q_p *is raised by the utility query* Q_{u_3}*.*
Now let us consider Q_p *and the following aggregated conjunctive utility query* Q_{u_4}*:*

Q_{u_4}: *SELECT ?o1 ?n1 MAX(?y1)*
WHERE { ?sm1 :associatedBuilding ?b1 . ?b1 rdf:type :Apartment .
?sm1 :associatedOccupier ?o1 . ?o1 :yearlyIncome ?y1 .
?o1 :numberOfpersons ?n1 }
GROUP BY ?o1 ?n1

In this case, conditions 1 and 2 of the theorem are satisfied. Both queries Q_p and Q_{u_4} compute the same aggregate MAX.
Through isomorphism I, we get:
$I(y_p)$=?n1 and $I(\bar{x}_p)$= ?y1.
As y_{u_3} = ?y1 and $\bar{x}_{u_3}$= ?o1 ?n1 ?a1, so $I(y_p) \neq y_{u_3}$ and $I(\bar{x}_p) \nsubseteq \bar{x}_{u_3}$. Thus, condition 3.2 of the Theorem 4.3.2 is not satisfied.
Therefore, no privacy risk for Q_p is raised by the utility query Q_{u_4}.

4.3.3 Characterizing privacy risk for a temporal aggregated conjunctive privacy query

In this section, we provide the characterization of a privacy risk when a privacy query Q_p is a temporal aggregated conjunctive query (TACQ).

Without loss of generality, by renaming variables within each query, we will consider that queries have no variable in common. We will use the following notations for a temporal aggregated conjunctive privacy query:

Q_p: SELECT $\bar{x}_p$ $agg_p(y_p)$
WHERE $\{TGP_p$. $FILTER_p\}$
GROUP BY $\bar{x}_p$
TIMEWINDOW ($Size_p$, $Step_p$)

We will use the following notations for the utility queries:

Q_{u_i}: SELECT $\bar{x}_{u_i}$ $agg_{u_i}(y_{u_i})$
WHERE $\{TGP_{u_i}$. $FILTER_{u_i}\}$
GROUP BY $\bar{x}_{u_i}$
TIMEWINDOW ($Size_{u_i}$, $Step_{u_i}$)

The answers to the temporal aggregated conjunctive query $TACQ$ are obtained by iteratively evaluating the aggregated conjunctive part of the $TACQ$ over each time interval, which is computed from the values of $Size$ and $Step$ specified in the time window definition. Therefore, characterization of a privacy risk for Q_p is only possible if there exists a (weak) privacy risk for the *aggregated conjunctive part* of Q_p when evaluated against the set of aggregated conjunctive parts of utility queries and there is a possibility to build at least one time window of Q_p over which an aggregated result of Q_p can be computed from the answers of one or more utility queries.

In this section, we will focus on the cases where a (weak) privacy risk is characterized by finding a possibility to build at least one time window of Q_p from the answers of utility queries in subset S. In the first case, we evaluate if the answers to utility queries contain all the values required to build at least one time window of Q_p and compute an aggregate result over it. For this it is necessary that the timestamp values for all the dynamic properties in the temporal graph patterns of utility queries in S are included in the answers of the utility queries in S. In the second case, we evaluate if the answers of one or two different utility queries contain all the partial aggregate results computed over different time windows of one or two different utility queries that can be combined to obtain at least one aggregate result of Q_p computed over a single time window.

In the following sections, we will characterize a (weak) privacy risk for Q_p by considering the cases mentioned above. In Section 4.3.3.1, we will present the first case. In Sections 4.3.3.2 and 4.3.3.3, we will present the second case. The generalization of the case presented in Section 4.3.3.3, which involves the consideration of several temporal aggregated conjunctive utility queries is left for future work.

4.3.3.1 Privacy risk raised by a subset of utility queries

Theorem 4.3.3 characterizes a (weak) privacy risk for Q_p by extending the Theorem 4.3.2 and finds the possibility to compute at least one time window of Q_p from the answers of utility queries in a subset S by evaluating if the timestamp variables of all dynamic properties in the temporal graph patterns of utility queries in S are included in the output variables of the utility queries in S.

Theorem 4.3.3. *Let Q_p be a temporal aggregated conjunctive privacy query. There exists a (weak) privacy risk for Q_p* if and only if *there exists a subset S of utility queries such that:*
1) *the set of aggregated conjunctive parts of all utility queries in S raises a (weak) privacy risk for the aggregated conjunctive part of Q_p,*
2) *the timestamp variables of all dynamic properties in the temporal graph patterns of utility queries in S are included in $\bigcup_{Q_{u_i} \in S} \bar{x}_{u_i}$.*

Proof. To compute an aggregate result over a single time window of Q_p is possible by combining the answers of utility queries in S, which contain:
- the set of non aggregated values needed to compute the aggregate of Q_p by applying agg_p,
- the timestamp values for all the dynamic properties are included in the temporal

graph patterns of utility queries in S in such a way that these timestamp values cover exactly the time window size of Q_p over which an aggregated result of Q_p is being computed.
These values can only be obtained from the set of answers to utility queries in S *if*:
1) according to Theorem 4.3.2, the set of conjunctive parts of utility queries in S raises a (weak) privacy risk for the aggregated conjunctive part of Q_p,
2) the timestamp variables of all dynamic properties in the temporal graph patterns of utility queries in S are included in $\bigcup_{Q_{u_i} \in S} \bar{x}_{u_i}$.

□

Example 16. *Let us consider the following privacy query Q_p and utility queries Q_{u_1} and Q_{u_2}.*
Q_p: SELECT ?sm ?timeWindowEnd MAX(?c)
WHERE { ?sm :associatedBuilding ?b . ?b rdf:type :Apartment .
(?sm :consumption ?c, ?ts) }
GROUP BY ?sm ?timeWindowEnd
TIMEWINDOW (6h, 6h)
Q_{u_1}: SELECT ?sm1 ?b1
WHERE { ?sm1 :associatedBuilding ?b1 . ?b1 rdf:type :Apartment }
Q_{u_2}: SELECT ?sm2 ?c1 ?ts1
WHERE { (?sm2 :consumption ?c1, ?ts1) }
According to Theorem 4.3.2, the aggregated conjunctive parts of Q_{u_1} and Q_{u_2} raise privacy risk for the aggregated conjunctive part of Q_p, thus condition 1 of the Theorem 4.3.3 is satisfied.
A timestamp variable ?ts1 of Q_{u_2} is an output variable. This satisfies condition 2 of the Theorem 4.3.3.
Therefore, the privacy risk for Q_p is raised by the two utility queries Q_{u_1} and Q_{u_2}. Now let us modify Q_{u_2} by removing the timestamp variable from its output variables. Q'_{u_2}: SELECT ?sm2 ?c1
WHERE { (?sm2 :consumption ?c1, ?ts1) }
In this case, the second condition of the Theorem 4.3.3 is not satisfied, so no privacy risk for Q_p is raised by the two utility queries Q_{u_1} and Q'_{u_2}.

4.3.3.2 Privacy risk raised by a subset containing only one TACQ

Now we will focus on the cases where a (weak) privacy risk is characterized by finding a possibility to build at least one time window of Q_p from the union of

some time windows of a single temporal aggregated conjunctive utility query Q_{u_i} of subset S that cover exactly one time window of Q_p. First, we will consider a case when Q_p and Q_{u_i} have *same time window definitions* (i.e; $Size_p = Size_{u_i}$ and $Step_p = Step_{u_i}$) and then we will consider a case when both Q_p and Q_{u_i} have *different time window definitions*.

In the first case, when the *aggregated conjunctive part* of Q_{u_i} raises a (weak) privacy risk for the *aggregated conjunctive part* of Q_p and both Q_p and Q_{u_i} have *same time window definitions*, the answer sets of Q_p and Q_{u_i} are computed by iterating over the same time intervals so all the time windows of Q_p can be obtained from the time windows of Q_{u_i}. Therefore, Theorem 4.3.2 is sufficient to characterize a privacy risk for Q_p.

Now we will consider the second case, when the *aggregated conjunctive part* of Q_{u_i} raises a (weak) privacy risk for the *aggregated conjunctive part* of Q_p and both Q_p and Q_{u_i} have *different time window definitions*. In this case, a (weak) privacy risk for Q_p is raised only if it is possible to build at least one time window I_p of Q_p from the union of some time windows I_{u_x} of Q_{u_i} as shown in Figure 4.1. Figure 4.1 illustrates that the aggregates SUM and $COUNT$ necessitate to build a time window of Q_p from the disjoint union of time windows I_{u_1} and I_{u_4} of Q_{u_i} to avoid double counting of an overlap whereas for the aggregates MAX and MIN a time window of Q_p can be built either from the disjoint union of time windows I_{u_1} and I_{u_4} of Q_{u_i}, or from the overlapping union of time windows I_{u_1}, I_{u_2}, I_{u_3}, and I_{u_4} of Q_{u_i}.

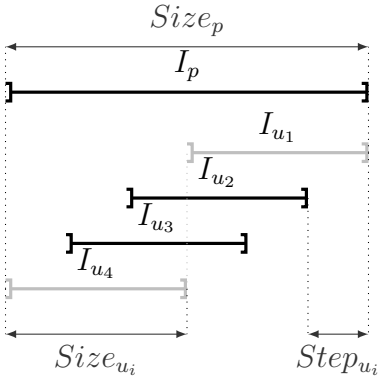


Figure 4.1: Union of time windows of a single utility query

Theorem 4.3.4 characterizes a (weak) privacy risk for Q_p by evaluating the values of $Size_p$, $Step_p$, $Size_{u_i}$ and $Step_{u_i}$ specified in the time window definitions of Q_p

and Q_{u_i} and finds the possibility to build at least one time window of Q_p from the union of some time windows of Q_{u_i} that cover exactly one time window of Q_p.

Theorem 4.3.4 (Characterizing privacy risk for a temporal aggregated conjunctive privacy query raised by a subset containing only one TACQ that computes the same aggregate as the temporal aggregated conjunctive privacy query)**.** *Let Q_p be a temporal aggregated conjunctive privacy query. A subset S containing only one temporal aggregated conjunctive utility query Q_{u_i} raises a (weak) privacy risk for Q_p* if and only if*:*
1) *the aggregated conjunctive part of Q_{u_i} raises a (weak) privacy risk for the aggregated conjunctive part of Q_p,*
and
2.1) *if agg_p is SUM or $COUNT$, $\exists m \in \mathbb{N}$ and $\exists n, \alpha \in \mathbb{N}^+$ such that $Size_p = Size_{u_i} + m \times Step_{u_i}$, $Size_{u_i} = n \times Step_{u_i}$ and $m = \alpha \times n - n$,*
or
2.2) *if agg_p is MIN or MAX, $\exists m \in \mathbb{N}$ such that $Size_p = Size_{u_i} + m \times Step_{u_i}$ and $Size_p - (m+1) \times Step_{u_i} \geq 0$.*

Proof. An answer of Q_p can be obtained from answers to a subset S containing only one temporal aggregated conjunctive utility query Q_{u_i}:
1) if according to Theorem 4.3.2, the aggregated conjunctive part of Q_{u_i} raises a (weak) privacy risk for the aggregated conjunctive part of Q_p,
2) if at least one time window of Q_p can be built from the union of some time windows of Q_{u_i}, that is only possible if the following conditions are satisfied:
(1) the union of m time windows of Q_{u_i} have the same size than a single time window of Q_p;
(2) a time window of Q_{u_i} and a time window of Q_p starts at the same time;
(3) the union must be disjoint for aggregates SUM and $COUNT$ (e.g. union of I_{u_1} and I_{u_4} in grey in Figure 4.1);
(4) the union can be overlapping (e.g. union of I_{u_1}, I_{u_2}, I_{u_3} and I_{u_4} in Figure 4.1) or disjoint (e.g. union of I_{u_1} and I_{u_4} in Figure 4.1) for aggregates MAX and MIN.
The given conditions are captured by the following equations:

$$\begin{cases} Size_p = Size_{u_i} + m \times Step_{u_i} & (1) \\ k_p \times Step_p = k_u \times Step_{u_i} & (2) \\ Size_{u_i} = n \times Step_{u_i} \text{ for } SUM \text{ and } COUNT & (3.1) \\ Size_p = \alpha \times Size_{u_i} \text{ for } SUM \text{ and } COUNT & (3.2) \\ Size_{u_i} \geq Step_{u_i} \text{ for } MAX \text{ and } MIN & (4) \end{cases}$$

where k_p, k_u and m are unknown integers, n and α are strictly positive unknown integers and $Size_p$, $Step_p$, $Size_{u_i}$ and $Step_{u_i}$ are constant integers.
Equation (2) clearly always has solutions (e.g. $k_p = k_u = 0$) and can be discarded. By combining equations (1) and (3.2) and equations (1) and (4), we obtain the following equations:

$$\begin{cases} Size_p = Size_{u_i} + m \times Step_{u_i} & (1) \\ Size_{u_i} = n \times Step_{u_i} \text{ for } SUM \text{ and } COUNT & (3.1) \\ (\alpha - 1) \times Size_{u_i} = m \times Step_{u_i} \text{ for } SUM \text{ and } COUNT & (3.2) \\ Size_p - m \times Step_{u_i} \geq Step_{u_i} \text{ for } MAX \text{ and } MIN & (4) \end{cases}$$

By combining equations (3.1) and (3.2) and subtracting $Step_{u_i}$ from both sides of equation (4), we obtain the following equations:

$$\begin{cases} Size_p = Size_{u_i} + m \times Step_{u_i} & (1) \\ Size_{u_i} = n \times Step_{u_i} \text{ for } SUM \text{ and } COUNT & (3.1) \\ m = \alpha \times n - n \text{ for } SUM \text{ and } COUNT & (3.2) \\ Size_p - (m+1) \times Step_{u_i} \geq 0 \text{ for } MAX \text{ and } MIN & (4) \end{cases}$$

From the above equations, we get the conditions 2.1 and 2.2 of theorem that raise a (weak) privacy risk for Q_p:
2.1) $\exists m \in \mathbb{N}$ and $\exists n, \alpha \in \mathbb{N}^+$ such that $Size_p = Size_{u_i} + m \times Step_{u_i}$, $Size_{u_i} = n \times Step_{u_i}$ and $m = \alpha \times n - n$ for aggregates SUM and COUNT.
2.2) $\exists m \in \mathbb{N}$ such that $Size_p = Size_{u_i} + m \times Step_{u_i}$ and $Size_p - (m+1) \times Step_{u_i} \geq 0$ for aggregates MAX and MIN. □

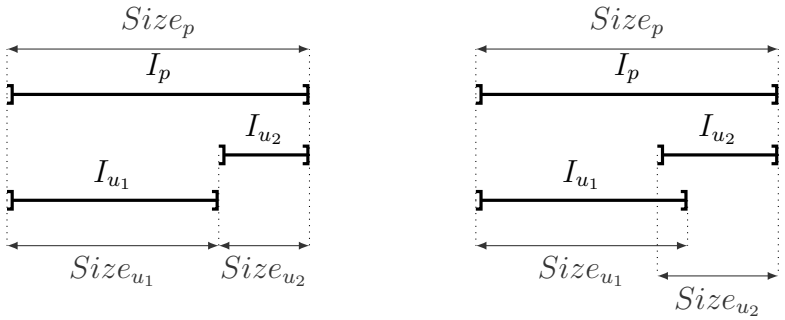
Example 17. *Let us consider the following privacy query PQ2 and the utility query UQ3 (up to variable renaming) of the scenario presented in Section 4.1.*
PQ2: SELECT ?timeWindowEnd SUM(?c)
WHERE (?sm :consumption ?c, ?ts)
GROUP BY ?timeWindowEnd

TIMEWINDOW (6h, 6h)
UQ3: SELECT ?sm1 ?timeWindowEnd SUM(?c1)
WHERE {(?sm1 :consumption ?c1, ?ts1)}
GROUP BY ?sm1 ?timeWindowEnd
TIMEWINDOW (3h, 1h)
In this case, the temporal graph patterns of PQ2 and UQ3 are isomorphic and UQ3 and PQ2 compute the same aggregate SUM, so condition 1 of the theorem is satisfied. Let us check the remaining conditions of Theorem 4.3.4 for the aggregate SUM.
Now we check if $\exists m \in \mathbb{N}$ *such that* $Size_p = Size_{u_i} + m \times Step_{u_i}$
As $Size_p = 6$, $Size_{u_i} = 3$ *and* $Step_{u_i} = 1$, *we get:*
$6 = 3 + m \times 1$
$m = 3$
Now we check if $\exists n \in \mathbb{N}^+$ *such that* $Size_{u_i} = n \times Step_{u_i}$*:*
$3 = n \times 1$
$n = 3$
Now we check if $\exists \alpha \in \mathbb{N}^+$ *such that* $m = \alpha \times n - n$*:*
$3 + 3 = \alpha \times 3$
$\alpha = 2$
The remaining conditions specified for the aggregate SUM in the theorem are also satisfied.
Thus, the privacy risk for PQ2 is raised by a single utility query UQ3.
This also implies that all the time windows of PQ2 can be computed from the disjoint union of two non successive time windows of UQ3. Thus, the aggregate SUM of Q_p *for all the time windows can be computed by taking sum of aggregates obtained from the non successive disjoint time windows of UQ3.*
Now let us replace the value of step in the time window definition of UQ3 by $2h$*.*
In this case, $m \notin \mathbb{N}$*, so no privacy risk is raised for PQ2 and it is also not possible to build any of the time windows of PQ2 from the disjoint union of time windows of a modified utility query.*

4.3.3.3 Privacy risk raised by two subsets that contain only one TACQ

Theorem 4.3.5 characterizes a (weak) privacy risk by evaluating a privacy query Q_p with two TACQs Q_{u_1} and Q_{u_2} that come from two different subsets S and S' such that each subset contains only one TACQ and it finds the possibility to build

at least one time window of Q_p from the union of some time windows coming from Q_{u_1} and Q_{u_2} that compute the same aggregate on different time windows. For example, Figure 4.2 illustrates the possibility to build a time window of Q_p from the union of two time windows coming from two different utility queries Q_{u_1} and Q_{u_2}. The aggregates SUM and $COUNT$ require two time windows I_{u_1} of Q_{u_1} and I_{u_2} of Q_{u_2} to be disjoint (as illustrated in Figure 4.2(a)) in order to prevent the double counting caused by overlapping, whereas for the aggregates MIN and MAX, a time window of Q_p can be built from a disjoint union of two time windows I_{u_1} of Q_{u_1} and I_{u_2} of Q_{u_2} (as illustrated in Figure 4.2(a)) or from an overlapping union of two time windows I_{u_1} of Q_{u_1} and I_{u_2} of Q_{u_2} (as illustrated in Figure 4.2(b)).



(a) For SUM or COUNT or MAX or MIN (b) For MAX or MIN

Figure 4.2: Union of time windows from two utility queries

Theorem 4.3.5 (Characterizing privacy risk for a temporal aggregated conjunctive privacy query raised by the two subsets that contain only one TACQ computing the same aggregate as the temporal aggregated conjunctive privacy query)**.** *Let Q_p be a temporal aggregated conjunctive privacy query. A subset S containing only one temporal aggregated conjunctive utility query Q_{u_1} and a subset S' containing only one temporal aggregated conjunctive utility query Q_{u_2} raise a (weak) privacy risk for Q_p* if and only if*:*
1) *the* aggregated conjunctive parts *of Q_{u_1} from S and Q_{u_2} from S' raise a (weak) privacy risk for the* aggregated conjunctive part *of Q_p,*
2.1) *agg_p is SUM or COUNT, $Size_p = Size_{u_1} + Size_{u_2}$ and $Size_p - Size_{u_1}$ is a multiple of* $\gcd(Step_{u_1}, \sigma_p \times Step_{u_2})$ *where* $\sigma_p = \frac{Step_p}{\gcd(Step_p, Step_{u_2})}$,
or
2.2) *agg_p is MAX or MIN, $Size_p \leq Size_{u_1} + Size_{u_2}$ and $Size_p - Size_{u_1}$ is a multiple of* $\gcd(Step_{u_1}, \sigma_p \times Step_{u_2})$ *where* $\sigma_p = \frac{Step_p}{\gcd(Step_p, Step_{u_2})}$.

Proof. An answer of Q_p can be obtained from answers to subsets S and S' that contain only one temporal aggregated conjunctive utility query:
1) if according to Theorem 4.3.2, the aggregated conjunctive parts of Q_{u_1} from S and Q_{u_2} from S' raise a (weak) privacy risk for the aggregated conjunctive part of Q_p,
2) if at least one time window I_p of Q_p can be built from the union of a time window I_{u_1} of Q_{u_1} and a time window I_{u_2} of Q_{u_2} as shown in Figure 4.2, that is only possible if the following conditions are satisfied (Q_{u_1} and Q_{u_2} can be inverted):
(1) the union of a time window of Q_{u_1} and a time window of Q_{u_2} have the same size as a single time window of Q_p;
(2) a time window of Q_{u_1} ends when a time window of Q_p ends;
(3) a time window of Q_{u_2} starts when the same time window of Q_p starts.
The given conditions are captured by the following equations:

$$\begin{cases} Size_p = Size_{u_1} + Size_{u_2} \text{ (a) or } Size_p \leq Size_{u_1} + Size_{u_2} \text{ (b)} & (1) \\ k_1 \times Step_{u_1} + Size_{u_1} = k_p \times Step_p + Size_p & (2) \\ k_p \times Step_p = k_2 \times Step_{u_2} & (3) \end{cases}$$

where k_1, k_2 and k_p are unknown integers and $Step_p$, $Size_p$, $Step_{u_1}$, $Size_{u_1}$, $Step_{u_2}$ and $Size_{u_2}$ are constant integers.
The positive integer solutions of equation (3) are of the form:

$$\begin{cases} k_p = \kappa \times \sigma_{u_2} & (3.1) \\ k_2 = \kappa \times \sigma_p & (3.2) \end{cases} \quad \text{with } \kappa \in \mathbb{N}$$

where $\sigma_p = \frac{Step_p}{\gcd(Step_p, Step_{u_2})}$ and $\sigma_{u_2} = \frac{Step_{u_2}}{\gcd(Step_p, Step_{u_2})}$.

Injecting solutions of equation (3.2) into equation (2), we obtain:

$$\begin{cases} Size_p = Size_{u_1} + Size_{u_2} \text{ (a) or } Size_p \leq Size_{u_1} + Size_{u_2} \text{ (b)} & (1) \\ k_1 \times Step_{u_1} - \kappa \times \sigma_p \times Step_{u_2} = Size_p - Size_{u_1} & (2) \\ k_p = \kappa \times \sigma_{u_2} & (3.1) \\ k_2 = \kappa \times \sigma_p & (3.2) \end{cases}$$

Equations (3.1) and (3.2) are always satisfied and can be discarded. According to Bachet-Bézout theorem, the Diophantine equation (2) has a solution if and only if $Size_p - Size_{u_1}$ is a multiple of $\gcd(Step_{u_1}, \sigma_p \times Step_{u_2})$. □

Remark 2. *If Q_{u_1} and Q_{u_2} raise a (weak) privacy risk for Q_p, the first occurrence of a time window of Q_p that can be built from the union of time windows of Q_{u_1} and Q_{u_2} is obtained from the smallest integer value of κ solution of Diophantine equation (2), noted κ_0, using equation (3.1): $k_{p_{min}} = \kappa_0 \times \frac{Step_{u_2}}{\gcd(Step_p, Step_{u_2})}$.*

Example 18. *Let us consider the privacy query PQ2 of the scenario in Section 4.1 and the following utility queries:*

PQ2: SELECT ?timeWindowEnd SUM(?c)
WHERE (?sm :consumption ?c, ?ts)
GROUP BY ?timeWindowEnd
TIMEWINDOW (6h, 6h)

Q_{u_1}: SELECT ?timeWindowEnd SUM(?c1)
WHERE {(?sm1 :consumption ?c1, ?ts1)}
GROUP BY ?timeWindowEnd
TIMEWINDOW (4h, 2h)

Q_{u_2}: SELECT ?timeWindowEnd SUM(?c2)
WHERE {(?sm2 :consumption ?c2, ?ts2)}
GROUP BY ?timeWindowEnd
TIMEWINDOW (2h, 1h)

In this case, the temporal graph patterns of PQ2 and Q_{u_1} as well as of PQ2 and Q_{u_2} are isomorphic. PQ2, Q_{u_1} and Q_{u_2} compute the same aggregate SUM, so condition 1 of the theorem is satisfied. Let us check the remaining conditions of Theorem 4.3.5 for the aggregate SUM.
As $Size_p = 6$ and $Size_{u_1} + Size_{u_2} = 6$.
Now we check if $Size_p - Size_{u_1}$ is a multiple of $\gcd(Step_{u_1}, \sigma_p \times Step_{u_2})$.
As $Step_p = 6$, $Step_{u_1} = 2$ and $Step_{u_2} = 1$, we get:
$\sigma_p = 6$, $\gcd(Step_{u_1}, \sigma_p \times Step_{u_2}) = 2$, and $Size_p - Size_{u_1} = 2$.
The remaining conditions specified for the aggregate SUM in the theorem are also satisfied.
Thus, the privacy risk for PQ2 is raised by Q_{u_1} and Q_{u_2}.
The first occurrence of a time window of PQ2 that can be built from the union of time windows of Q_{u_1} and Q_{u_2} can be obtained from the smallest κ solution of Diophantine equation (2). From Diophantine equation (2), we get the smallest integer solution for the pair ($k_1 = 1$, $\kappa = 0$). Substituting the value for κ in equation (3.1), we get $k_{p_{min}} = 0$. This implies the first time window of PQ2 can be built from the disjoint union of time windows of Q_{u_1} and Q_{u_2}.
Now let us replace the size of time window of Q_{u_1} by $3h$.

In this case, the $Size_p \neq Size_{u_1} + Size_{u_2}$*, so no privacy risk is raised for* $PQ2$ *and it is also not possible to build any of the time windows of* $PQ2$ *from the disjoint union of time windows of modified utility query* Q_{u_1} *and the utility query* Q_{u_2}*.*

4.4 Algorithms for detecting privacy risks

To make our approach effective, we have designed and implemented[1] several algorithms based on the theorems presented in Section 4.3. In Section 4.3, we presented the theorems to characterize a privacy risk by distinguishing the cases when a privacy query is a conjunctive query or an aggregated conjunctive query or a temporal aggregated conjunctive query. In order to provide a comprehensive assessment of privacy risk for any type of privacy query, we have designed and implemented a general Algorithm 1 that integrates three sub-algorithms, each corresponding to theorem(s) formalized within each case of a privacy query.

Each query provided as input to the algorithms consists of the following parts, with pre-defined default values:

- $\{\bar{x}\}$ is a tuple of variables (output or grouping variables) and is empty if not specified in the query;
- agg is an aggregate function and is null if not specified in the query;
- y is an aggregate variable and is null if agg is not specified in the query;
- $\{(?s\ p\ ?o,\ ?ts)\}$ is a finite set of temporal patterns and must be specified in the query. If ?ts is not specified in the query then its default value is null;
- $FILTER$ is a boolean expression and is true if not specified in the query;
- $(Size, Step)$ represents two time durations, where $Size$ is the time duration of each time window and $Step$ is the time duration separating consecutive time windows. If $Size$ and $Step$ are not specified in the query then their default values are $(\infty, 0)$.

Following the terminologies used in Section 4.3, the algorithms return any one of the three possible outcomes: privacy risk denoted as $\mathcal{P}^{risk}$, weak privacy risk denoted as $\mathcal{W}\_\mathcal{P}^{risk}$, or no privacy risk denoted as $\mathcal{N}\_\mathcal{P}^{risk}$.

[1]We used the python 3.9.6: `https://www.python.org/downloads/release/python-396/` and our code is available at the GitHub repository: `https://github.com/fr-anonymous/puck`

In our approach, we consider that queries have no variable names in common, so at the start of Algorithm 1, we replace each distinct variable name with a new one in each privacy query and utility query. The naming convention employed for renaming the variables in each query consists of three parts, which are denoted as $?Q\_T\_I$, where:

- Q represents the query name;
- T represents the variable type. If the variable name corresponds to an output variable then it will be o, if the variable name corresponds to an aggregate variable then it will be a, if the variable name corresponds to a timestamp variable then it will be t, otherwise it will be v;
- I represents an incrementing integer that increases by 1 each time a distinct variable name appears in the query.

In the algorithms, we will denote the variable name $?Q\_T\_I$ as Var and each of its parts will be denoted as Var.QueryName, Var.Type and Var.Int.

Example 19 illustrates how the algorithm employs the naming convention to generate an equivalent query with new variable names.

Example 19. *Let us consider the following privacy query PQ*1 *of Example 12:*
PQ_1*: SELECT ?sm ?y*
WHERE { *?sm :associatedOccupier ?o . ?o :yearlyIncome ?y .*
FILTER(?y $>$ 80000*)* }
After applying the naming convention to PQ_1*, we get:*
PQ_1*: SELECT* $?PQ_1\_o\_1$ $?PQ_1\_o\_2$
WHERE { $?PQ_1\_o\_1$ *:associatedOccupier* $?PQ_1\_v\_3$.
$?PQ_1\_v\_3$ *:yearlyIncome* $?PQ_1\_o\_2$. *FILTER(*$?PQ_1\_o\_2 > 80000$*)* }

Once the variable names in all privacy and utility queries are replaced with new ones, the Algorithm 1 proceeds to execute steps that involve evaluating different parts of a privacy query expression with the corresponding parts of the utility query expressions in a cascading manner, aiming to detect a (weak) privacy risk. The details of these steps are presented in Algorithms 2, 3 and 4 in the following sections.

Section 4.4.1 presents the algorithm that is designed and implemented to detect a (weak) privacy risk by evaluating the conjunctive part of a privacy query with the set of utility queries. Section 4.4.2 details the algorithm that is designed and

Algorithm 1: Detecting privacy risks

Input: a set $\mathcal{P}$ of m privacy queries, PQ_i= ($\{\bar{x}_{p_i}\}$, agg_{p_i}, y_{p_i}, $\{(s\ p\ o, ?ts)\}_{p_i}$, $FILTER_{p_i}$, $Size_{p_i}$, $Step_{p_i}$)
a set $\mathcal{U}$ of n utility queries, UQ_i= ($\{\bar{x}_{u_i}\}$, agg_{u_i}, y_{u_i}, $\{(s\ p\ o, ?ts)\}_{u_i}$, $FILTER_{u_i}$, $Size_{u_i}$, $Step_{u_i}$)

Output: returns privacy risk ($\mathcal{P}^{risk}$) or weak privacy risk ($\mathcal{W}\_\mathcal{P}^{risk}$) or no privacy risk ($\mathcal{N}\_\mathcal{P}^{risk}$)

```
1  Risks_main(P, U)
2  forall PQ_i ∈ P do
3      rename_variables(PQ_i)                          // renaming variables in privacy query
4  forall UQ_i ∈ U do
5      rename_variables(UQ_i)                          // renaming variables in utility query
6  Risks:={ }                                  // sets that are utilized by subsequent functions
7  forall PQ_i ∈ P do
8      Step_1 := test_conjunctive_part(PQ_i,U)
9      if Step_1 ≠ N_P^risk and agg_{p_i} ≠ ∅ then
10         Step_2 := test_aggregtaed_conjunctive_part(PQ_i,U, Step_1)
11         if Step_2 ≠ N_P^risk and Size_{p_i} ≠ ∞ then
12             Step_3 := test_time_window_definitions(PQ_i,U, Step_2)
13             return Step_3
14         else
15             return Step_2
16     else
17         return Step_1
```

implemented to detect a (weak) privacy risk by evaluating aggregated conjunctive part of a privacy query with the subset of utility queries. Section 4 details the algorithm that is designed and implemented to detect a (weak) privacy risk by evaluating the time window definitions of privacy and utility queries.

4.4.1 Testing conjunctive part

Algorithm 2 detects a (weak) privacy risk for a conjunctive part of a privacy query (or conjunctive privacy query) by evaluating it against a set of utility queries. According to the Theorem 4.3.1, there is a (weak) privacy risk if and only if:
1) a conjunctive privacy query returns an answer when evaluated over the freezing of the variables in the union of temporal graph patterns of the utility queries,
2) the conjunction of the FILTER conditions of privacy and utility queries is satisfiable.

To test the first condition of the theorem, at the start of Algorithm 2, we obtain

the general freezing of the variables in the union of temporal graph patterns of all utility queries (lines 7 to 9) over which the conjunctive privacy query can be evaluated. The freezing of the variables in the temporal graph pattern of each utility query is computed by generating constants for all the distinct variable names in the utility query. The constant is named in a manner that reflects the variable name and consists of three parts, which are denoted as Q_C_I, where:

- Q represents the query name;
- T represents the constant type. If it represents an output constant then it will be o, if it is an aggregate constant then it will be a, if it is a timestamp constant then it will be t, otherwise it will be v;
- I represents an incrementing integer that increases by 1 each time a constant is generated for a distinct variable in the query.

In the algorithms, we will denote the constant Q_T_I as Con and each of its parts will be denoted as Con.QueryName, Con.Type and Con.Int.

Example 20 illustrates how the algorithm computes the temporal graph U_TG by replacing the variables with constants in the union of temporal graph patterns in a set of utility queries.

Example 20. *Let us consider the utility queries UQ_1, UQ_2 and UQ_3 corresponding to the utility queries (up to variable renaming) of the scenario illustrated in Section 4.1 and following utility query UQ_4:*

UQ_1: SELECT ?UQ_1_o_1 ?UQ_1_o_2 ?UQ_1_o_3
WHERE { ?UQ_1_o_1 :associatedOccupier ?UQ_1_o_2
. ?UQ_1_o_2 :numberOfPersons ?UQ_1_o_3 }

UQ_2: SELECT ?UQ_2_o_1 ?UQ_2_o_2
WHERE { ?UQ_2_o_1 :yearlyIncome ?UQ_2_o_2 . ?UQ_2_o_1 :owns ?UQ_2_v_3
. FILTER (?UQ_2_o_2 > 75000) }

UQ_3: SELECT ?UQ_3_o_1 ?UQ_3_o_2 SUM(?UQ_3_a_3)
WHERE {(?UQ_3_o_1 :consumption ?UQ_3_a_3, ?UQ_3_t_4)}
GROUP BY ?UQ_3_o_1 ?UQ_3_o_2
TIMEWINDOW (3h, 1h)

UQ_4: SELECT ?UQ_4_o_1 ?UQ_4_o_2 ?UQ_4_o_3 MAX(?UQ_4_a_4)
WHERE { ?UQ_4_o_1 :associatedBuilding ?UQ_4_o_2 .
?UQ_4_o_2 rdf:type ?UQ_4_o_3 .
(?UQ_4_o_1 :consumption ?UQ_4_a_4, ?UQ_4_t_5) .

$FILTER\ (?UQ_4\_t\_5 > 2023-04-01T00:00:00)\ \}$
$GROUP\ BY\ ?UQ_4\_o\_1\ ?UQ_4\_o\_2\ ?UQ_4\_o\_3$

The general freezing of variables with the constants in the temporal graph patterns of the utility queries is as follows:
freeze$(\{(?s\ p\ .o,\ ?ts)\}_{u_1}) = UQ_1\_o\_1$ *:associatedOccupier* $UQ_1\_o\_2$. $UQ_1\_o\_2$ *:numberOfPersons* $UQ_1\_o\_3$
freeze$(\{(?s\ p\ ?o,\ ?ts)\}_{u_2}) = UQ_2\_o\_1$ *:yearlyIncome* $UQ_2\_o\_2$. $UQ_2\_o\_1$ *:owns* $UQ_2\_v\_3$
freeze$(\{(?s\ p\ ?o,\ ?ts)\}_{u_3}) = (UQ_3\_o\_1$ *:consumption* $UQ_3\_a\_3,\ UQ_3\_t\_4)$
freeze$(\{(?s\ p\ ?o,\ ?ts)\}_{u_4})$*:* $UQ_4\_o\_1$ *:associatedBuilding* $UQ_4\_o\_2$. $UQ_4\_o\_2$ *rdf:type* $UQ_4\_o\_3$. $(UQ_4\_o\_1$ *:consumption* $UQ_4\_a\_4,\ UQ_4\_t\_5)$
The union of the temporal graph U_TG obtained by freezing variables with constants is as follows:
$UQ_1\_o\_1$ *:associatedOccupier* $UQ_1\_o\_2$. $UQ_1\_o\_2$ *:numberOfPersons* $UQ_1\_o\_3$. $UQ_2\_o\_1$ *:yearlyIncome* $UQ_2\_o\_2$. $UQ_2\_o\_1$ *:owns* $UQ_2\_v\_3$. $(UQ_3\_o\_1$ *:consumption* $UQ_3\_a\_3,\ UQ_3\_t\_4)$. $UQ_4\_v\_1$ *:associatedBuilding* $UQ_4\_v\_2$. $UQ_4\_v\_2$ *rdf:type* $UQ_4\_o\_3$. $(UQ_4\_v\_1$ *:consumption* $UQ_4\_a\_4,\ UQ_4\_t\_5)$

Obtaining answers for the conjunctive privacy query when evaluated over U_TG is only possible if:
1) the output variables of conjunctive privacy query correspond to output constants in U_TG,
2) the join conditions in the temporal graph pattern TGP_p of a privacy query are satisfied, which means the join variables involved in join conditions correspond to output constants in U_TG,
3) the conjunction of the FILTER conditions of privacy and utility queries is satisfiable.

First, the conjunctive part of the given privacy query is obtained (line 6). To obtain the join conditions, TGP_p is evaluated and the join conditions in TGP_p are extracted and stored in the form of ordered pairs, denoted as (V_1, V_2), where V_1 represents the first occurrence of a join variable in TGP_p and V_2 represents the recurrence of a variable in the temporal graph pattern. A new variable name denoted as N_V_2 is generated for each variable name V_2. A new variable name is also generated using the same naming convention already presented in Section 4.4. In this case, while generating a new variable name, the first integer value for I is computed by incrementing the last integer value assigned to a variable while renaming it. A new temporal graph pattern TGP' is computed by replacing

each recurring join variable V_2 with N_V_2. Example 21 and lines 13 to 21 in the algorithm illustrate the complete process of obtaining join conditions and TGP'.

Example 21. *Temporal graph pattern of PQ_1 of Example 19:*
{ $?PQ_1\_o\_1$ *:associatedOccupier* $?PQ_1\_v\_3$. $?PQ_1\_v\_3$ *:yearlyIncome* $PQ_1\_o\_2$ }
Join conditions in temporal graph pattern of PQ_1:
Joins=($?PQ_1\_v\_3$, $?PQ_1\_v\_3$)
New variable name generated for $?PQ_1\_v\_3$ is $?PQ_1\_v\_4$.
Replacing the recurring $?v\_PQ_1\_3$ in temporal graph pattern of PQ_1:
TGP' = { $?PQ_1\_o\_1$ *:associatedOccupier* $?PQ_1\_v\_3$. $?PQ_1\_v\_4$ *:yearlyIncome* $PQ_1\_o\_2$ }
A pair of join conditions after replacing recurring $?PQ_1\_v\_3$ with $?PQ_1\_v\_4$:
N_Joins=($?PQ_1\_v\_3$, $?PQ_1\_v\_4$)

To obtain the mapping for all the output, aggregate and join variables of a privacy query, a modified version of a privacy query PQ' (without the FILTER condition) is computed such that it contains all the variables in TGP' as output variables. Example 22 and lines 22 to 24 in the algorithm illustrate these steps.

Example 22. *PQ' obtained for PQ_1 of Example 19 by considering all variables included in the temporal graph pattern of $PQ1$ as output variables:*
PQ': SELECT $?PQ_1\_o\_1$ $?PQ_1\_v\_3$ $?PQ_1\_v\_4$ $?PQ_1\_o\_2$
WHERE { $?PQ_1\_o\_1$ *:associatedOccupier* $?PQ_1\_v\_3$.
$?PQ_1\_v\_4$ *:yearlyIncome* $?PQ_1\_o\_2$ }

Then PQ' is evaluated over U_TG using SPARQL engine[2] (line 25). The results of the evaluation are obtained in tabular form, with each line (or row) in the table containing the mappings of constants obtained for all variables of PQ'. Each line L of the result is evaluated one after the other. First, it evaluates if all the output variables and/or aggregate variable in PQ' correspond to the output constants in L (lines 29 to 32) and then the algorithm tests the join conditions by evaluating if all the join variables also correspond to output constants in L (lines 33 to 35). If any of the join variables in a pair does not correspond to the output constant, then there is no privacy risk, but if both the join variables in each pair correspond to the output constants, then this means that by freezing each pair of join variables with the same constant in U_TG will return an answer to the privacy query. However, this is only possible if the conjunction of the FILTER conditions of the privacy and utility queries is satisfiable. If $FILTER_p$ is true and the conjunction of FILTER

[2]We used the python RDFLib library: `https://pypi.org/project/rdflib/`

conditions of utility queries is satisfiable then a privacy risk is raised for the privacy query. If the FILTER condition in privacy query is specified, then the conjunction of the FILTER conditions of privacy and utility queries is computed to test the condition 2 of the Theorem 4.3.1. For each line of the result, the FILTER condition is computed by replacing the variables corresponding to the privacy query with the variables of the utility queries by using the same constant assigned to the variables in result and U_TG. The satisfiability between the conjunction of the FILTER conditions of the queries is evaluated using the Constraint Satisfaction Problem (CSP) Solver [3]. If the conjunction of the FILTER conditions of privacy and utility queries is satisfiable then the line involved in a privacy risk is stored to test the conditions of other theorems and a weak privacy risk is detected by the algorithm. These steps are illustrated in lines 37 to 44 of the algorithm and in Example 23.

Example 23. *The result of evaluating PQ' of previous Example 22 over U_TG:*

Line #	$?PQ_1\_o\_1$	$?PQ_1\_v\_3$	$?PQ_1\_v\_4$	$?PQ_1\_o\_2$
1	$UQ_1\_o\_1$	$UQ_1\_o\_2$	$UQ_2\_o\_1$	$UQ_2\_o\_2$

Output variables $?PQ_1\_o\_1$ and $?PQ_1\_o\_2$ correspond to the output constants $UQ_1\_o\_2$ and $UQ_2\_o\_2$ in the first line of the result.
The pair of join variables ($?PQ_1\_v\_3$, $?PQ_1\_v\_4$) when equated with the constants obtained from the first line of the result, we get: $UQ_1\_o\_2 = UQ_2\_o\_1$. As both constants correspond to output constants, it implies that it is possible to obtain an answer to privacy query $PQ1$ of Example 12 by freezing the $UQ_1\_o\_2$ and $UQ_2\_o\_1$ with the same constant only if the conjunction of the FILTER conditions of privacy and utility queries is satisfiable.
Conjunction of the FILTER conditions of PQ_1 and UQ_2:
$?PQ_1\_o\_4 > 80000$ && $?UQ_2\_o\_2 > 75000$
Conjunction of the FILTER conditions obtained by replacing the variable of PQ_1 with the corresponding variable of UQ_2:
$?UQ_2\_o\_2 > 80000$ && $?UQ_2\_o\_2 > 75000$
The conjunction of the FILTER conditions of PQ_1 and UQ_2 is satisfiable so the algorithm returns a weak privacy risk.

[3] We used the python CSP library: `https://pypi.org/project/CSP-Solver/`

Algorithm 2: Testing conjunctive part

Input: a privacy query with renamed variables, PQ= ($\{\bar{x}_p\}$, agg_p, y_p, $\{(?s\ p\ ?o,\ ?ts)\}_p$, $FILTER_p$, $Size_p$, $Step_p$)
a set $\mathcal{U}$ of n utility queries with renamed variables, UQ_i= ($\{\bar{x}_{u_i}\}$, agg_{u_i}, y_{u_i}, $\{(?s\ p\ ?o,\ ?ts)\}_{u_i}$, $FILTER_{u_i}$, $Size_{u_i}$, $Step_{u_i}$)

Output: returns privacy risk ($\mathcal{P}^{risk}$) or weak privacy risk ($\mathcal{W}\_\mathcal{P}^{risk}$) or no privacy risk ($\mathcal{N}\_\mathcal{P}^{risk}$)

```
1  test_conjunctive_part(PQ,U)
2  TGP':= { }                  // a set of modified {(?s p ?o, ?ts)}_p with replaced join conditions
3  N_Joins:= { }                                                          // a set of joining tuples
4  U_TG:= { }                                          // a set of union of temporal graphs of all UQ_i
5  C_Filter:=true                                          // a set of conjunction of all FILTER_{u_i}
6  V_p:= { }                                                          // a tuple of variables in TGP'
7  foreach UQi ∈ U do
8      freeze({(?s p ?o, ?ts)}_{u_i})                    // replace variables with generated constants
9      U_TG:= U_TG ∪ freeze({(?s p ?o, ?ts)}_{u_i})
10     C_Filter:=C_Filter ∧ FILTER_{u_i}
11 Conj(PQ):=({x̄_p} ∪ {y_p}, {(?s p ?o, ?ts)}_p, FILTER_p)                  // conjunctive part of PQ
12 Joins:= extract_join_conditions({(?s p ?o, ?ts)}_p)
13 foreach (V1, V2) ∈ Joins do
14     generate N_V2 for V2                                          // N_V2 is a new variable name
15     foreach (?s p ?o, ?ts) in Conj(PQ) do
16         if s = V2 or o = V2 then
17             replace V2 with N_V2 to compute (?s p ?o, ?ts)'
18             TGP':= TGP' ∪ (?s p ?o, ?ts)'
19             N_Joins:= N_Joins ∪(V1, N_V2)
20         else
21             TGP' := TGP' ∪ (?s p ?o, ?ts)
22 foreach Var ∈ TGP' do
23     V_p= V_p ∪ TGP'
24 PQ':= "SELECT V_p WHERE TGP'"
25 Result:=evaluate(PQ', U_TG)
26 Risks:=Risks ∪ U_TG ∪ V_p
27 PQ_risk:=N_P^risk
28 Risks_lines:=P^risk
29 foreach line in Result do
30     foreach Var ∈ V_p do
31         foreach Con in line do
32             if Var.Type = o or Var.Type = a does not correspond to Con.Type = o or
                 Con.Type = a then Risks_lines:=N_P^risk;
33     if Risks_lines=P^risk and N_Joins ≠ ∅ then
34         foreach (V1, N_V2) ∈ N_Joins do
35             if V1 or N_V2 does not correspond to Con.Type = o in line then
                   Risks_lines:=N_P^risk;
36     if Risks_lines=P^risk then
37         Fil_Conj:=FILTER_p ∧ C_Filter
38         Fil_Expression:=rewrite(Fil_Conj)   // replace Var of FILTER_p with Var of C_Filter
39         Conjunction_FILTER:=evaluate(Fil_Expression)
40         if Conjunction_FILTER is satisfiable then
41             if Filter_p=true then PQ_risk:=P^risk;
42             if PQ_risk ≠ P^risk then PQ_risk:=W_P^risk;
43     if Risks_lines=P^risk then Risks:=Risks∪line;
44 return PQ_risk
```

4.4.2 Testing aggregated conjunctive part

Algorithm 3 is based on Theorem 4.3.2 and only focuses on evaluating the conditions 2 and 3 of Theorem 4.3.2, as the first condition is evaluated by applying Algorithm 2. According to the Theorem 4.3.2, there exists a (weak) privacy risk for Q_p *if and only if* there exists a subset S of utility queries such that:
1) the set of conjunctive parts of all utility queries in S raises a (weak) privacy risk for the conjunctive part of Q_p,
and
2) the union of the temporal graph patterns of the utility queries in S is isomorphic to TGP_p (through an isomorphism I), or can be made isomorphic to TGP_p (through an isomorphism I) by replacing some output variables by constants and/or by equating some output variables of some utility queries in S,
and either
3.1) $I(\bar{x}_p \cup \{y_p\}) \subseteq \bigcup_{Q_{u_i} \in S} \bar{x}_{u_i}$,
or
3.2) the subset S contains at least one aggregated conjunctive query Q_{u_i} where $I(\bar{x}_p) \subseteq \bar{x}_{u_i}$ and $agg_p = agg_{u_i}$ and $I(y_p) = y_{u_i}$.

To test the isomorphism (condition 2 of the Theorem 4.3.2) between the union of the temporal graph patterns of the utility queries in S and TGP_p, first the algorithm computes the subset S by evaluating each line L of the result that raised a (weak) privacy risk for the conjunctive part of a privacy query. Each line L contains the constants, where the first part of constant refers to the utility query name involved in raising privacy risk for the conjunctive part of a privacy query. The first part from each constant in L is extracted to obtain the subset S as illustrated in lines 7 to 14 of the algorithm. After obtaining the utility query names, the algorithm computes a set TG_c by extracting all the constants starting with the same query names from U_TG (lines 15 to 18). The isomorphism between the union of temporal patterns of utility queries in S and TGP_p is tested by evaluating all the constants in L and TG_c. If all of the constants in TG_c correspond to the constants in L then this implies that all the temporal patterns in TGP_p correspond to the union of temporal patterns of the utility queries in S, thus TGP_p is isomorphic to the union of temporal patterns of the utility queries in S (lines 19 to 22).

To test condition 3.1 of the Theorem 4.3.2, it is evaluated if the output and aggregate variables of a privacy query PQ' correspond to the output constant in L

and if this condition is satisfied then, it returns a (weak) privacy risk for a privacy query as illustrated in lines 24 to 31 in the algorithm otherwise, it tests condition 3.2 of the Theorem 4.3.2. To test condition 3.2 of the Theorem 4.3.2, the algorithm evaluates if any of the utility queries in S has the same aggregate function as the privacy query. If this condition is satisfied, then it tests if the aggregate variable of the privacy query PQ' correspond to the aggregate constant starting with the same utility query name in L. If this condition is satisfied then the algorithm checks if all the output variables of privacy query PQ' in L corresponds to the output constants of output variables of UQ_i that are obtained from U_TG and if this condition is also satisfied, then the algorithm returns a (weak) privacy risk for privacy query. These steps are covered in the algorithm from lines 32 to 38.

Example 24 illustrates the steps of Algorithm 2 by detecting a weak privacy risk for an aggregated conjunctive privacy query.

Example 24. *Let us consider the following privacy query PQ_1 and the same set of utility queries UQ_1, UQ_2, UQ_3 and UQ_4 of Example 20.*
PQ_1: SELECT $?PQ_1\_o\_3$ MAX($?PQ_1\_a\_4$)
WHERE { $?PQ_1\_v\_1$ *:associatedBuilding* $?PQ_1\_v\_2$.
$?PQ_1\_v\_2$ *rdf:type* $?PQ_1\_o\_3$.
($?PQ_1\_v\_1$ *:consumption* $?PQ_1\_a\_4$, $?PQ_1\_t\_5$) .
FILTER ($?PQ_1\_t\_5 > 2023-03-01T00:00:00$) }
GROUP BY $?PQ_1\_o\_3$
Rewritten PQ_1 with all the variables as output variables:
PQ': SELECT $?PQ_1\_v\_1$ $?PQ_1\_v\_2$ $?PQ_1\_v\_6$ $?PQ_1\_o\_3$ $?PQ_1\_v\_7$ $?PQ_1\_a\_4$ $?PQ_1\_t\_5$
WHERE { $?PQ_1\_v\_1$ *:associatedBuilding* $?PQ_1\_v\_2$.
$?PQ_1\_v\_6$ *rdf:type* $?PQ_1\_o\_3$.
($?PQ_1\_v\_7$ *:consumption* $?PQ_1\_a\_4$, $?PQ_1\_t\_5$) }
The result of evaluating PQ' over U_TG which was obtained in Example 20:

Line #	$?PQ_1\_v\_1$	$?PQ_1\_v\_2$	$?PQ_1\_v\_6$	$?PQ_1\_o\_3$	$?PQ_1\_v\_7$	$?PQ_1\_a\_4$	$?PQ_1\_t\_5$
1	$UQ_4\_o\_1$	$UQ_4\_o\_2$	$UQ_4\_o\_2$	$UQ_4\_o\_3$	$UQ_4\_o\_1$	$UQ_4\_a\_4$	$UQ_4\_t\_5$

By applying the Algorithm 2, it returns a weak privacy risk for the conjunctive part of a privacy query.
Extracting the query names from the first line of the result, we get the subset that only consists of one utility query UQ_4.
Extracting constants that start with the utility query name UQ_4 from U_TG, we

Algorithm 3: Testing aggregated conjunctive part

Input: a privacy query with renamed variables, PQ= ($\{\bar{x}_p\}$, agg_p, y_p, $\{(s\ p\ o, ?ts)\}_p$, $FILTER_p$, $Size_p$, $Step_p$)
a set $\mathcal{U}$ of n utility queries with renamed variables, UQ_i= ($\{\bar{x}_{u_i}$, agg_{u_i}, y_{u_i}, $\{(s\ p\ o, ?ts)\}_{u_i}$, $FILTER_{u_i}$, $Size_{u_i}$, $Step_{u_i}$)
Step_1

Output: returns privacy risk ($\mathcal{P}^{risk}$) or weak privacy risk ($\mathcal{W}\_\mathcal{P}^{risk}$) or no privacy risk ($\mathcal{N}\_\mathcal{P}^{risk}$)

```
1  test_aggregated_conjunctive_part(PQ,U,Step_1)
2  UQ_names:={ }                          // a set of extracted query names from constants
3  S:={ }                  // a subset of U raising privacy risk for conjunctive part of PQ
4  TG_c:={ }              // a set of constants extracted from union of frozen (s p o, ?ts)}_{u_i}
5  S_line:={ }            // lines raising privacy risk for aggregated conjunctive part of PQ
6  if PQ_risk≠ N_P^risk and agg_p ≠ ∅ then
7      foreach line in Risks do
8          foreach Con in line do
9              if Con.QueryName ∉ UQ_names then
10                 UQ_names:=UQ_names ∪ Con.QueryName
11         foreach UQ_i ∈ U do
12             foreach Con.QueryName ∈ UQ_names do
13                 if UQ_i=Con.QueryName then
14                     S:= S ∪ UQ_i          // computing a subset S of utility queries
15         foreach UQ_i ∈ S do
16             foreach Con in U_TG do
17                 if Con.QueryName = UQ_i then
18                     TG_c=TG_c ∪ Con
19         foreach Con ∈ TG_c do
20             foreach Con in line do
                                                                  // Checking isomorphism
21                 if Con not in line then
22                     PQ_risk:=N_P^risk
23         if PQ_risk ≠ N_P^risk then
24             foreach Var ∈ V_p do
25                 foreach Con in line do
26                     if Var.Type=a does not correspond to Con.Type=o
27                     or
28                     Var.Type=o does not correspond to Con.Type=o then
29                         PQ_risk:=N_P^risk
30                     else
31                         PQ_risk≠ N_P^risk, S_line:=S_line ∪ line, Risks:= Risks ∪ S_line
32                     if PQ_risk:=N_P^risk then
33                         foreach UQ_i ∈ S do
34                             if agg_p = agg_{u_i} then
35                                 if Var.Type=a does not correspond to
                                     Con.QueryName=UQ_i and Con.Type=a
36                                 or
37                                 Var.Type=o does not correspond to Con.QueryName=UQ_i
                                    and Con.Type=o then
38                                     PQ_risk:=N_P^risk
39         if PQ_risk ≠ N_P^risk then Risks:= Risks ∪S;
40 return PQ_risk
```

get:
TG_c: {UQ_4_o_1, UQ_4_o_2, UQ_4_o_2, UQ_4_o_3, UQ_4_o_1, UQ_4_a_4, UQ_4_t_5}
Comparing the constants in TG_c with the constants in the first line of the result shows that the temporal graph patterns of PQ_1 and UQ_4 are isomorphic.
The aggregate function MAX is the same in both PQ_1 and UQ_4, the aggregate variables ?PQ_1_a_4 of PQ_1 and ?UQ_4_a_4 of UQ4 correspond to the aggregate constant UQ_4_a_4. The output variable ?PQ_1_o_3 of PQ_1 and the output variable ?UQ_4_o_3 of UQ4 correspond to the output constant UQ_4_o_3. It is possible to compute an aggregate result of PQ_2 from the answers of UQ_4, thus the Algorithm 3 will return a weak privacy risk for PQ_1.

4.4.3 Testing time window definitions

Algorithm 4 outlines the conditions of three Theorems 4.3.3, 4.3.4 and 4.3.5 that characterize a (weak) privacy risk for a temporal aggregated conjunctive privacy query by evaluating it against one (or two) subset(s) of utility queries. The following condition 1 provided in all the theorems is tested by applying Algorithm 2: *1)* if the aggregated conjunctive parts of utility queries in one subset (in Theorems 4.3.3 and 4.3.4) or two subsets (in Theorem 4.3.5) raise a (weak) privacy risk for the aggregated conjunctive part of a privacy query.

Algorithm 4 detects a privacy risk by evaluating if there is a possibility to build at least one time window of a privacy query from the answers of utility queries in a subset S or from the disjoint union (when aggregate function is SUM or COUNT) or from the union (when aggregate function is MAX or MIN) of time windows of one or two utility queries coming from one or two different subsets.

In the start of Algorithm 4, it evaluates the case when the subset S does not contain any of the utility queries that compute the same aggregate as the privacy query, it tests if the timestamp variables of all dynamic properties in the temporal graph patterns of utility queries in S are included in the union of the output variables of utility queries in S. To test this condition, it evaluates the line of the result involved in a privacy risk in Algorithm 3. If all the timestamp variables (ie.; Var.Type=t) of privacy query PQ' correspond to the output constant (i.e.; Con.Type=o) in line (lines 5 to 9) then this implies it is possible to build at least one time window of a privacy query from the answers of utility queries in subset S, thus the algorithm returns a (weak) privacy risk.

If the privacy query and a utility query compute the same aggregate as the privacy query but on different time window definitions, then the Algorithm 4 determines the subsequent steps on the basis of the aggregate of the privacy and utility queries and tests the possibility of building at least one time window of the privacy query from the union of some time windows of a single utility query by evaluating the following conditions 2.1 and 2.2 of the Theorem 4.3.4 by substituting the values of $Size_p$, $Size_{u_i}$ and $Step_{u_i}$ in the equations specified below:
2.1) if agg_p is SUM or $COUNT$, $\exists m \notin \mathbb{N}$, $\exists n, \alpha \in \mathbb{N}^+$ such that $Size_p = Size_{u_i} + m \times Step_{u_i}$, $Size_{u_i} = n \times Step_{u_i}$ and $m = \alpha \times n - n$,
or
2.2) if agg_p is MAX or MIN, $\exists m \in \mathbb{N}$ such that $Size_p = Size_{u_i} + m \times Step_{u_i}$ and $Size_p - (m+1) \times Step_{u_i} \geq 0$.

Lines 16 to 19 in the algorithm test condition 2.1 of the Theorem 4.3.4 and if m, n and α have integer solutions that satisfy the given condition, then this implies it is possible to build at least one time window of the privacy query from the disjoint union of some time windows of a single utility query. Lines 20 to 23 in the algorithm test condition 2.2 of the Theorem 4.3.4 and if m have integer solutions that satisfy the given condition, then this implies it is possible to build at least one time window of a privacy query from the disjoint or overlapping union of some time windows of a single utility query. If the both conditions 2.1 or 2.2 are not satisfied then the utility query is stored and evaluated in a pair with another utility query to test the following conditions 2.1 and 2.2 of the Theorem 4.3.5:
2.1) agg_p is SUM or COUNT, $Size_p = Size_{u_1} + Size_{u_2}$ and $Size_p - Size_{u_1}$ is a multiple of $\gcd(Step_{u_1}, \sigma_p \times Step_{u_2})$ where $\sigma_p = \frac{Step_p}{\gcd(Step_p, Step_{u_2})}$,
or
2.2) agg_p is MAX or MIN, $Size_p \leq Size_{u_1} + Size_{u_2}$ and $Size_p - Size_{u_1}$ is a multiple of $\gcd(Step_{u_1}, \sigma_p \times Step_{u_2})$ where $\sigma_p = \frac{Step_p}{\gcd(Step_p, Step_{u_2})}$.

Lines 28 to 30 in the algorithm test condition 2.1 of Theorem 4.3.5 and if the condition is satisfied then, this implies it is possible to build at least one time window of the privacy query from the disjoint union of two time windows of two different utility queries coming from different subsets. Lines 31 to 33 test condition 2.2 of the Theorem 4.3.5 and if the condition is satisfied, then this implies it is possible to build at least one time window of the privacy query from the disjoint or overlapping union of some time windows of two different utility queries coming from

different subsets. If any of the given conditions are satisfied, then the algorithm returns a (weak) privacy risk for a privacy query.

If the privacy and utility queries compute the same aggregate as the privacy query but on the same time window definitions, then evaluating condition 1 of the Theorem 4.3.4 is sufficient to prove that there exists a privacy risk and the algorithm returns a (weak) privacy risk (lines 13 to 14).

Example 24 illustrates the steps of Algorithm 3 by detecting a weak privacy risk for a temporal aggregated conjunctive privacy query.

Example 25. *Let us consider the privacy query PQ_2 of the scenario illustrated in Section 4.1 and the same set of utility queries UQ_1, UQ_2, UQ_3 and UQ_4 of Example 23.*
PQ_2: SELECT $?PQ_2\_o\_1$ SUM($?PQ_2\_a\_3$)
WHERE { ($?PQ_2\_v\_2$:consumption $?PQ_2\_a\_3$, $?PQ_2\_t\_4$) }
GROUP BY $?PQ_2\_o\_1$
TIMEWINDOW (6h, 6h)
Rewritten PQ_2 with all the variables as output variables:
PQ': SELECT $?PQ_2\_v\_2$ $?PQ_2\_a\_3$ $?PQ_2\_t\_4$
WHERE { ($?PQ_2\_v\_2$:consumption $?PQ_2\_a\_3$, $?PQ_2\_t\_4$) }
The result of evaluating PQ' over U_TG which was obtained in Example 23:

Line #	$?PQ_2\_v\_2$	$?PQ_2\_a\_3$	$?PQ_2\_t\_4$
1	$UQ_3\_o\_1$	$UQ_3\_a\_3$	$UQ_3\_t\_4$

By applying the Algorithm 3, it returns a privacy risk for the aggregated conjunctive part of a privacy query. PQ_2 and UQ_3 compute the same aggregate SUM on different time window definitions. The algorithm tests the conditions of Theorem 4.3.4 and computes the values for m, n and α by substituting the values of $Size_p$, $Size_{u_i}$ and $Step_{u_i}$ as follows:
$m = ((6-3)/1)$, *m= 3;*
$n = (3/1)$, *n= 3;*
$\alpha = ((3+3)/3)$, *α=2.*
As m, n and α have integer solutions, this implies it is possible to build a time window of PQ_2 from the disjoint union of some time windows of UQ_3, so the algorithm will return a privacy risk for PQ_2.

Algorithm 4: Testing time window definitions

Input: a privacy query with renamed variables, PQ= ($\{\bar{x}_p\}$, agg_p, y_p, $\{(?s\ p\ ?o,\ ?ts)\}_p$, $FILTER_p$, $Size_p$, $Step_p$)
a set $\mathcal{U}$ of n utility queries with renamed variables, UQ_i= ($\{\bar{x}_{u_i}\}$, agg_{u_i}, y_{u_i}, $\{(s\ p\ o,\ ?ts)\}_{u_i}$, $FILTER_{u_i}$, $Size_{u_i}$, $Step_{u_i}$)
Step_2

Output: returns privacy risk ($\mathcal{P}^{risk}$) or weak privacy risk ($\mathcal{W}\_\mathcal{P}^{risk}$) or no privacy risk ($\mathcal{N}\_\mathcal{P}^{risk}$)

```
1  test_time_window_definitions(PQ,U,Step_2)
2  Subsets={ }            // subsets that does not raise a privacy risk when checked in isolation
3  Agg_PQ_risk:= PQ_risk                                   // output of Algorithm 3
4  if PQ_risk≠ N_P^risk and Size_p ≠ ∞ then
5      foreach S_line ∈ Risks do
6          foreach Var ∈ V_p do
7              foreach Con in S_line do
8                  if Var.Type =t does not correspond to Con.Type=o then
9                      PQ_risk:= N_P^risk
10     foreach S ∈ Risks do
11         foreach UQ_i ∈ S do
12             if Size_ui ≠ ∞ then
13                 if Size_p = Size_ui and Step_p = Step_ui then
14                     PQ_risk:=Agg_PQ_risk
15                 if Size_p ≠ Size_ui or Step_p ≠ Step_ui then
16                     if (agg_p = SUM) or (agg_p = COUNT) then
17                         find m = ((Size_p − Size_ui)/Step_ui) and n = (Size_ui/Step_ui) and
                             α = ((m + n)/n)
18                         if m ∉ ℕ or n, α ∉ ℕ+ then
19                             PQ_risk:= N_P^risk, Subsets:=Subsets ∪UQ_i
20                     if (agg_p = MAX) or (agg_p = MIN) then
21                         find m = ((Size_p − Size_ui)/Step_ui)
22                         if m ∉ ℕ or (Size_p − (m + 1) × Step_ui ≱ 0) then
23                             PQ_risk:= N_P^risk, Subsets:=Subsets ∪UQ_i
24     σ_p = (Step_p)/(Step_p, Step_u2)
25     s_gcd = gcd(Step_u1, σ_p × Step_u2)
26     if Subsets ≠ ∅ then
27         foreach pair of UQ_i ∈ Subsets do
28             if (agg_p = SUM) or (agg_p = COUNT) then
29                 if Size_p ≠ Size_u1 + Size_u2 or Size_p − Size_u1 is not a multiple of s_gcd then
30                     PQ_risk:= N_P^risk
31             if (agg_p = MAX) or (agg_p = MIN) then
32                 if Size_p ≱ Size_u1 + Size_u2 or Size_p − Size_u1 is not a multiple of s_gcd then
33                     PQ_risk:= N_P^risk
34 return PQ_risk
```

Chapter 5

Explanation and negotiation of privacy risks

For data producers who lack familiarity with formal query language syntax, comprehending the results of formal framework presented in Section 4.4 can be quite daunting. To help data producers in understanding the privacy risks raised by utility queries from service providers, an explanation for each detected privacy risk is constructed based on examples that are built to make explicit how some (specific) answers to utility queries can induce an answer to a privacy query. Then, as the basis of a negotiation mechanism, some options for modifying utility queries are proposed to the data producer to remove the detected privacy risks.

In this chapter, we present our approach to help data producers in understanding and removing privacy risks. This chapter is organized as follows. In Section 5.1, we present the approach used for constructing the explanation of privacy risks. In Section 5.2, we present the approach used for constructing the negotiation options to remove privacy risks. In Section 5.3, we present the components and functionalities of the user-friendly interface that we have built for helping data producers to understand and negotiate the privacy risks. In Section 5.4, we present the methodology used for the evaluation of user interface along with its results.

5.1 Construction of explanation

In all the cases of privacy queries presented in Sections 4.3 and 4.4, a privacy risk is raised by a subset S of utility queries if there exists a freezing of the variables

in the union of the temporal graph patterns of the utility queries on which the evaluation of the conjunctive part of a privacy query at least provides an answer. Such a freezing is a synthetic dataset instantiating the body of a combination of utility queries in S, that is built by the algorithm of risk detection to enforce the existence of an answer to the privacy query. A freezing can be turned into an example explaining the risk by replacing the synthetic constants used in the freezing by plausible constants for the domain. Considering the aforementioned fact, we provide two levels of explanations for a privacy risk associated to a privacy query. The first level only points out the queries involved in a privacy risk by indicating the utility queries in a subset S identified by the algorithm as leading to a risk (i.e., inferring an answer to the privacy query from answers to a combination of utility queries in S). The second level exploits the synthetic dataset built from the freezing to show an example where some answers to the utility queries in S can reveal the presence of data from which an answer to the privacy query can be obtained.

To obtain a synthetic dataset, the synthetic constants are replaced with plausible constants by exploiting the ranges and domains of the involved properties as follows:

- For synthetic constants subject of a property, they are renamed using the name of the class declared as domain of the property appended with an integer serving as index in case of several synthetic constants of the same class having to be renamed. The same process is applied to rename the synthetic constants in the position of object for an object property.
- For synthetic constants in the position of value for a datatype property, since the corresponding ranges (integer, string or date) are in general too broad to determine the actual possible ranges of plausible values for each datatype property, we extend the ontology in order to make more precise the actual ranges of each datatype property of the domain by providing typical values (or ranges for integers and dates) within a comment associated to the property (using the rdfs property rdfs:comment). If a FILTER condition is applied to (a) synthetic constant(s), they are replaced by plausible values satisfying the corresponding constraint.

In our approach the examples for explaining the privacy risks are constructed by distinguishing the cases when a privacy query involved in a privacy risk is a conjunctive query or an aggregated conjunctive query or a temporal aggregated

conjunctive query.

The example explaining a privacy risk for a conjunctive privacy query illustrates:

- the synthetic answers corresponding to the output variables of utility queries;
- a synthetic dataset from which the synthetic answers of utility queries can be inferred;
- an answer to a privacy query that can be deduced from the synthetic answers of utility queries.

Example 26. *Let us consider the following privacy query PQ*1 *and the utility queries UQ*1 *and UQ*2 *of the scenario illustrated in Section 4.1:*
*PQ*1*: SELECT ?sm ?y*
WHERE { *?sm :associatedOccupier ?o . ?o :yearlyIncome ?y* }
*UQ*1*: SELECT ?sm ?o ?n*
WHERE { *?sm :associatedOccupier ?o . ?o :numberOfPersons ?n* }
*UQ*2*: SELECT ?o ?y*
WHERE { *?o :yearlyIncome ?y . o :owns ?s .*
FILTER (?y > 75000) }
*The example constructed for explaining a privacy risk raised for the privacy query PQ*1 *by the utility queries UQ*1 *and UQ*2 *is as follows:*
Answering utility queries may provide the following answers:
*(MeterId1, Occupier1, 1) for UQ*1
*(Occupier1, 75001) for UQ*2
Thus revealing the presence of the following facts in the data:
{ *MeterId1 :associatedOccupier Occupier1 . Occupier1 :numberOfPersons 1 .*
Occupier1 :yearlyIncome 75001 . Occupier1 :owns owns1 }
*From which an answer of PQ*1 *can be deduced, namely: (MeterId1, 75001)*

The example explaining a privacy risk for an aggregated conjunctive privacy query illustrates:

- the synthetic answers corresponding to the output and/or aggregate variables of utility queries;
- a synthetic dataset exhibiting a replication of a property (with different constant assigned to aggregate variable) over which an aggregate is computed and from which the synthetic answers of utility queries can be inferred;

- an answer to a privacy query that can be deduced from the answers of utility queries.

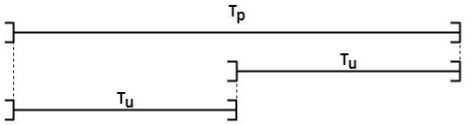
Example 27. *Let us consider the following privacy query Q_p and the utility query Q_{u_3} of Example 15.*
Q_p: SELECT ?y MAX(?n)
WHERE { ?sm :associatedBuilding ?b . ?b rdf:type :Apartment .
?sm :associatedOccupier ?o . ?o :yearlyIncome ?y .
?o :numberOfpersons ?n }
GROUP BY ?y
Q_{u_3}: SELECT ?o1 ?y1 ?a1 MAX(?n1)
WHERE { ?sm1 :associatedBuilding ?b1 . ?b1 rdf:type ?a1 .
?sm1 :associatedOccupier ?o1 . ?o1 :yearlyIncome ?y1 .
?o1 :numberOfpersons ?n1 }
GROUP BY ?o1 ?y1 ?a1
The example constructed for explaining a privacy risk raised for the privacy query Q_p by the utility query Q_{u_3} is as follows:
Answering Q_{u_3} may provide the following answer:
(Occupier1, 75000, :Apartment, 2)
Thus revealing the presence of the following facts in the data:
{ MeterId1 :associatedBuilding Building1 . Building1 rdf:type :Apartment .
MeterId1 :associatedOccupier Occupier1 . Occupier1 :yearlyIncome 75000 .
Occupier1 :numberOfPersons 1 . Occupier1 :numberOfPersons 2 }
As Q_{u_3} and Q_p compute the same aggregate, so the following answer of Q_p can be deduced: (75000, 2)

The example explaining a privacy risk for a temporal aggregated conjunctive privacy query illustrates:

- the synthetic answers corresponding to the output, aggregate and timestamp variables of utility queries;
  - multiple synthetic answers are constructed corresponding to different time windows of one or more utility queries that cover exactly a time window of a privacy query, thus allowing the computation of an answer to a privacy query.
- an answer to a privacy query that can be computed from the answers of utility queries;

- a figure demonstrating the building of a time window of a privacy query from the union of time windows of one or more utility queries.

Example 28. *Let us consider the following privacy query PQ2 and the utility queries Q_{u_1} and Q_{u_2} of Example 18:*
PQ2: SELECT ?timeWindowEnd SUM(?c)
WHERE (?sm :consumption ?c, ?ts)
GROUP BY ?timeWindowEnd
TIMEWINDOW (6h, 6h)
Q_{u_1}: SELECT ?timeWindowEnd SUM(?c1)
WHERE {(?sm1 :consumption ?c1, ?ts1)}
GROUP BY ?timeWindowEnd
TIMEWINDOW (4h, 2h)
Q_{u_2}: SELECT ?timeWindowEnd SUM(?c2)
WHERE {(?sm2 :consumption ?c2, ?ts2)}
GROUP BY ?timeWindowEnd
TIMEWINDOW (2h, 1h)
The example constructed for explaining a privacy risk raised for the privacy query PQ2 by the utility queries Q_{u_1} and Q_{u_2} is as follows:
Answering utility queries over two contiguous time windows that cover exactly a time window of PQ2, may provide the following answers:
(07-08-2023 12:22:33, 6) for Q_{u_1}
(07-08-2023 08:22:33, 4) for Q_{u_2}
As utility queries and PQ2 compute the same aggregate, these two answers to utility queries can be combined to compute the following answer of PQ2:
(07-08-2023 12:22:33, 10)



5.2 Construction of negotiation options

The explanation provided in the previous section points out the queries involved in a privacy risk and the results based on the formal framework presented in Section 4.4 help in building a synthetic dataset revealing the privacy risk. Several options are constructed for removing the privacy risks by negotiating the utility queries involved in privacy risks. The negotiated utility queries are supposed to serve as

the basis for finding an acceptable trade-off in terms of utility while guaranteeing data privacy for each data producer, so these options are constructed with an objective of minimal utility loss.

Given a privacy query, several options are proposed to negotiate each utility query involved in the corresponding privacy risk. In cases where multiple utility queries are involved in a privacy risk, modifying a single utility query using one of the provided options is sufficient to remove the associated privacy risk.

In our approach, the negotiation options are constructed by distinguishing the cases when a privacy query involved in a privacy risk is a conjunctive query or an aggregated conjunctive query or a temporal aggregated conjunctive query. The options constructed for negotiating each utility query contributing to a privacy risk raised for a conjunctive privacy query are:

- refusing to answer a utility query;
- removing the output variables from a utility query that either correspond to the output variables or to the join variables of a privacy query;
- generalizing some properties, for which rules of generalization have been user-defined (see Section 3.2.2).

Example 29. *Let us consider the following privacy and utility queries PQ1, UQ1 and UQ2 of the scenario illustrated in Section 4.1:*
PQ1: SELECT ?sm ?y
WHERE { ?sm :associatedOccupier ?o . ?o :yearlyIncome ?y }
UQ1: SELECT ?sm ?y ?n
WHERE { ?sm :associatedOccupier ?y . ?y :numberOfPersons ?n }
UQ2: SELECT ?o ?y
WHERE { ?o :yearlyIncome ?y .
o :owns ?s . FILTER (?y > 75000) }
The following options will be constructed to negotiate the utility queries UQ1 and UQ2 in order to remove a privacy risk raised for the privacy query PQ1 of the Example 26.
Options for negotiating the utility query UQ1:
- Refuse to answer this query (privacy risk for PQ1).
- Remove ?sm from the output (privacy risk for PQ1).
- Remove ?o from the output(privacy risk for PQ1).
Options for negotiating the utility query UQ2:

- Refuse to answer this query (privacy risk for PQ1).
- Remove ?y from the output (privacy risk for PQ1).
- Remove ?o from the output (privacy risk for PQ1).
- Generalize the property :yearlyIncome with the property :yearlyIncomeRange (privacy risk for PQ1).

In addition to the options listed in the first case, the options provided for negotiating each utility query contributing to a privacy risk raised for an aggregated conjunctive privacy query are:

- removing the output variable from a utility query corresponding to the aggregate variable of a privacy query (in cases where the subset of utility queries involved in a privacy risk consists of only conjunctive queries);
- modification of the aggregate function corresponding to the same aggregate function of a privacy query (in cases where the subset of utility queries involved in a privacy risk consists of an aggregated conjunctive query).

Example 30. *Let us consider the following privacy query Q_p and utility query Q_{u_3} of Example 27:*
Q_p: SELECT ?y MAX(?n)
WHERE { ?sm :associatedBuilding ?b . ?b rdf:type :Apartment .
?sm :associatedOccupier ?o . ?o :yearlyIncome ?y .
?o :numberOfpersons ?n }
GROUP BY ?y
Q_{u_3}: SELECT ?o1 ?y1 ?a1 MAX(?n1)
WHERE { ?sm1 :associatedBuilding ?b1 . ?b1 rdf:type ?a1 .
?sm1 :associatedOccupier ?o1 . ?o1 :yearlyIncome ?y1 .
?o1 :numberOfpersons ?n1 }
GROUP BY ?o1 ?y1 ?a1
The following options will be constructed to negotiate the utility query Q_{u_3} in order to remove a privacy risk raised for the privacy query Q_p.
Options for negotiating the utility query Q_{u_3}:
- Refuse to answer this query (privacy risk for Q_p).
- Remove ?y1 from the output (privacy risk for Q_p).
- Generalize the property :yearlyIncome with the property :yearlyIncomeRange (privacy risk for Q_p).
- Generalize the property :numberOfPersons with the property :familySize (privacy risk for Q_p).

- Replace the aggregate MAX with the aggregate SUM or MIN (privacy risk for Q_p).

In addition to the options mentioned above for both cases, an additional option provided for negotiating each utility query contributing to a privacy risk raised for a temporal aggregated conjunctive privacy query is the modification of the size or step specified in the time window of a temporal aggregated conjunctive utility query.

Example 31. *Let us consider the following privacy query PQ2 and the utility queries Q_{u_1} and Q_{u_2} of Example 28:*
PQ2: SELECT ?timeWindowEnd SUM(?c)
WHERE (?sm :consumption ?c , ?ts)
GROUP BY ?timeWindowEnd
TIMEWINDOW (6h, 6h)
Q_{u_1}: SELECT ?timeWindowEnd SUM(?c1)
WHERE {(?sm1 :consumption ?c1, ?ts1)}
GROUP BY ?timeWindowEnd
TIMEWINDOW (4h, 2h)
Q_{u_2}: SELECT ?timeWindowEnd SUM(?c2)
WHERE {(?sm2 :consumption ?c2, ?ts2)}
GROUP BY ?timeWindowEnd
TIMEWINDOW (2h, 1h)
The following options will be constructed to negotiate the utility query Q_{u_1} and Q_{u_2} in order to remove a privacy risk raised for the privacy query PQ2.
Options for negotiating the utility query Q_{u_1}:
- Refuse to answer this query (privacy risk for PQ2).
- Replace the aggregate SUM with the aggregate MAX or MIN (privacy risk for PQ2).
- Modify the size or the step defined in the time window (privacy risk for PQ2).
Options for negotiating the utility query Q_{u_2}:
- Refuse to answer this query (privacy risk for PQ2).
- Replace the aggregate SUM with the aggregate MAX or MIN (privacy risk for PQ2).
- Modify the size or the step defined in the time window (privacy risk for PQ2).

5.3 Interactive user interface

PrivEx is an interactive user-friendly interface, built[1] on top of the implementation of formal framework presented in Section 4.4. *PrivEx* is run locally by each data producer and it offers the following functionalities to a data producer:

- it provides a form-based interface for constructing the privacy queries;
- it detects the privacy risks and explains each detected privacy risk by providing an example;
- it provides a negotiation interface that provides several options for modifying the utility queries to remove the privacy risks.

Widgets are key elements of user interface design, serving as a direct means of user interaction. Figure 5.1 presents the several widgets used to design the components of *PrivEx* interface, enabling data producers to interact and access its functionalities.

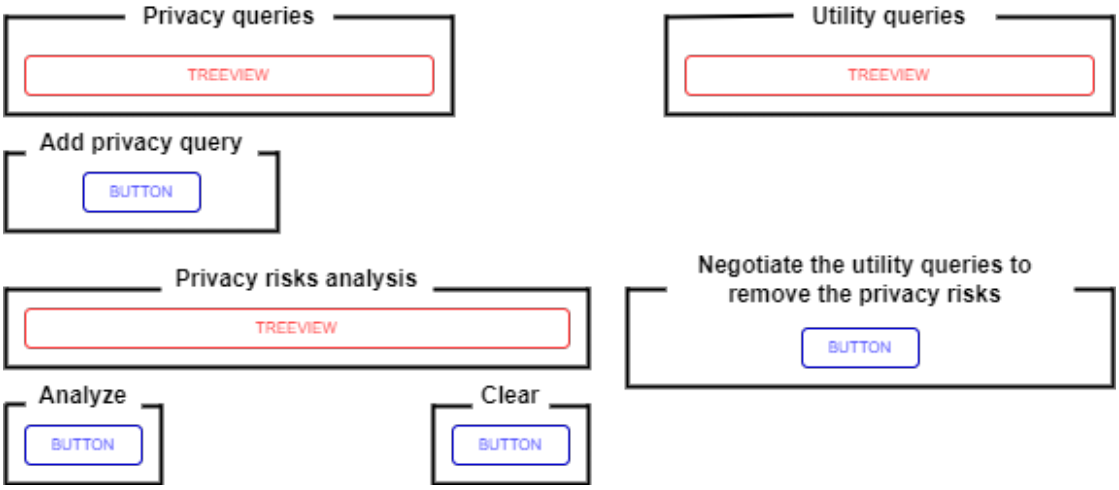


Figure 5.1: PrivEx interface design

The purpose of each component of the *PrivEx* interface design is described below. **Privacy queries:** displays the textual form and SPARQL-like syntax of each privacy query that is constructed using the form-based interface.

[1]We used the python tkinter library: `https://docs.python.org/3/library/tkinter.html` and our code is available at GitHub repository: `https://github.com/repository-code/PrivEx`

Utility queries: facilitates the interpretation of each utility query provided by the service provider in a textual form, accompanied by its SPARQL-like syntax.
Add privacy query: provides access to the form-based interface that enables the construction of privacy queries.
Analyze: analyzes the given set of privacy queries with a given set of utility queries to detect privacy risks.
Privacy risks analysis: displays the detected privacy risks and their explanations using two different levels.
Clear: clear the displayed results of the privacy risks analysis.
Negotiate the utility queries to remove the privacy risks: provides access to the negotiation interface that lists several options for modifying the utility queries to remove the privacy risks.

In this section, we will illustrate the functionalities of *PrivEx* through the queries of the smart meter scenario presented in Section 4.1. This section is structured as follows. In section 5.3.1, we present the form-based interface that guides data producers in construction of privacy queries. In Section, 5.3.2, we present the interface specifically designed for detecting and explaining privacy risks to data producers. In Section 5.3.3, we present the negotiation interface, which enables data producers to remove privacy risks by negotiating the utility queries.

5.3.1 Construction of privacy queries

The form-based interface is designed to facilitate data producers in the step by step construction of privacy queries through continuous guidance from ontology. The form-based interface is structured with several components filled with widgets as presented in the Figure 5.2. The user interaction is done by several types of widgets, such as text box, label, button, drop-down list and checkbox. Widgets are linked with the underlying ontology and the user input provided at each step.

The purpose of each component of the form-based interface is described below.
Select schema: allows to choose the ontology to be guided by in the process of query construction while displaying the properties extracted from it.
Privacy query in words: allows to enter the textual description of a query to be constructed.
Properties: allows to choose the interesting properties to be included in the temporal graph pattern of a query to be constructed. Choosing properties allows the automatic construction of a temporal graph pattern of a query in a formal query

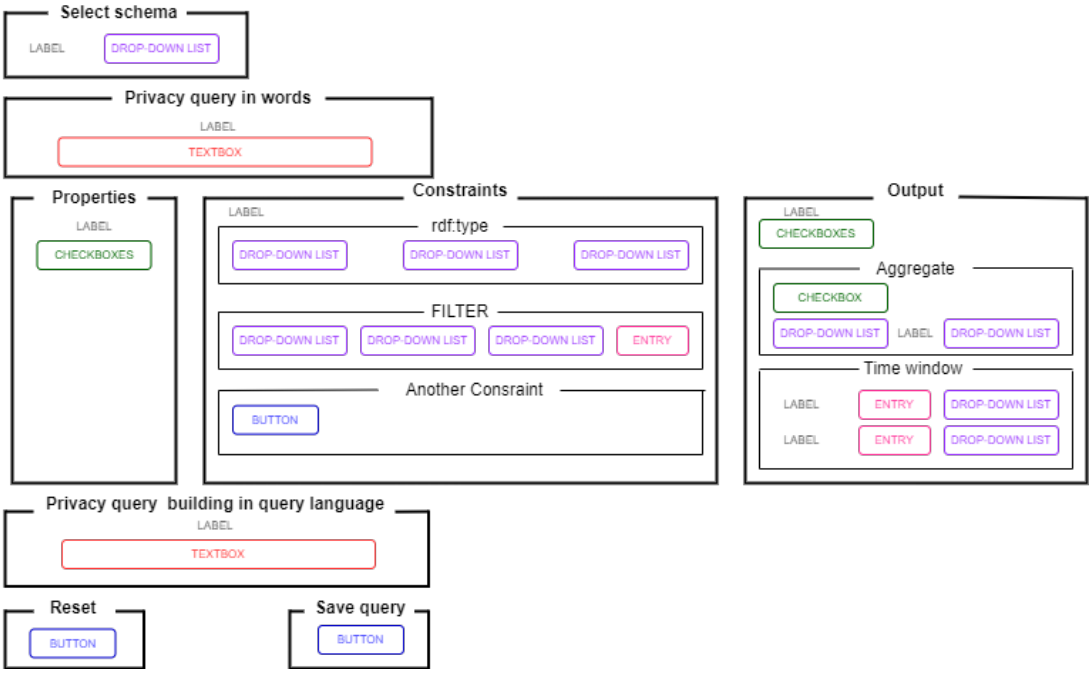


Figure 5.2: Form-based interface design

language syntax. The construction of each temporal pattern is guided by the ontology and specific rules defined for each property type, namely object, datatype and dynamic. All properties are linked to its type, domain and range specified in the underlying ontology. The temporal pattern for each selected property is constructed by examining its property type and exploiting its domain and range. The approach followed for constructing a temporal pattern for each property type is as follows:

- object property: the subject is a mirror of the property's domain, constructed as a variable. The property is a mirror of the chosen property, constructed as a URI. The object is a mirror of the property's range, constructed as a variable.
- datatype property: the subject is a mirror of the property's domain, constructed as a variable. The property is a mirror of the chosen property, constructed as a URI. The object is a mirror of the chosen property, constructed as a variable.
- dynamic property: the subject is a mirror of the property's domain, constructed as a variable. The property is a mirror of the chosen property, constructed as a URI. The object is a mirror of the chosen property, constructed as a variable. Other than subject, property and object, the timestamp variable is also constructed and included in a temporal pattern.

Constraints: allows to add constraints (if any) either by constructing a FILTER expression for filtering specific data values or by constructing temporal patterns for filtering the resources of specific type. Constraints are applied one at a time to the generated variables. Upon selection of the variable, the data producer is guided to enter choices for filtering depending on the property type it is associated to.
Output: allows to choose the output variables, specify the aggregate term and the time window definition and include them in the query being constructed. If the aggregate is being computed on a dynamic property then the interface guides the data producer to define the time window definition by specifying the *Size* and *Step* to obtain the time intervals over which aggregation must be computed.
Privacy query building in query language: demonstrates the automatic construction of the query in formal query language syntax through guidance from ontology and the user input provided at each step.
Reset: resets the form-based interface to construct a new query.
Save query: displays the constructed query in the list of privacy queries.

The Figure 5.3 presents the steps followed for the construction of the privacy query $PQ2$ of the scenario illustrated in Section 4.1.

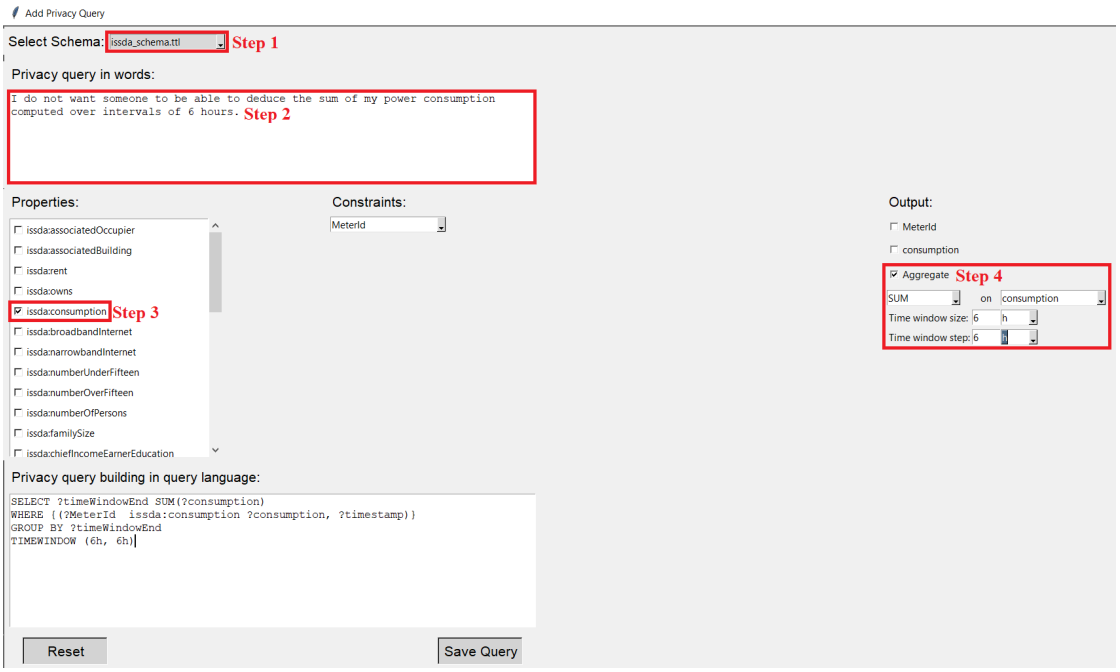


Figure 5.3: Steps followed for the construction of privacy query PQ2

5.3.2 Detection and explanation of privacy risks

The *PrivEx* interface design illustrated in Figure 5.1 presents the components specifically designed for detecting and explaining privacy risks. The data producer can perform the privacy risks analysis by clicking the *Analyze* button. In case if privacy risks are detected then each privacy risk is explained using two different levels. The first level points out the privacy query violated by one or more utility queries. The second level explains which answers of privacy query can be inferred from the utility queries involved in a privacy risk and this is explained by providing an example built from the synthetic data graph and the queries involved in a privacy risk as already detailed in the Section 5.1. For example, the Figure 5.4 explains the privacy risk at two different levels for the following privacy query $PQ2$ that is constructed using form-based interface in the last section and the utility query $UQ3$ of the scenario presented in Section 4.1.

PQ2 and UQ3 of the scenario presented in Section 4.1

```
PQ2: SELECT ?timeWindowEnd SUM(?consumption)
     WHERE { (?MeterId :consumption ?consumption, ?timestamp) }
     GROUP BY ?timeWindowEnd
     TIMEWINDOW (6h, 6h)
UQ3: SELECT ?sm ?timeWindowEnd SUM(?c)
     WHERE { (?sm :consumption ?c, ?ts) }
     GROUP BY ?sm ?timeWindowEnd
     TIMEWINDOW (3h, 1h)
```

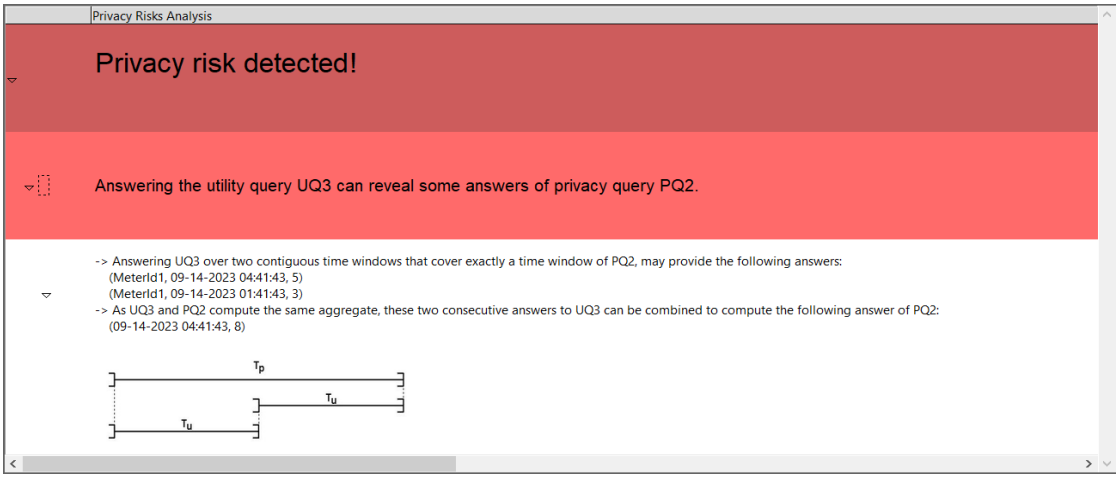


Figure 5.4: Explanation of detected privacy risk for privacy query PQ2 at two different levels

5.3.3 Negotiation for removing privacy risks

The negotiation interface helps to remove privacy risks by providing several options for negotiating the utility queries involved in privacy risks. The construction of the options for negotiating the utility queries involved in privacy risks is already explained in the Section 5.2. The interface also demonstrates the impact of options being selected by constructing and displaying the modified versions of the utility queries that can be sent to the data consumer as a basis for negotiation. The main components of the interface are presented in Figure 5.5. For simplicity, in Figure 5.5, we depict only the components for negotiating a single utility query. However, in reality, the user interface displays as many components as there are utility queries involved in privacy risks.

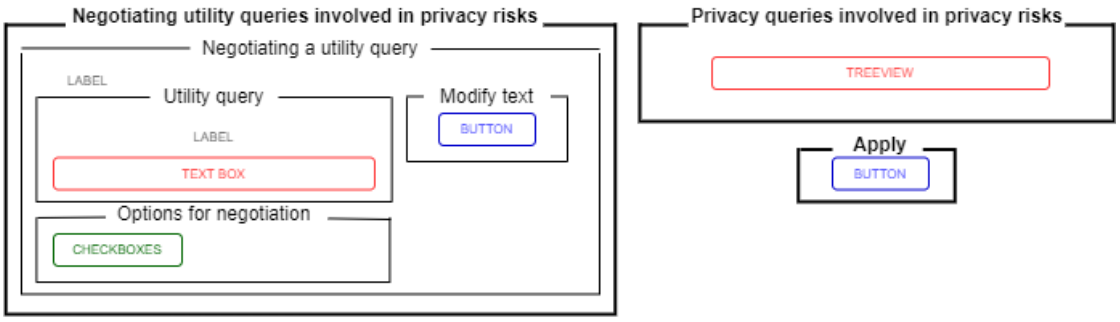


Figure 5.5: User interface design for negotiating utility queries

The purpose of each component of the interface design is described below.
Utility query: interprets each utility query involved in a privacy risk in textual form, along with its corresponding SPARQL-like syntax.
Options for negotiation: provides several options for negotiating a utility query involved in a privacy risk, either by refusing to answer it, or by modifying its output variables or by generalizing its properties, or by changing the aggregate function, or by changing the time window size or step. For each chosen option it displays the new modified version of a utility query in SPARQL-like syntax on interface.
Modify text: allows the modification of the textual description of the utility query being negotiated.
Privacy queries involved in privacy risks: interprets each privacy query involved in a privacy risk in textual form, along with its corresponding SPARQL-like syntax.
Apply: applies the modifications to the utility query displayed in the list of utility

queries.

The Figure 5.6 presents the options provided for negotiating the utility query UQ3 to mitigate the privacy risk for the privacy query $PQ2$ (presented in last section) and it also presents the modified version of utility query UQ3 which is constructed by changing the step between each consumption computation from 1 hour to 2 hours.

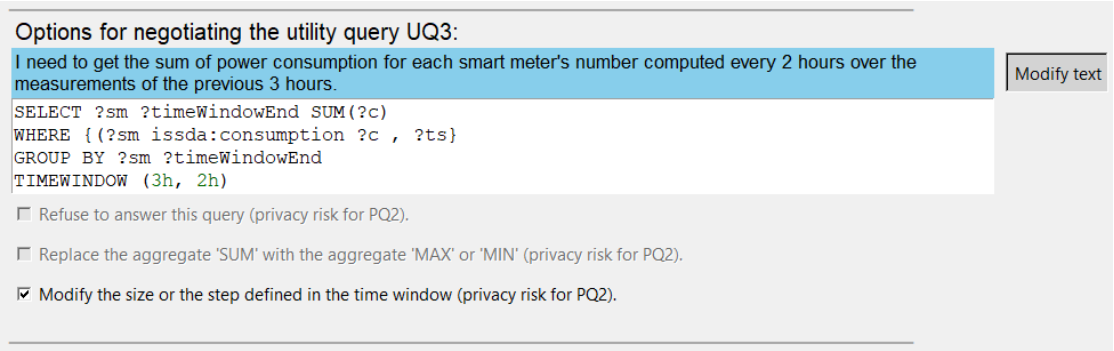


Figure 5.6: Negotiating the utility query UQ3

5.4 User study evaluation

To evaluate the practical usability and effectiveness of the *PrivEx* interface, we conducted a user study with the following three objectives:

1. Evaluate its usefulness in interpreting privacy and utility policies;
2. Evaluate its effectiveness in explaining detected privacy risks;
3. Evaluate its utility in building and testing privacy queries.

This section is organized as follows. In Section 5.4.1, we present the methodology that we employed to assess the effectiveness of our framework via a user study. In Section 5.4.2, we present the results of conducted user study.

5.4.1 Evaluation methodology

For our user study, we created[2] an online questionnaire with 5 consecutive sections to be filled by participants. The first section consisted of demographic questions.

[2]We used Google Forms: `https://www.google.com/forms/about/`

The next three sections were mandatory and focused on answering the questionnaire based on the same scenario presented in Section 4.1. The fifth section, added at the end of questionnaire was non-mandatory and required downloading and executing our interface to answer the remaining questions related to building and testing of privacy queries. The full questionnaire is presented in Appendix A.

For evaluating the first objective of the user study, participants were introduced to a scenario to gauge its relevance to their daily lives. The scenario was explained with the help of privacy and utility policies. The participants were asked to answer the following three questions divided into two sections:
Question 1: How realistic do you find this scenario?
Question 2: To what degree do you understand these utility queries?
Question 3: To what degree do you understand these privacy queries?

For evaluating the second objective of the user study, participants were presented with the explanation of privacy risks detected for the privacy queries $PQ1$ and $PQ2$. The participants were asked to answer the following two questions:
Question 4: How do you find the explanation of this privacy risk raised by the privacy query PQ1?
Question 5: How do you find the explanation of this privacy risk raised by the privacy query PQ2?

For evaluating the third objective of the user study, participants were provided with a short tutorial guiding them through the main steps of building and testing privacy queries using the *PrivEx* interface. First, participants were instructed to build two privacy queries using the form-based interface, where the one query $PQ3$ was a conjunctive query and the other $PQ4$ was a temporal aggregated conjunctive query. Afterward, participants were prompted to analyze the privacy risks associated to privacy queries they had built. Finally, participants were asked to answer the following questions:
Question 6: Provide the query syntax for privacy queries they had built.
Question 7: How difficult did you find to build these two privacy queries?
Question 8: What is the result of privacy risks analysis for the privacy queries PQ3 and PQ4?

Our user study was primarily aimed at assessing the effectiveness of the *PrivEx* interface among participants, with or without prior knowledge of Semantic Web languages such as SPARQL or RDF. In conducting this evaluation, we required to achieve a distribution that ensures that the evaluation of the *PrivEx* interface

is unbiased, capturing insights from both groups of participants. Additionally, to enrich the comprehensiveness of our insights, we further categorized these participant groups based on their age and gender. We looked for volunteer participants by disseminating the purpose of user study along with a hyperlink to an online questionnaire via email. The email was sent to various recipients including research groups, Masters students, and non-IT professionals to ensure a wide range of participants. The demographic summary of the respondents who participated in both the mandatory and non-mandatory parts of the online questionnaire, along with the groups derived from the data, are presented in the Table 5.1.

Table 5.1: Summary of Demographics

Groups		**Mandatory part**	**Non-mandatory part**
Total participants		57	22
Age:	20-29 years	28.0%	36.4%
	30-39 years	43.0%	36.4%
	40-49 years	19.0%	27.2%
	>= 50 years	10.0%	0.0%
Gender:	Male	47.4%	40.9%
	Female	52.6%	59.1%
Knowledge of Semantic Web languages:	Yes	36.8%	36.4%
	No	68.2%	63.6%

5.4.2 Evaluation results

We now present the results of our user study, which consisted of 8 questions as discussed in the previous section. Sections 5.4.2.1 and 5.4.2.2 present the results for mandatory part of user study, which is comprised of 5 questions divided into three sections. Section 5.4.2.3 presents the results for non-mandatory part, which is comprised of 3 questions.

5.4.2.1 Results of evaluating usefulness in policies interpretation

The responses to "Question 1" were collected using a Likert scale ranging from 1 to 5, where the option "1" represents "Completely realistic", and the option "5" represents "Completely unrealistic". Figure 5.7 shows the results. We observe that, on average, the majority of participants chose either "completely realistic or "somewhat realistic", i.e., options 1 and 2 in the Likert scale. Furthermore, it turns out that a significant majority of participants, regardless of their age, gender or familiarity with Semantic Web languages chose options 1 and 2 on the Likert

scale. The results clearly show that the majority of participants agreed that the scenario was realistic, emphasising its applicability.

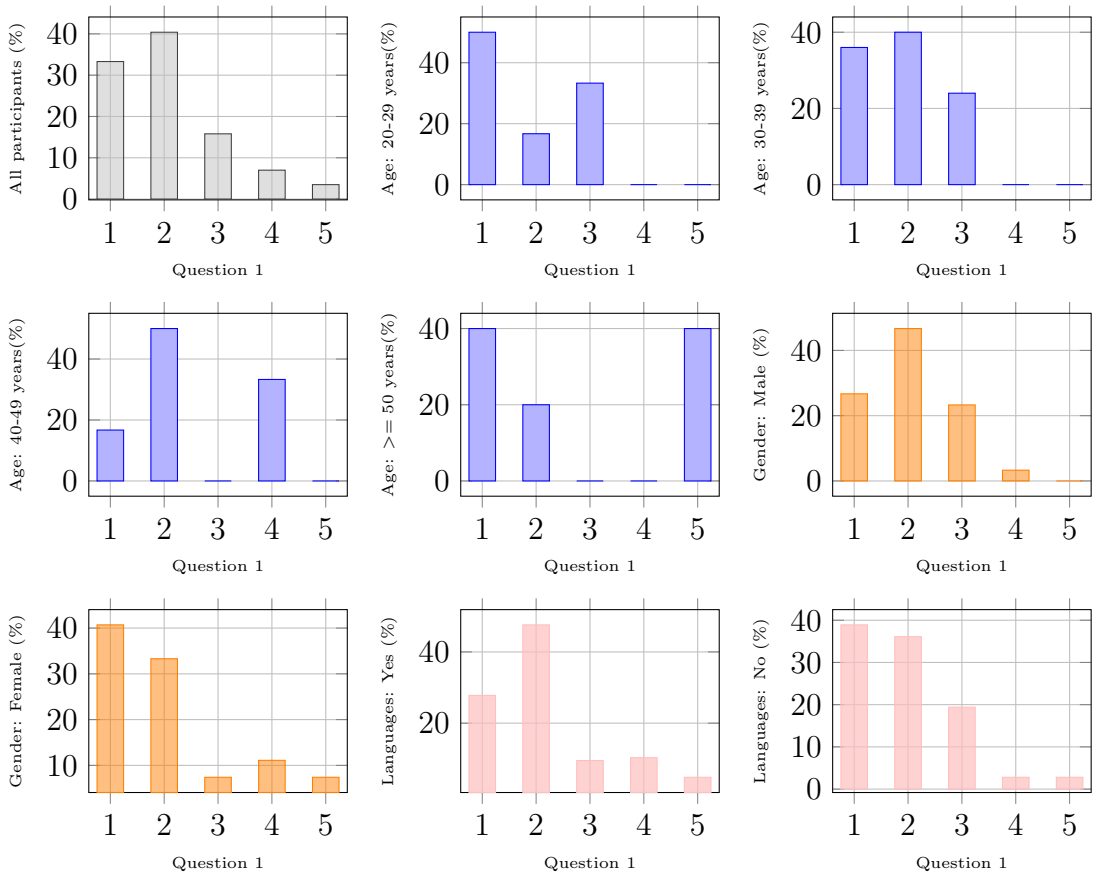


Figure 5.7: Results for Question 1

To assess the interpretability of utility queries among participants, the responses to "Question 2" were collected in the form of five-option scale, where first option represents "I completely understand these utility queries" and last option represents "I do not understand any of these utility queries". Figure 5.8 shows the results. Notably, 50.9% of the participants chose the first option while no participant chose the last option, indicating that the majority of them either completely or partially understood the queries. Furthermore, majority of the participants in the age group between 20 to 29 years chose the first option, whereas understanding the queries appeared to be more challenging for those aged 40 and above. Additionally,the results show insignificant variance based on gender. Moreover, the results shows that 66.7% of participants familiar with Semantic Web languages

opted for the first option, in contrast to 41.7% of the participants unfamiliar with Semantic Web languages who made the same choice.

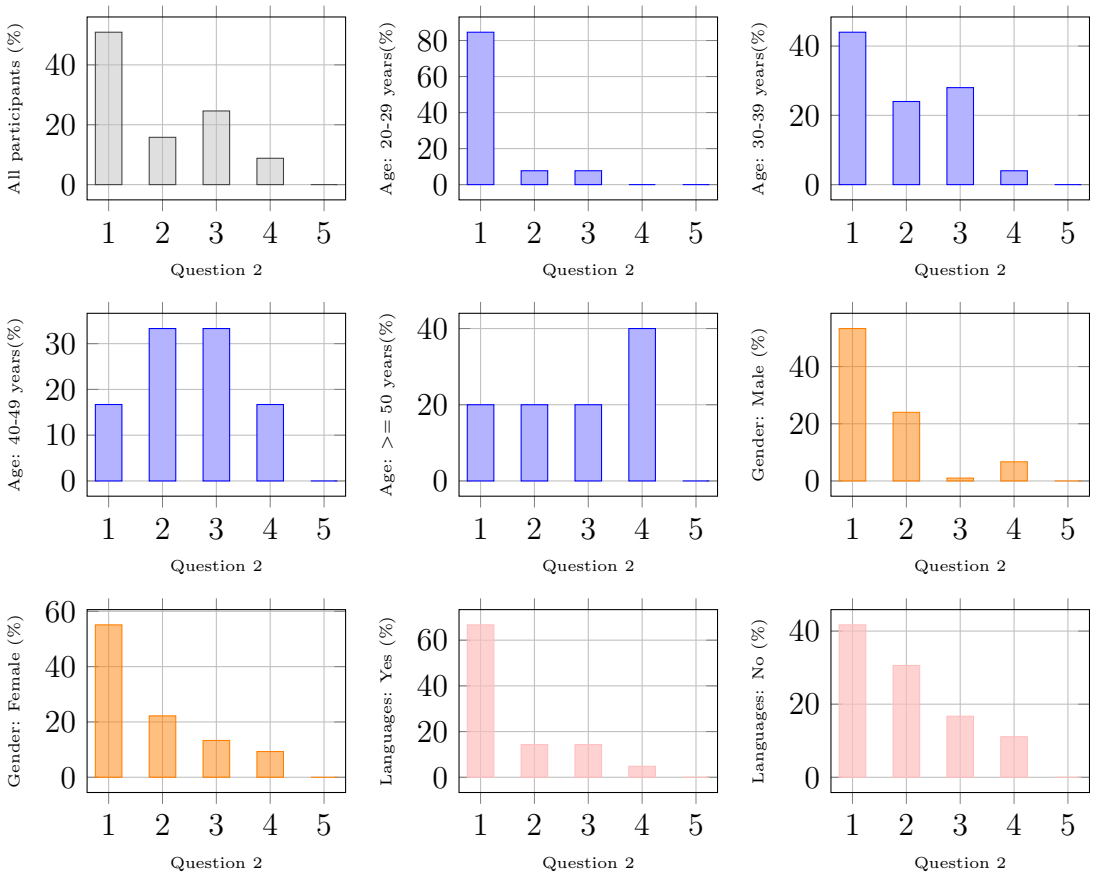


Figure 5.8: Results for Question 2

To evaluate the interpretability of privacy queries among participants, the responses to "Question 3" were collected in the form of five-option scale, where first option represents "I completely understand both privacy queries" and last option represents "I do not understand any of these privacy queries". Figure 5.9 shows the results. We observed that 61.4% of the participants chose the first option while no participant chose the last option, indicating that the majority of them either completely or partially understood the queries. Furthermore, majority of the participants in the age groups between 20 to 39 years chose the first option, whereas understanding the queries seemed to be more challenging for those aged 40 and above. Additionally, the results show insignificant variance based on gender. Moreover, the results shows that 66.7% of participants who are familiar with

Semantic Web languages opted for the first option, in contrast to 57.1% of the participants unfamiliar with Semantic Web languages who made the same choice.

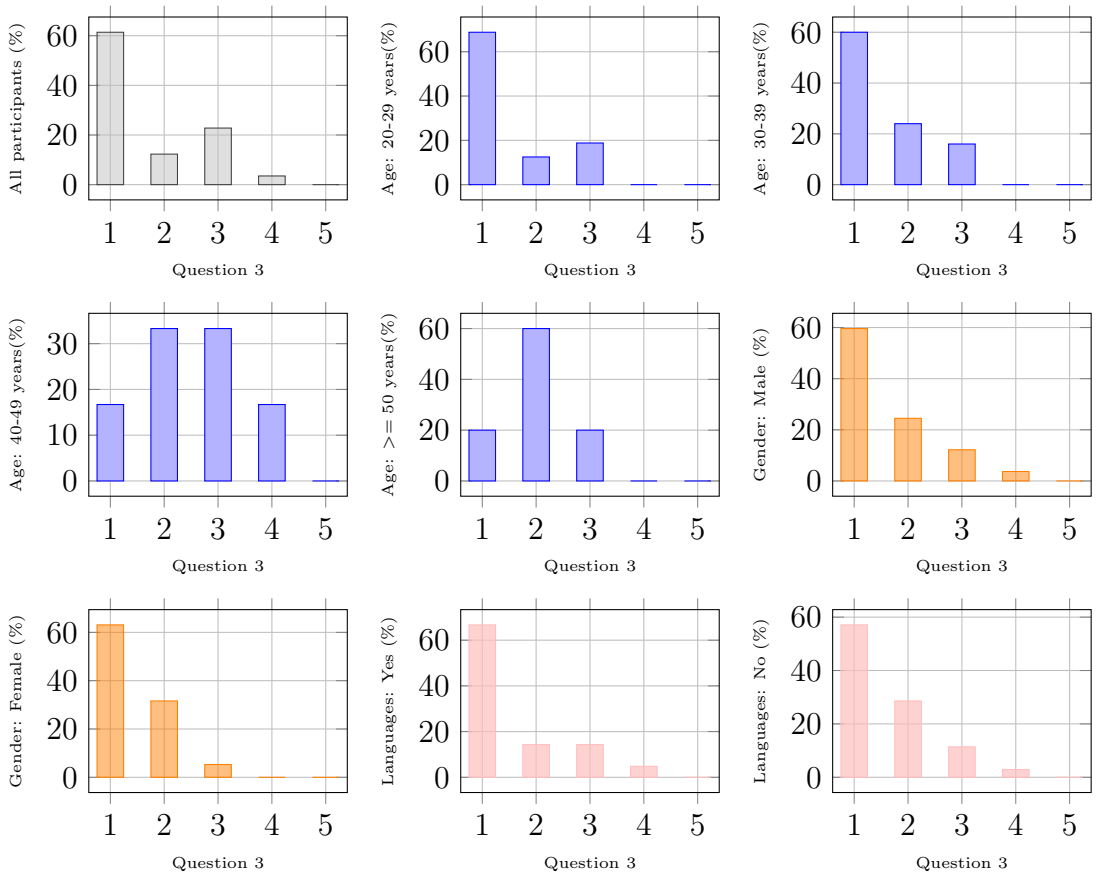


Figure 5.9: Results for Question 3

5.4.2.2 Results of evaluating effectiveness in explaining privacy risks

To assess the understanding of the explanation provided for privacy risk for privacy query PQ1, responses to "Question 4" were collected using a Likert scale that ranged from 1 to 5, where the option "1" represents "Completely helpful", and the option "5" represents "Not at all helpful". Figure 5.10 shows the results. The majority of participants, specifically 40.4% indicated the explanation was "Very helpful" and 28.1% of participants found it "Somewhat helpful", i.e., options 2 and 3 in the Likert scale, whereas 21.1% of participants chose option 1. Furthermore, we observed a consistent trend across the age and gender groups, with minor variance in results, as majority of participants from each group also chose

option 2 and 3. Moreover, the results indicate that participants familiar with Semantic Web languages chose option 1, whereas majority of those unfamiliar with Semantic Web languages chose option 2. Overall, both groups exhibited a nearly identical percentage, as the majority of participants in each group indicated that the explanation was helpful in understanding a privacy risk.

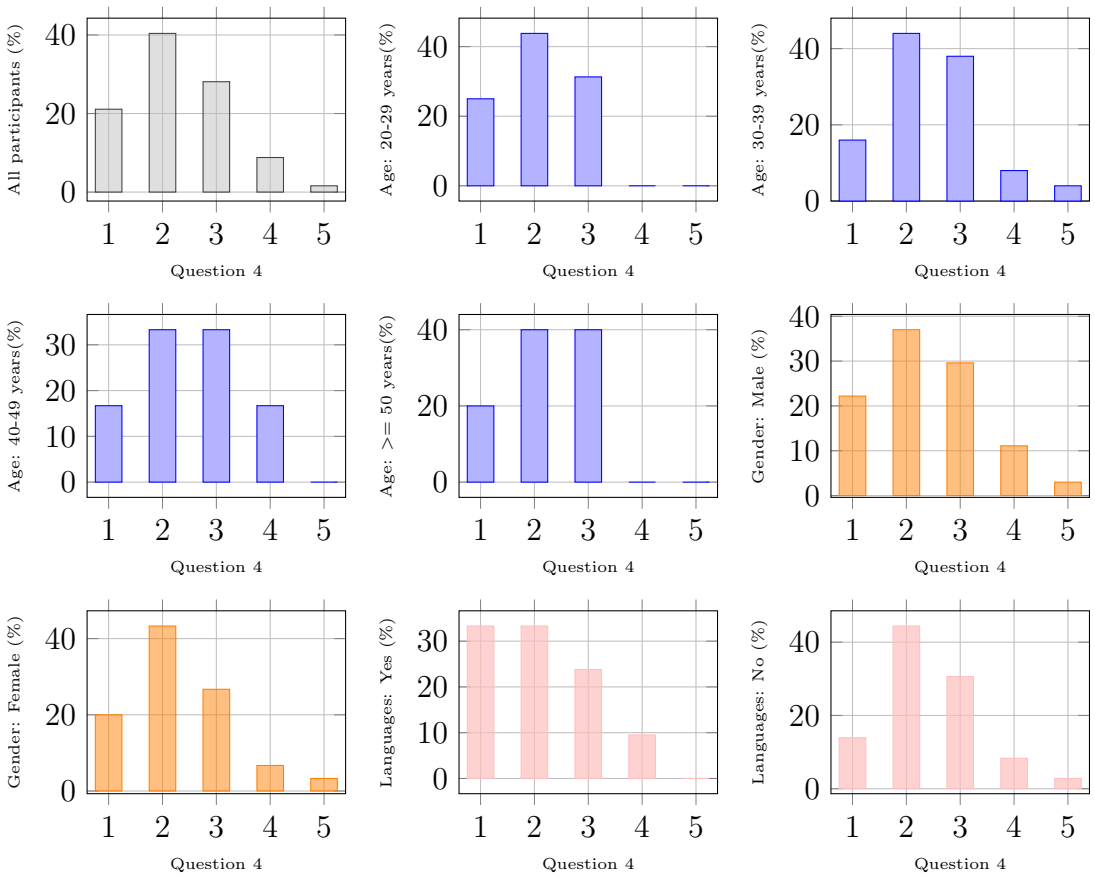


Figure 5.10: Results for Question 4

To assess the understanding of the explanation provided for privacy risk for privacy query PQ2, responses to "Question 5" were collected using a Likert scale that ranged from 1 to 5, where the option "1" represents "Completely helpful", and the option "5" represents "Not at all helpful". Figure 5.11 shows the results. The majority of participants, specifically 36.8% indicated the explanation was "Very helpful" and 29.4% of participants found it "Somewhat helpful", i.e., options 2 and 3 in the Likert scale, whereas 17.5% of participants chose option 1. In contrast to the results of Question 4, it appeared to be more challenging to understand a

privacy risk associated to the temporal aggregated conjunctive query $PQ2$. Furthermore, we observed a consistent trend across the age groups between 20 to 39 years, as most of them chose option 3. However, among those aged 40 and above, found the explanation to be "Little helpful", i.e., option 4. Moreover, none of the female participants chose option 1 and a few of them also opted for option 5, which contrasts with the choices made by male participants. Participants who were familiar with Semantic Web languages found the explanation to be more helpful compared to the those who were unfamiliar with Semantic Web languages, although the results showed only slight variation between the two groups.

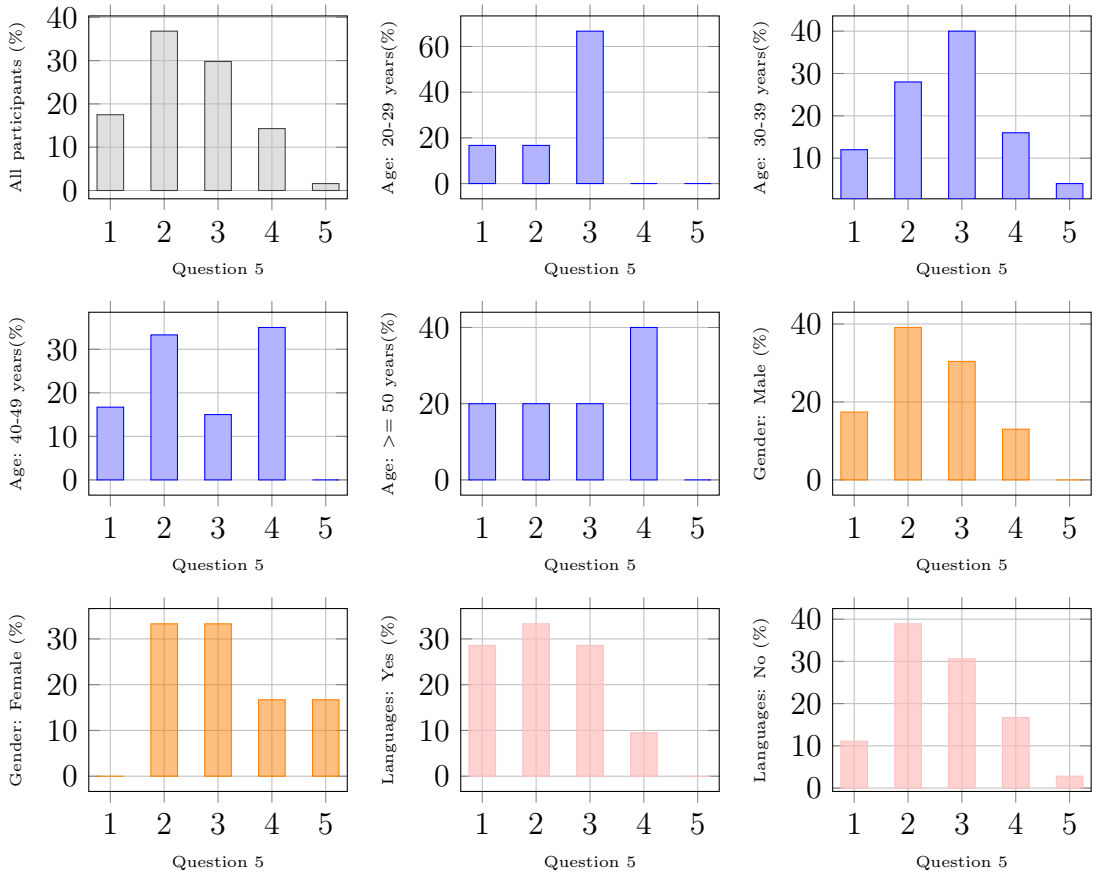


Figure 5.11: Results for Question 5

5.4.2.3 Results of evaluating utility in query building and testing

A total 22 participants responded to non-mandatory questions designed to measure the usefulness of *PrivEx* interface in building and testing privacy queries.

To assess the utility of the form-based interface in building privacy queries, we evaluated the number of participants who successfully built correct queries.

The outcomes of the responses to "Question 6" are presented using a Likert scale with two options "correct" and "incorrect" i.e., options 1 and 2. Figure 5.12 shows the results. Overall 81.8% of the participants built the correct conjunctive query $PQ3$, whereas 86.3% of the participants built the correct temporal aggregated conjunctive query $PQ4$. Furthermore, we observed a consistent trend across the age groups between 20 to 39 years, as 75% of the participants built both the queries correct. However, among those aged 40 and 49 years, 50% built $PQ3$ correctly and 66.6% built $PQ4$ correctly. We also observed that all female participants built $PQ4$ correctly, while a higher percentage of male participants correctly built $PQ3$. Moreover, all participants who were familiar with Semantic Web languages correctly built $PQ4$. In comparison, a higher percentage (i.e., 84.6%) of those unfamiliar with Semantic Web languages correctly built $PQ3$.

The responses to "Question 7" were collected in the form of five-option scale, where first option represents "I found it easy" and last option represents "I did not manage to build any of these two queries". Figure 5.13 shows the results. We observed that 63.6% of the participants chose the first option while 9.1% of the participant chose the last option, indicating that they found the form-based interface helpful in building privacy queries. An equal percentage of participants specifically 13.6%, opted for both option 3, "Difficult to build PQ4" and option 4 "Difficult to build both PQ3 and PQ4". Furthermore, none of the participants in the both age groups between 20 to 29 years and 40 to 49 years chose the last option. Additionally, a higher percentage (i.e., 66.7%) of male participants chose option 1 and none of them opted for last option, in contrast to the female participants. We also observed that participants who were not familiar with Semantic Web languages also found the form-based interface easy to use. Specifically, 69.9% of them chose the first option, which is relatively close in percentage to the participants familiar with Semantic Web languages, where the percentage was 75%.

Question 8 was designed to evaluate whether participants could successfully detect the privacy risks for the newly built privacy queries, $PQ3$ and $PQ4$, using the $PrivEx$ interface. The responses to "Question 8" were collected in the form of five-option scale. If a participant managed to build accurate queries, then the correct option representing the analysis results of privacy risks for both $PQ3$ and $PQ4$ was the first option, which is "PQ3 raises privacy risk". We observed that

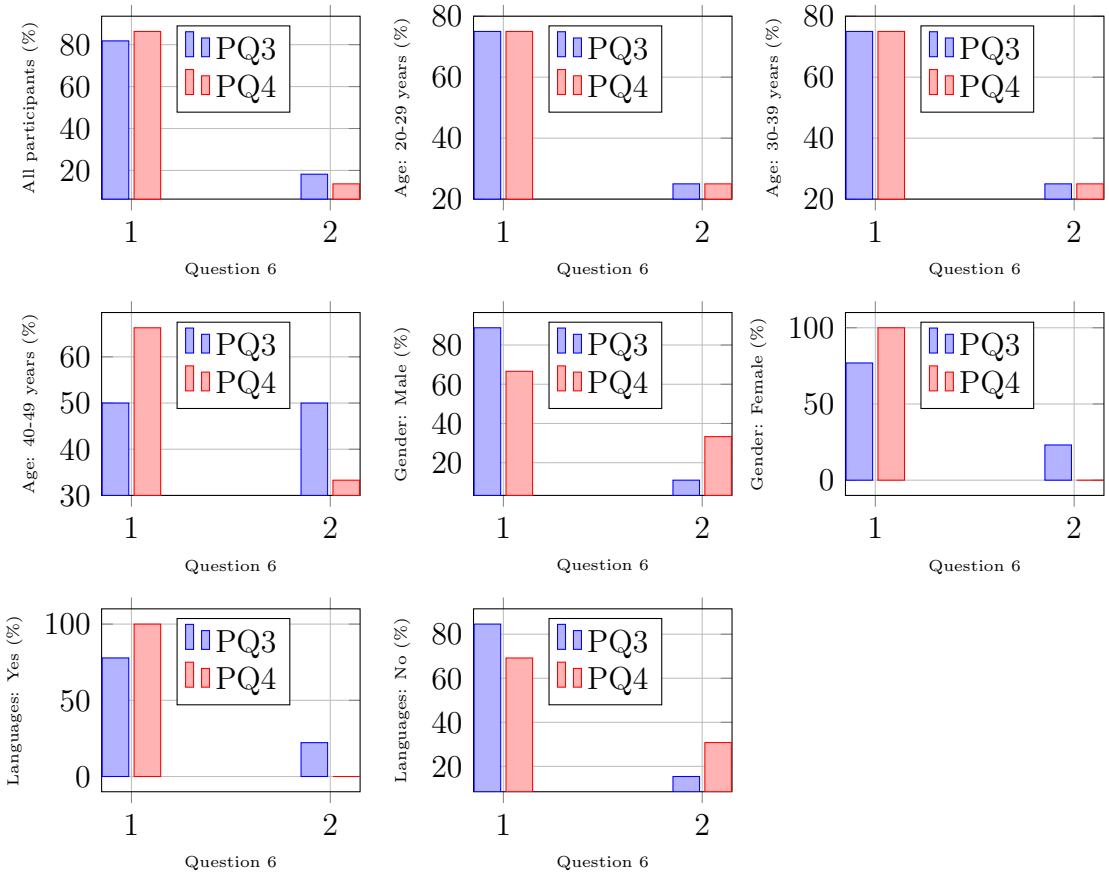


Figure 5.12: Results for Question 6

81.8% of the participants chose the first option, while the remaining participants opted for the last option: "I cannot answer as I did not manage to add PQ3 and PQ4." Figure 5.14 shows the results. In summary, the analysis of responses to Question 8 provides insights that majority of participants were able to successfully use the *PrivEx* interface to identify privacy risks.

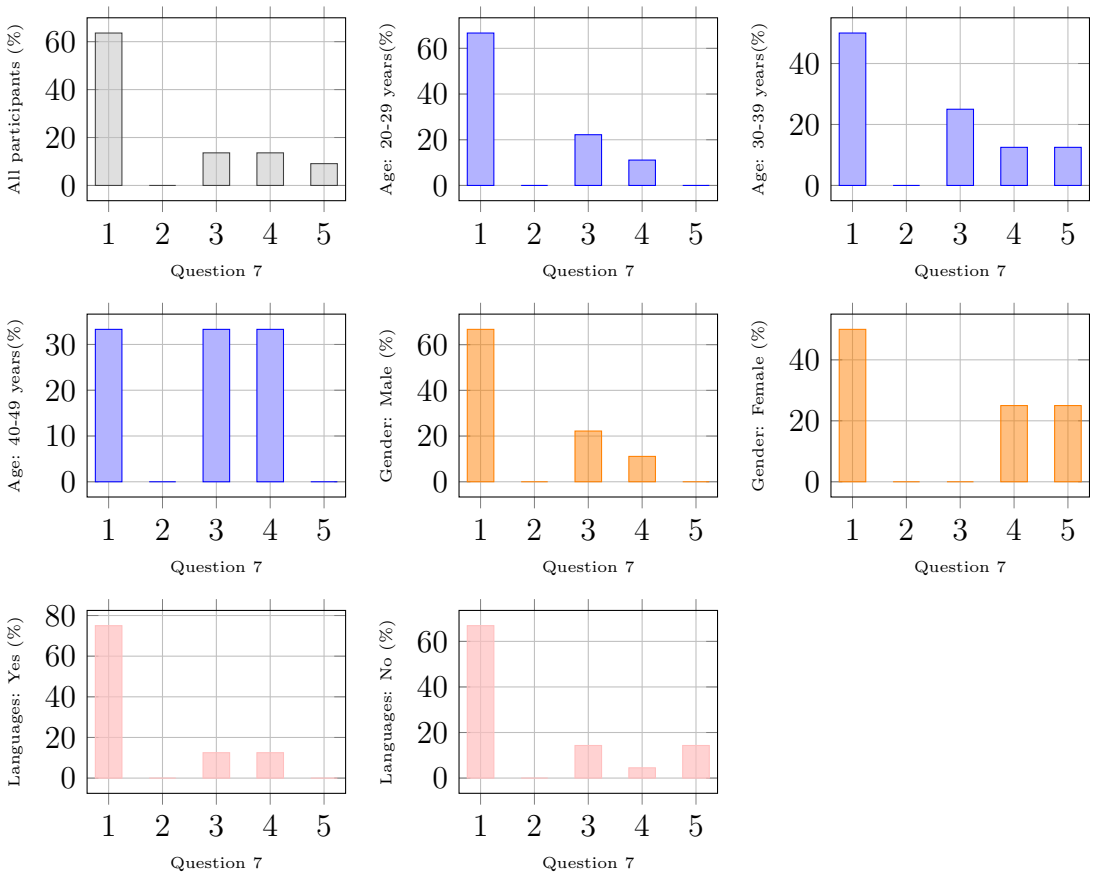


Figure 5.13: Results for Question 7

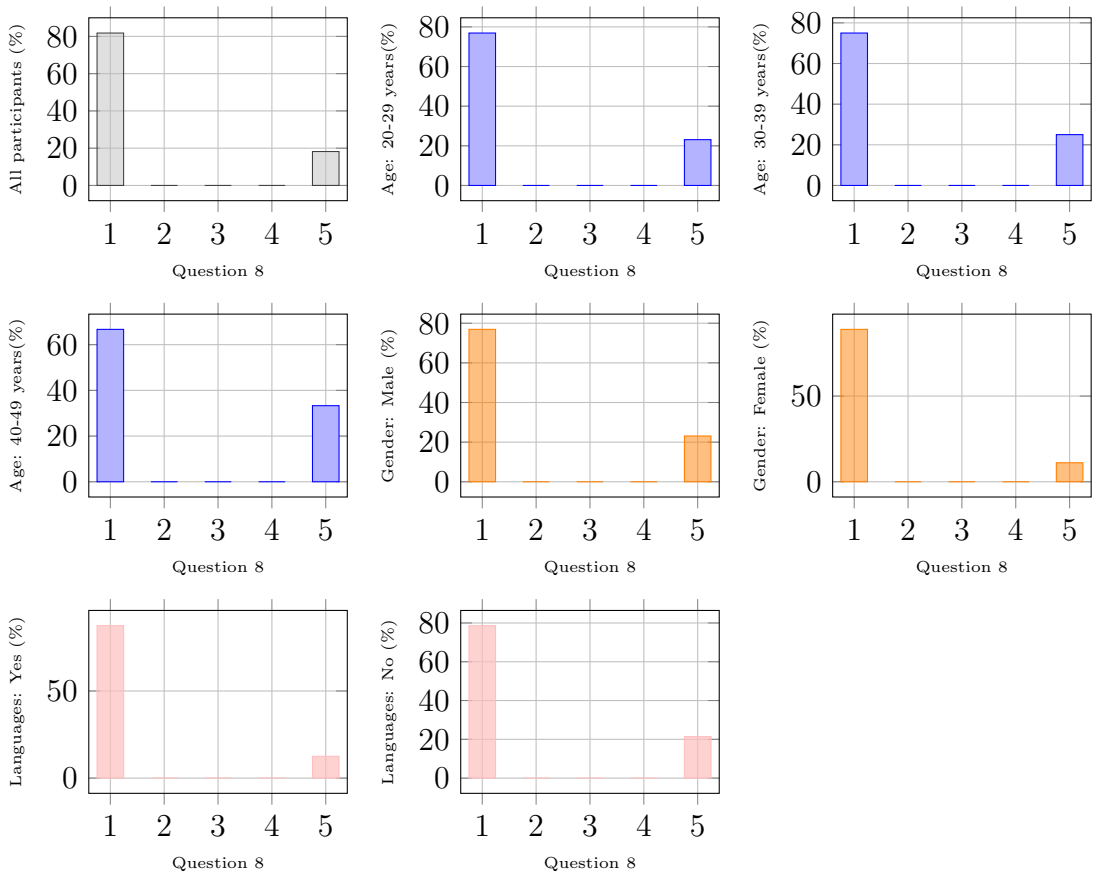


Figure 5.14: Results for Question 8

Chapter 6

Conclusion and perspectives

This chapter is organized as follows. In Section 6.1, we conclude our thesis. In Section 6.2, we discuss some possible research perspectives.

6.1 Conclusion

As the number of different service providers that can access and process the personal data stored on data servers increases, the risk of leakage of individual's personal data also increases. To protect the privacy of individual's personal data, we have proposed a data-independent approach that addresses the privacy versus utility dilemma. This approach allows data producers to keep control over the protection of their data in several real-world situations where personal data are collected by mobile devices or smart environments.

In Chapter 1, we provided an overview of the context in which our approach can be applied. We proposed a setting where the data producers keep their data on decentralized personal data servers and only disclose data to data consumers over secure communication links according to their privacy policies (a set of privacy queries). Data consumers specify the data needs in the form of utility policies (a set of utility queries) and explain for which task or service they are requesting the data from data producers. In our approach, we have considered that data producers and data consumers understand each other through a common vocabulary using the same ontology.

In Chapter 2, we summarized the existing privacy preserving methods for RDF

and Linked Data as well as within the context of IoT. We outlined the limitations of existing methods and discussed how our approach is different from existing methods. We have concluded that, unlike existing methods that alter personal data through generalization or noise addition, our data-independent framework serves as a proactive measure against potential privacy breaches. Furthermore, in contrast to existing query-based logical frameworks for RDF data, our framework handles complex queries with aggregates and is suitable for the privacy of temporal RDF data.

In Chapter 3, we introduced the main definitions and standards on which our thesis is based. We summarized the RDF and RDFS standards for describing data and ontologies on the Semantic Web and introduced the temporal extension of RDF and RDFS to capture temporal data and dynamic properties. We formally defined temporal aggregated conjunctive query (TACQ) with a SPARQL-like syntax extended with a time window definition for capturing aggregate on time and introduced the semantics for the evaluation of a $TACQ$ over temporal RDF graphs. Apart from $TACQ$ in its general form, we also considered simpler forms of $TACQs$ in our approach that are without aggregate terms named as conjunctive queries and without time window definitions named as aggregated conjunctive queries. Additionally, we demonstrated the use of existing technologies such as RDF, RDF-star, SPARQL and SPARQL-star for the implementation of temporal RDF graphs and temporal aggregated conjunctive queries.

In Chapter 4, we provided a query-based specification of privacy and utility policies and a formal definition of privacy risk. We provided the characterization of privacy risks by distinguishing the cases when a privacy query is a conjunctive query or an aggregated conjunctive query or a temporal aggregated conjunctive query. In our data-independent framework, the characterization of privacy risks is done by evaluating the query expressions only. We illustrated the characterization of privacy risks with the help of examples by considering a smart meter scenario inspired by a real-world use-case and built all the queries using the same ontology that we designed from the dataset provided by the *Irish Social Science Data Archive (ISSDA) Commission for Energy Regulation (CER)* [1]. This dataset includes time series of electrical consumptions of different house owners. In addition, pseudonymized metadata are available on house owners' demographics, home sizes and equipment associated to the electric consumption time series. Moreover, based on the theorems and their proofs, we designed and implemented several

[1] `https://www.ucd.ie/issda/data/commissionforenergyregulationcer/`

algorithms for the detection of privacy risks raised by the utility queries of data consumers.

In Chapter 5, we extended our data-independent framework to provide explanations of detected privacy risks and some options for modifying utility queries to remove the detected privacy risks. Each privacy risk is explained using two different levels. The first level simply points out the privacy queries that are violated by some utility queries and the second level exhibits the corresponding privacy risk by providing an example in the form of synthetic data built from the ontology and the utility and privacy queries involved in privacy risk. We also developed an interactive user-friendly interface that helps data producers in understanding and removing the privacy risks raised by utility queries. The interactive interface also provides a form-based interface that facilitates the data producers in the construction of privacy queries by taking guidance from the ontology. We also discussed the methodology and results of a user study that was conducted to evaluate the practical usability and effectiveness of the user interface, in which the participants were exposed to a smart meter scenario inspired by a real-world use case. The results of the user study showed that participants found it easier to construct privacy queries using the form-based interface and they also found the explanation helpful in understanding the detected privacy risks.

6.2 Perspectives

There are several directions in which our research can be extended, including the following:

Generalizing the case for characterizing the privacy risk for a temporal aggregated conjunctive privacy query: In Section 4.3.3, we presented different cases for characterizing the privacy risk for a temporal aggregated conjunctive privacy query Q_p and in Section 4.3.3.3, we provided the characterization of the privacy risk for Q_p by evaluating it against two sets, where each set contains only one temporal aggregated conjunctive utility query that computes the same aggregate as Q_p but on different time window definitions. Examining the characterization of the privacy risk for Q_p when there are more than two sets is one possible direction for future work. The objective is to formalize a generalized theorem capable of characterizing the privacy risk for Q_p by evaluating it against several sets, where each set contains only one temporal aggregated conjunctive utility query that computes the same aggregate as Q_p but on different time window

definitions.

Optimizing scalability and efficiency of data-independent framework: In this thesis, for characterizing the privacy risk, each privacy query is evaluated against a set of limited utility queries (maximum 10 utility queries) and we only considered a small ontology extracted from the ISSDA data set. Our data-independent framework could be optimized for scalability and efficiency in future work, particularly for handling a large number of utility queries and large ontologies (derived from big datasets).

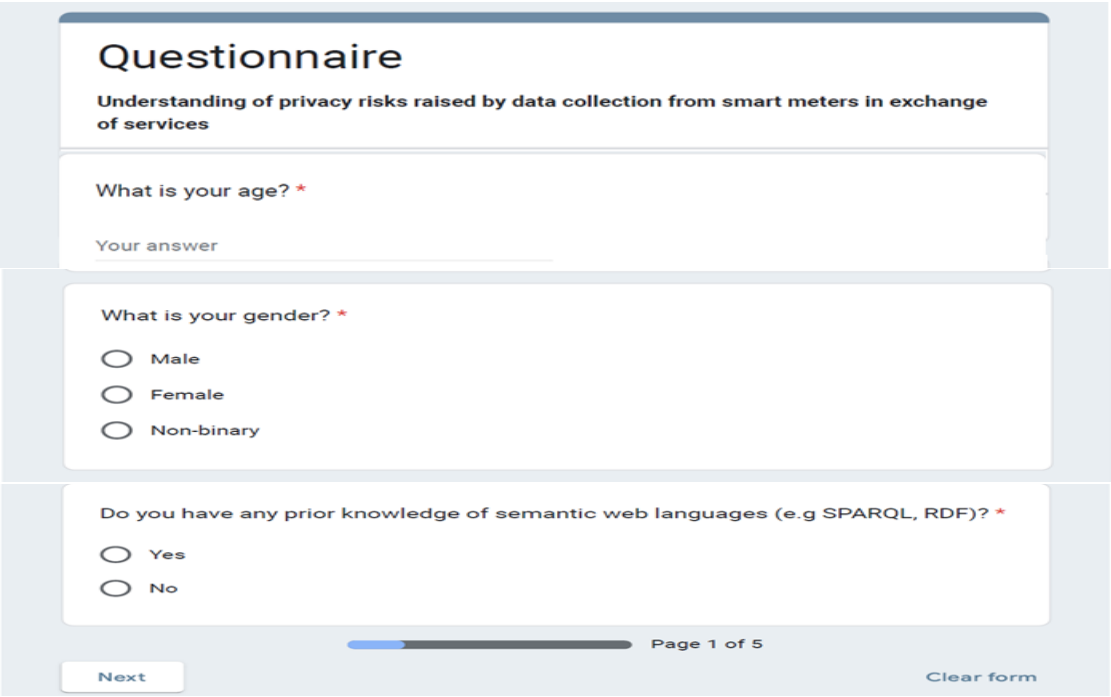
Integrating with language processing method: In our approach, all privacy and utility queries are expressed in the form of query language syntax and it can be daunting for a user to interpret and understand them who is not familiar with query language syntax. As a future direction, integrating language processing method for automatically generating queries in a query language syntax when they are expressed in natural language can be explored. This extension will enhance the practicality and usability of our framework.

Measuring the probability of privacy risks detected by the complex combination of several utility queries: In our approach, all the privacy breaches are detected in a systematic way but without distinguishing the risk of their occurrence. For instance, a privacy breach caused by the complex combination of several utility queries corresponds to an attack that is less probable to occur than an attack based on a single utility query. A research direction could be to measure the probability of the privacy risks that are detected.

Appendix A

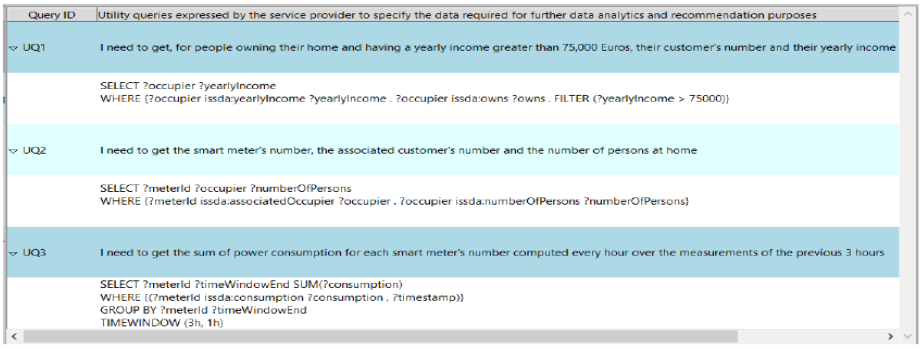
Online Questionnaire

This appendix presents the online questionnaire that was created for the user study evaluating the *PrivEx* interface. Other details and results of this questionnaire are presented in Section 5.4.



Scenario

Suppose you are a customer with smart meters installed at your home and all the data of your smart meters as well as your personal data is stored on your local data storage. Assume that a service provider offers you as a service to compute some useful statistics on the power consumption of homes that are similar to yours and to make you recommendations for energy saving products. For providing such services to their customers, the service provider expresses their data needs in the form of utility queries as illustrated in the screenshot below. All utility queries are expressed in query language syntax as well as in words.



Query ID	Utility queries expressed by the service provider to specify the data required for further data analytics and recommendation purposes
UQ1	I need to get, for people owning their home and having a yearly income greater than 75,000 Euros, their customer's number and their yearly income
	SELECT ?occupier ?yearlyIncome WHERE {?occupier issda:yearlyIncome ?yearlyIncome . ?occupier issda:owns ?owns . FILTER (?yearlyIncome > 75000)}
UQ2	I need to get the smart meter's number, the associated customer's number and the number of persons at home
	SELECT ?meterId ?occupier ?numberOfPersons WHERE {?meterId issda:associatedOccupier ?occupier . ?occupier issda:numberOfPersons ?numberOfPersons}
UQ3	I need to get the sum of power consumption for each smart meter's number computed every hour over the measurements of the previous 3 hours
	SELECT ?meterId ?timeWindowEnd SUM(?consumption) WHERE {{?meterId issda:consumption ?consumption . ?timestamp}} GROUP BY ?meterId ?timeWindowEnd TIMEWINDOW (3h, 1h)

How realistic do you find this scenario? *

- ◯ Completely realistic
- ◯ Somewhat realistic
- ◯ Neutral
- ◯ Somewhat unrealistic
- ◯ Completely unrealistic

To what degree do you understand these utility queries? *

- ◯ I completely understand these three utility queries
- ◯ I completely understand one or two of these utility queries
- ◯ I partially understand these three utility queries
- ◯ I partially understand one or two of these utility queries
- ◯ I do not understand any of these utility queries

Page 2 of 5

Back Next Clear form

Specification of your sensitive data

The following privacy queries (that are kept secret) specify the data that you do not want to be deduced by service providers if you accept to evaluate some of their utility queries on your local data.

Query ID	Specification of your sensitive data: No answer to following privacy queries should be deduced
PQ1	I do not want someone to be able to deduce my yearly income from my smart meter's number
	SELECT ?MeterId ?yearlyIncome WHERE {?MeterId issda:associatedOccupier ?Occupier . ?Occupier issda:yearlyIncome ?yearlyIncome}
PQ2	I do not want someone to be able to deduce the sum of my power consumption computed over intervals of 6 hours
	SELECT ?timeWindowEnd SUM(?consumption) WHERE {(?MeterId issda:consumption ?consumption, ?timestamp)} GROUP BY ?timeWindowEnd TIMEWINDOW (6h, 6h)

To what degree do you understand these privacy queries? *

- ◯ I completely understand both privacy queries
- ◯ I completely understand one of these privacy queries
- ◯ I partially understand both privacy queries
- ◯ I partially understand one of these privacy queries
- ◯ I do not understand any of these privacy queries

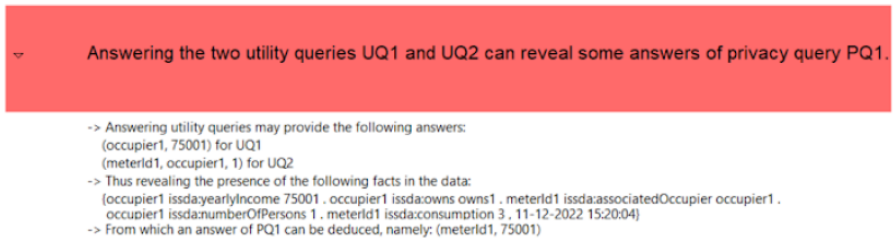
Page 3 of 5

Back Next Clear form

Explanation of the detected privacy risks

Privacy risks are detected if it is possible to deduce an answer to a privacy query from the answers of utility queries.

The following screenshot shows the explanation of one privacy risk raised by the privacy query PQ1.

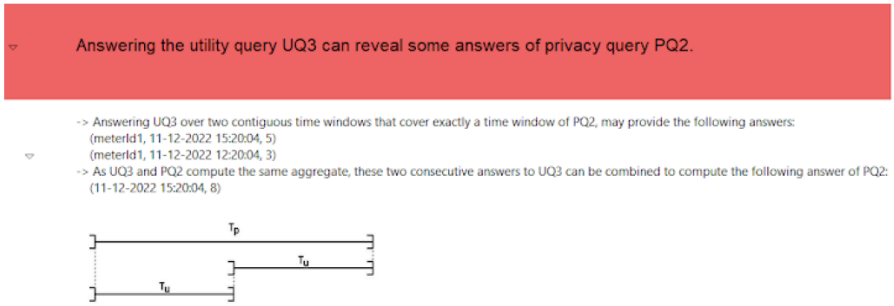


PQ1: *"I do not want someone to be able to deduce my yearly income from my smart meter's number"*
UQ1: *"I need to get, for people owning their home and having a yearly income greater than 75,000 Euros, their customer's number and their yearly income"*
UQ2: *"I need to get the smart meter's number, the associated customer's number and the number of persons at home"*

How do you find the explanation of this privacy risk raised by the privacy query PQ1? *

- ○ Completely helpful
- ○ Very helpful
- ○ Somewhat helpful
- ○ Little helpful
- ○ Not at all helpful

The following screenshot shows the explanation of one privacy risk raised by the privacy query PQ2.



PQ2: *"I do not want someone to be able to deduce the sum of my power consumption computed over intervals of 6 hours"*

UQ3: *"I need to get the sum of power consumption for each smart meter's number computed every hour over the measurements of the previous 3 hours"*

How do you find the explanation of this privacy risk raised by the privacy query PQ2? *

- ○ Completely helpful
- ○ Very helpful
- ○ Somewhat helpful
- ○ Little helpful
- ○ Not at all helpful

Page 4 of 5

Back Next Clear form

Building and testing your privacy queries

This part of questionnaire is not mandatory but your participation will be helpful for this user study. If you do not want to participate then you can proceed with submission of this questionnaire without responding to the remaining questions.

If you agree to take part then it will take around 15 minutes and you will have to build two privacy queries using a form-based interface. This form-based interface can be accessed by downloading the executable file using one of the links given below.

1. For Windows OS: https://cloud.univ-grenoble-alpes.fr/s/9j9fnoRNDgJ2Jd3

2. For Linux OS: https://cloud.univ-grenoble-alpes.fr/s/nQZ2m7go5ko29Ta

3. For Mac OS: https://cloud.univ-grenoble-alpes.fr/s/esJHiriQ7EHERwx

Instructions for executing the downloaded file on each OS are given below. Follow the instructions specific to your OS to open the interface titled as "Detecting and Explaining Privacy Risks".

1. For Windows OS: Double click the downloaded file. Your OS may ask to approve the execution of this unsigned application. To approve the execution click "More info" and then click "Run Anyway".

2. For Linux OS: Right-click (or control-click) the downloaded file, then choose "Properties" from the shortcut menu, then click the "Permissions" tab, then select the check-box "Allow executing file as a program" and then double click the downloaded file.

3. For Mac OS: Right-click (or control-click) the downloaded app, then choose "Open" from the shortcut menu and then click the "Open" button at the next dialog warning.

Click the "Add Privacy Query" button to open the form-based interface for building queries. This form-based interface will help you to build each part of the privacy query progressively by using dynamically constructed drop-downs, buttons, check-boxes and entry boxes. The use of interface components is briefly described in the following steps followed to build and add the privacy query.

Step 1: Enter your privacy query in words.

Step 2: Select the necessary properties.

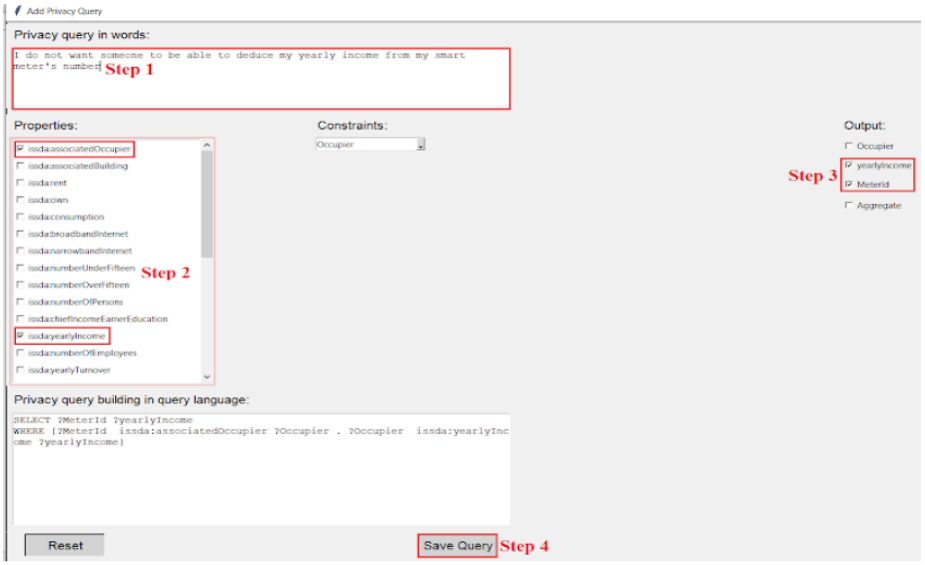
Step 3: You can add some constraints on the selected properties.

Step 4: Select the results of your privacy query that you want to keep secret. These results can also include an aggregated values that can be computed by time window intervals.

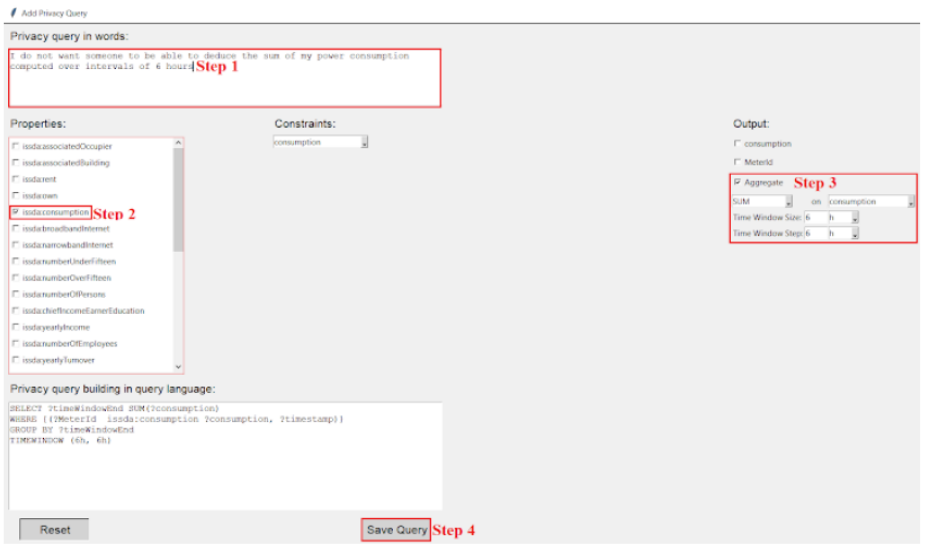
Step 5: Click the "Save Query" button to add your produced privacy query to the list of privacy queries.

If you are unable to see all the mentioned components on the window screen of your computer then please scroll down, scroll up, scroll right or scroll left by clicking at the scrollbar icons (small black triangles) and use the desired components.

The screenshot below presents the steps followed for adding the privacy query PQ1.



The screenshot below presents the steps followed for adding the privacy query PQ2.



The objective is to build the following privacy query using the form-based interface.
PQ3: "I *do not want someone to be able to deduce the number of persons at my home"*

Copy the produced privacy query in query language syntax and paste it below.

Your answer

If you have not clicked the "Save Query" button before then please click it to add the produced privacy query PQ3 to the list of privacy queries.

The objective is to build the following privacy query using the form-based interface.
PQ4: *"I do not want someone to be able to deduce the aggregated maximum reading of my power consumption computed over intervals of 3 hours"*

Copy the produced privacy query in query language syntax and paste it below.

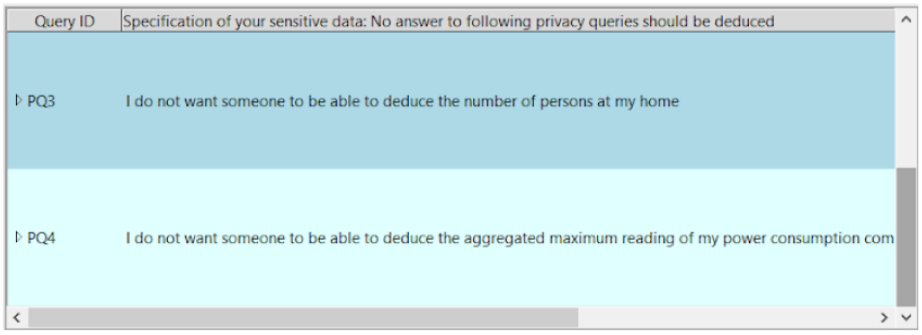
Your answer

If you have not clicked the "Save Query" button before then please click it to add the produced privacy query PQ4 to the list of privacy queries.

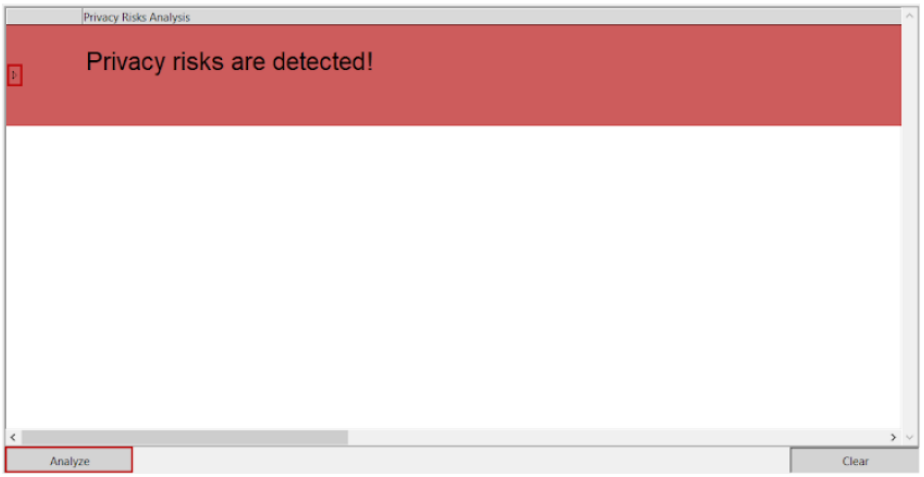
How difficult did you find to build these two privacy queries?

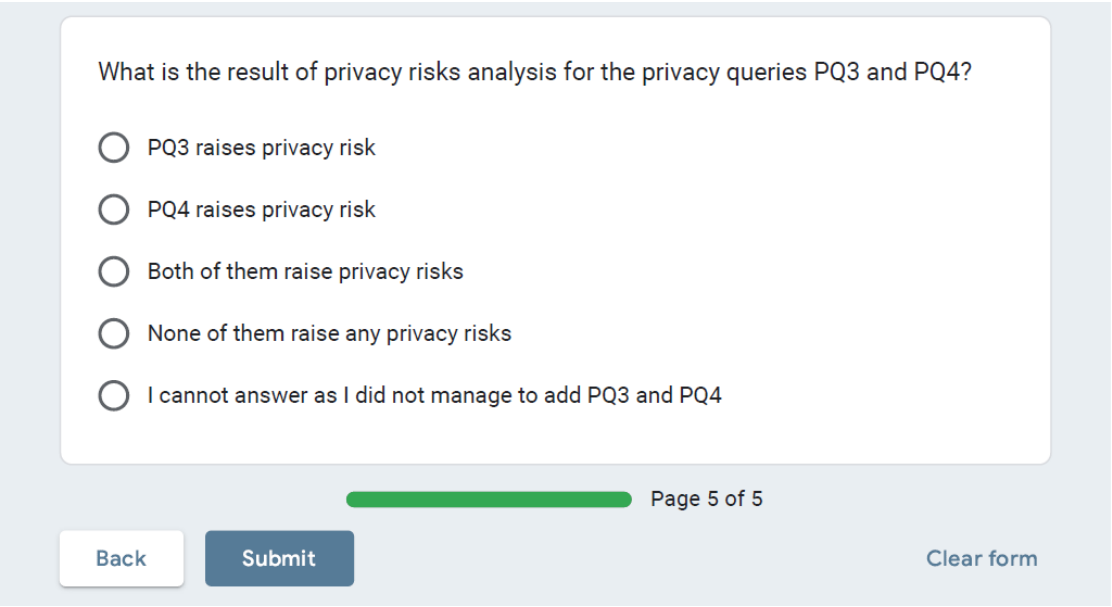
- ◯ I found it easy
- ◯ Difficult to build PQ3
- ◯ Difficult to build PQ4
- ◯ Difficult to build both PQ3 and PQ4
- ◯ I did not manage to build any of these two queries

New privacy queries PQ3 and PQ4 will appear in the list of privacy queries as shown in the screenshot below.



Click the "Analyze" button to detect privacy risks raised by the privacy queries PQ3 and/or PQ4. Then click the small triangle icon (highlighted with red color in the screenshot below) to see the detected privacy risks and their explanation.




What is the result of privacy risks analysis for the privacy queries PQ3 and PQ4?
PQ3 raises privacy risk
PQ4 raises privacy risk
Both of them raise privacy risks
None of them raise any privacy risks
I cannot answer as I did not manage to add PQ3 and PQ4
Page 5 of 5
Back
Submit
Clear form

Bibliography

[RAD78a] R.L. Rivest, L. Adleman, and M.L. Deaouzos. "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems". In: New York: ACM, 1978, pp. 169–179.

[RAD78b] R.L. Rivest, L. Adleman, and M.L. Deaouzos. "On data banks and privacy homomorphisms". In: New York: Foundations of secure computation, 1978, pp. 169–180.

[Yao82] Andrew C. Yao. "Protocols for Secure Computations". In: *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science*. SFCS '82. USA: IEEE Computer Society, 1982, pp. 160–164.

[Sam01] P. Samarati. "Protecting Respondents' Identities in Microdata Release". In: *IEEE Trans. on Knowl. and Data Eng.* 13.6 (2001), pp. 1010–1027. ISSN: 1041-4347. DOI: 10.1109/69.971193. URL: https://doi.org/10.1109/69.971193.

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

[Swe02] Latanya Sweeney. "K-Anonymity: A Model for Protecting Privacy". In: *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.* 10.5 (2002), pp. 557–570. ISSN: 0218-4885. DOI: 10.1142/S0218488502001648. URL: https://doi.org/10.1142/S0218488502001648.

[TPC03] David Trastour, Chris Preist, and Derek Coleman. "Using Semantic Web Technology to Enhance Current Business-to-Business Integration Approaches". In: Brisbane, Australia: Seventh IEEE International Enterprise Distributed Object Computing Conference, 2003.

[Pis04] Domenico Pisanelli. *Ontologies in Medicine*. Vol. 102. IOS Press, 2004. ISBN: 978-1-60750-945-5.

[Rec04a] W3C Recommendation. *RDF Semantics*. 2004. URL: https://www.w3.org/TR/rdf-mt/ (visited on 04/30/2023).

[Rec04b] W3C Recommendation. *RDF Semantics*. 2004. URL: https://www.w3.org/TR/rdf-mt/#RDFSRules (visited on 04/30/2023).

[Coh05] Sara Cohen. "Containment of aggregate queries". In: *ACM SIGMOD Record* 34.1 (2005), pp. 77–85.

[Gie05] Mark Giereth. "On Partial Encryption of RDF-Graphs". In: vol. 3729. Nov. 2005, pp. 308–322. ISBN: 978-3-540-29754-3. DOI: 10.1007/11574620_24.

[RFJ05] Pavan Reddivari, Timothy W. Finin, and Anupam Joshi. "Policy-Based Access Control for an RDF Store". In: *International Joint Conference on Artificial Intelligence*. 2005.

[Dwo06] Cynthia Dwork. "Differential Privacy". In: *Proceedings of the 33rd International Conference on Automata, Languages and Programming - Volume Part II*. ICALP'06. Venice, Italy: Springer-Verlag, 2006, pp. 1–12. ISBN: 3540359079. DOI: 10.1007/11787006_1. URL: https://doi.org/10.1007/11787006_1.

[JF06] Amit Jain and Csilla Farkas. "Secure Resource Description Framework: An Access Control Model". In: *Proceedings of the Eleventh ACM Symposium on Access Control Models and Technologies*. SACMAT '06. Lake Tahoe, California, USA: Association for Computing Machinery, 2006, pp. 121–129. ISBN: 1595933530. DOI: 10.1145/1133058.1133076. URL: https://doi.org/10.1145/1133058.1133076.

[Mac+06] A. Machanavajjhala et al. "L-diversity: privacy beyond k-anonymity". In: *22nd International Conference on Data Engineering (ICDE'06)*. 2006, pp. 24–24. DOI: 10.1109/ICDE.2006.1.

[Abe+07] Fabian Abel et al. "Enabling advanced and context-dependent access control in RDF stores". In: *International Semantic Web Conference*. Springer. 2007, pp. 1–14.

[LLV07] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. "t-Closeness: Privacy Beyond k-Anonymity and l-Diversity". In: *2007 IEEE 23rd International Conference on Data Engineering*. 2007, pp. 106–115. DOI: 10.1109/ICDE.2007.367856.

[Ger08] Sabrina Gerbracht. "Possibilities to Encrypt an RDF-Graph". In: (Apr. 2008). DOI: 10.1109/ICTTA.2008.4530288.

[Hep08] Martin Hepp. "Goodrelations: an Ontology for Describing Products and Services Offers on the Web". In: Acitrezza, Italy: Sixteenth Inter-

national Conference on Knowledge Engineering and Knowledge Management, Springer, 2008.

[KSW08] Jonathan Katz, Amit Sahai, and Brent Waters. "Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products". In: *Advances in Cryptology – EUROCRYPT 2008*. Ed. by Nigel Smart. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 146–162. ISBN: 978-3-540-78967-3.

[CLS09] Hilary Cheng, Yi-Chuan Lu, and Calvin Sheu. "An Ontology-based Business Intelligence Application in a Financial Knowledge Management System". In: *Expert Systems with Applications* 36.2 (2009), pp. 3614–22.

[Rec09] W3C Recommendation. *Feature:AggregateFunctions*. 2009. URL: `https://www.w3.org/2009/sparql/wiki/Feature:AggregateFunctions#Feature_description` (visited on 04/30/2023).

[TSH09] Ioan Toma, Elena Simperl, and Graham Hench. "A Joint Roadmap for Semantic Technologies and the Internet of Things". In: Crete, Greece: Proceedings of the Third STI Roadmapping Workshop, 2009.

[All+10] Tristan Allard et al. "Secure personal data servers: a vision paper". In: *Proceedings of the VLDB Endowment* 3.1-2 (2010), pp. 25–35.

[CGa10] Oscar Corcho, Raul Garcia-Castro, and et al. "Five Challenges for the Semantic Sensor Web". In: *Semantic Web* 1 (2010), pp. 121–125.

[Dij+10] Marten van Dijk et al. "Fully Homomorphic Encryption over the Integers". In: *Advances in Cryptology – EUROCRYPT 2010*. Ed. by Henri Gilbert. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 24–43. ISBN: 978-3-642-13190-5.

[Flo+10] Giorgos Flouris et al. "Controlling Access to RDF Graphs". In: *Future Internet - FIS 2010*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 107–117. ISBN: 978-3-642-15877-3.

[FWa10] Benjamin C. M. Fung, Ke Wang, and et al. "Privacy-Preserving Data Publishing: A survey of Recent Developments". In: *ACM Computing Surveys* 42(4) (2010).

[BWa12] Payam Barnaghi, Wei Wang, and et al. "Semantics for the Internet of Things: Early Progress and Back to the Future". In: *International journal on Semantic Web and information systems* 8.1 (2012), pp. 1–21.

[Lop+12] Nuno Lopes et al. "A Logic Programming approach for Access Control over RDF". In: *International Conference on Logic Programming*. 2012.

[Pap+12] Vassilis Papakonstantinou et al. “Access Control for RDF Graphs Using Abstract Models”. In: *Proceedings of the 17th ACM Symposium on Access Control Models and Technologies*. SACMAT ’12. Newark, New Jersey, USA: Association for Computing Machinery, 2012, pp. 103–112. ISBN: 9781450312950. DOI: 10.1145/2295136.2295155. URL: https://doi.org/10.1145/2295136.2295155.

[Aro13] Yotam Aron. *Information Privacy for Linked Data*. 2013.

[IMa13] Seiji Isotani, Riichiro Mizoguchic, and et al. “A Semantic Web-based authoring tool to Facilitate the Planning of Collaborative Learning Scenarios Compliant with Learning Theories”. In: *Computers and Education* 63 (2013), pp. 267–284.

[KS13] Andreas Kasten and Ansgar Scherp. “Towards Search on Encrypted Graph Data”. In: Jan. 2013.

[Rec13] W3C Recommendation. *SPARQL 1.1 Query Language*. 2013. URL: https://www.w3.org/TR/sparql11-query/ (visited on 04/30/2023).

[RKa14] Jyothsna Rachapalli, Vaibhav Khadilkar, and et al. “Towards Fine Grained RDF Access Control”. In: London, Ontario, Canada: SACMAT ’14: Proceedings of the 19th ACM symposium on Access Control Models and Technologies, 2014, pp. 165–176.

[Rec14a] W3C Recommendation. *RDF 1.1 Concepts and Abstract Syntax*. 2014. URL: https://www.w3.org/TR/rdf11-concepts/ (visited on 04/30/2023).

[Rec14b] W3C Recommendation. *RDF 1.1 Semantics*. 2014. URL: https://www.w3.org/TR/rdf11-mt/#reification (visited on 04/30/2023).

[Rec14c] W3C Recommendation. *RDF Schema 1.1*. 2014. URL: https://www.w3.org/TR/rdf-schema/ (visited on 04/30/2023).

[RGP15] Filip Radulovic, Raul Garcia-Castro, and Asuncion Gomez Perez. “Towards the Anonymisation of RDF Data”. In: Pittsburg, Philadelphia, Estados Unidos: 27th International Conference on Software Engineering and Knowledge Engineering, 2015.

[Say+15] Tarek Sayah et al. “Inference Leakage Detection for Authorization Policies over RDF Data”. In: *Data and Applications Security and Privacy XXIX*. Ed. by Pierangela Samarati. Cham: Springer International Publishing, 2015, pp. 346–361. ISBN: 978-3-319-20810-7.

[SS15] Ram D Siram and Amit Seth. “Internet of Things Perspectives”. In: *IT Professional* 17.3 (2015), pp. 60–63.

[GK16] Bernardo Cuenca Grau and Egor V. Kostylev. “Logical foundations of Privacy-Preserving Publishing of Linked Data”. In: AAAI Press, 2016.

[SW16] Ioan Szilagyi and Patrice Wira. “Ontologies and Semantic Web for the Internet of Things - a survey”. In: Florence, Italy: 42nd Annual Conference of the IEEE Industrial Electronics Society, 2016.

[BRa17] Rachelle Bosua, Megan Richardson, and et al. *Privacy in a world of the Internet of Things A Legal and Regulatory Perspective*. 2017.

[Dwo+17] Cynthia Dwork et al. “Exposed! A Survey of Attacks on Private Data”. In: *Annual Review of Statistics and Its Application* 4 (Mar. 2017). DOI: 10.1146/annurev-statistics-060116-054123.

[Fer+17] Javier Fernandez et al. “Self-Enforcing Access Control for Encrypted RDF”. In: May 2017, pp. 607–622. ISBN: 978-3-319-58067-8. DOI: 10.1007/978-3-319-58068-5_37.

[GP17] Maria Ganzha and Marcin Paprzycki. “Semantic interoperability in the Internet of Things: An overview from the INTER-IoT perspectives”. In: *Journal of Network and Computer Applications* 81 (2017), pp. 111–124.

[GPa17] Amelie Gyrard, Pankesh Patel, and et al. “Semantic Web meets Internet of Things (IoT) and Web of Things (WoT)”. In: Perth, Australia: 26th International World Wide Web Conference, 2017.

[HHD17] Benjamin Heitmann, Felix Hermsen, and Stefan Decker. “k- RDF-neighbourhood anonymity: Combining Structural and Attribute-based Anonymisation for Linked Data”. In: Achen: In 5th Workshop on Society, Privacy, the Semantic Web–Policy, and Technology, 2017.

[KMD17] Sabrina Kirrane, Alessandra Mileo, and Stefan Decker. “Access control and the Resource Description Framework: A survey”. In: *Semantic Web* 8.2 (2017), pp. 311–352.

[SLa17] Roney Reis C. Silva, Bruno C. Leal, and et al. “A Differentially Private Approach for Querying RDF Data of Social Networks”. In: *In Proceedings of the 21st International Database Engineering and Applications Symposium* (2017), pp. 74–81.

[DBa18] Remy Delanaux, Angela Bonifati, and et al. “Query-based Linked Data Anonymization”. In: Monterey, United States: The 17th International Semantic Web Conference, 2018, pp. 530–546.

[DC19] Irvin Dongo and Richard Chbeir. "RiAiR: A Framework for Sensitive RDF Protection". In: *Journal of Web Engineering* 18(1) (2019), pp. 43–96.

[GK19] Bernardo Cuenca Grau and Egor V. Kostylev. "Logical Foundations of Linked Data Anonymisation". In: *Journal of Artificial Intelligence Research* 64 (2019), pp. 253–314.

[Sal19] Julián Salas. "Sanitizing and measuring privacy of large sparse datasets for recommender systems". In: *Journal of Ambient Intelligence and Humanized Computing* (July 2019). DOI: 10.1007/s12652-019-01391-2.

[Bur20] Maxime Buron. "Efficient reasoning on large and heterogeneous graphs". Theses. École Polytechnique, Oct. 2020. URL: https://hal.inria.fr/tel-03107689.

[Ver+20] Ruben Verborgh et al. "HDT Crypt: Compression and Encryption of RDF Datasets". In: *Semant. Web* 11.2 (2020), pp. 337–359. ISSN: 1570-0844. DOI: 10.3233/SW-180335. URL: https://doi.org/10.3233/SW-180335.

[ABR21] Hira Asghar, Christophe Bobineau, and Marie-Christine Rousset. "Compatibility checking between privacy and utility policies : a query - based approach." In: *37th Conference on Data Management- Principles Technologies and Applications (BDA)*. 2021.

[DA21] Iman Dakhil Idan Saeedi and Ali Al-Qurabat. "A Systematic Review of Data Aggregation Techniques in Wireless Sensor Networks". In: *Journal of Physics: Conference Series* 1818 (Mar. 2021), p. 012194. DOI: 10.1088/1742-6596/1818/1/012194.

[YKD21] Shamim Yousefi, Hadis Karimipour, and Farnaz Derakhshan. "Data Aggregation Mechanisms on the Internet of Things: A Systematic Literature Review". In: *Internet of Things* 15 (2021), p. 100427. ISSN: 2542-6605. DOI: https://doi.org/10.1016/j.iot.2021.100427. URL: https://www.sciencedirect.com/science/article/pii/S2542660521000718.

[Arn+22] Dörthe Arndt et al. *RDF-star and SPARQL-star*. Draft Community Group Report, 2022. URL: https://w3c.github.io/rdf-star/cg-spec/editors_draft.html.

[ABR22] Hira Asghar, Christophe Bobineau, and Marie-Christine Rousset. "Identifying Privacy Risks Raised By Utility Queries". In: *Web Information Systems Engineering – WISE 2022: 23rd International Conference,*

November 1–3, 2022, Proceedings. Biarritz, France: Springer-Verlag, 2022, pp. 309–324. ISBN: 978-3-031-20890-4. URL: https://doi.org/10.1007/978-3-031-20891-1_22.

[GS22] H. Goyal and S. Saha. "Multi-Party Computation in IoT for Privacy-Preservation". In: *2022 IEEE 42nd International Conference on Distributed Computing Systems (ICDCS)*. Los Alamitos, CA, USA: IEEE Computer Society, 2022, pp. 1280–1281. DOI: 10.1109/ICDCS54860.2022.00133. URL: https://doi.ieeecomputersociety.org/10.1109/ICDCS54860.2022.00133.

[Mar+22] Chiara Marcolla et al. "Survey on Fully Homomorphic Encryption, Theory, and Applications". In: *Proceedings of the IEEE* 110.10 (2022), pp. 1572–1609. DOI: 10.1109/JPROC.2022.3205665.

[ABR23] Hira Asghar, Christophe Bobineau, and Marie-Christine Rousset. "Explanation based Tool for Helping Data Producers to Reduce Privacy Risks". In: *20th Extended Semantic Web Conference (ESWC), May 28-June 1, 2023, Proceedings*. Hersonissos, Greece, 2023.

[Qas+23] Fahad Qaswar et al. "Applications of Ontology in the Internet of Things: A Systematic Analysis". In: *Electronics* 12.1 (2023). ISSN: 2079-9292. DOI: 10.3390/electronics12010111. URL: https://www.mdpi.com/2079-9292/12/1/111.

[Gie] Mark Giereth. In: ed. by Mark Giereth. URL: https://www.bibsonomy.org/bibtex/2805e8f3e4a1159ff18d2e65c25383d73/bergo.