Privacy-aware aggregation of IoT streams in multi-owner networks

Renato Caminha Juaçaba Neto

UNIVERSITÉ DE STRASBOURG

ÉCOLE DOCTORALE MATHÉMATIQUES, SCIENCES DE L'INFORMATION ET DE L'INGÉNIEUR

Laboratoire ICube – UMR 7357

THÈSE présentée par :

[Renato CAMINHA JUAÇABA NETO]

Défendu le : **12 Juillet 2022**

pour obtenir le grade de : **Docteur de l'Université de Strasbourg**

Discipline/ Spécialité : Informatique

Privacy-aware Aggregation of IoT Streams in Multi-Owner Networks

THÈSE dirigée par :
Dr. THÉOLEYRE Fabrice Chargé de recherche, CNRS, France

CO-ENCADRANT :
Dr. MÉRINDOL Pascal Maitre de conférences, Université de Strasbourg, France

RAPPORTEURS :
Prof. VALOIS Fabrice Professeur, INSA Lyon, France
Prof. GUÉRIN LASSOU Isabelle Professeur, Université Claude Bernard, France

---

AUTRES MEMBRES DU JURY :
Dr. CARNEIRO VIANA Aline Directrice de recherche, INRIA Saclay, France
Prof. GUITTON Alexandre Professeur, Université Clermont Auvergne, France
Dr. CHOLEZ Thibault Maitre de conférences, Université de Lorraine, France

Université de Strasbourg

École doctorale MSII

Privacy-aware Aggregation of IoT Streams in Multi-Owner Networks

THÈSE

pour obtenir le grade de

Doctorat de l'Université de Strasbourg

(mention informatique)

Defended on 12 July 2022

présentée par

Renato CAMINHA JUAÇABA NETO

Composition du jury

Directeur de thse:	Dr. Fabrice THÉOLEYRE, CNRS, France
Co-encadrant:	Dr. Pascal MÉRINDOL, Universit de Strasbourg, France
Rapporteurs:	Prof. Fabrice VALOIS, INSA Lyon, France Prof. Isabelle GUÉRIN LASSOUS, Universit Claude Bernard, Lyon, France
Examinateurs:	Dr. Aline CARNEIRO VIANA, INRIA Saclay, France Prof. Alexandre GUITTON, Universit Clermont Auvergne, France Dr. Thibault CHOLEZ, Universit de Lorraine, France

Laboratoire des sciences de l'ingnieur, de l'informatique et de l'imagerie
ICUBE — UMR 7357

Acknowledgments

And thus I reach this huge milestone after some of the best years of my life. The work I done and the people I met in these three years gave me incredible memories and changed my life forever. This thesis will stay as the main materialization of this great period.

I must begin by thanking Fabrice Théoleyre and Pascal Mérindol for making this work possible with all the mentoring and camaraderie that made this possible. I doubt I will ever find other directors as involved and dedicated as you. I must also thank Carina de Oliveira, the person who first saw my potential in academics, I never imagined that it would lead me this far.

I would like to also thank Prof. Isabelle Guerrin Lassous, Prof. Fabrice Valois, Dr. Aline Carneiro Viana, Prof. Alexandre Guitton, and Dr. Thibault Cholez for having accepted to be member of my PhD jury. Their constructive comments and remarks helped me to improve the quality of this manuscript.

Not to forget my colleagues from the Reseaux research group, professors and students. With a special thanks to the PhD students that made my adaptation to this new country so much easier. Rodrigo and Ana, a couple that I can always count on. Julian and Melanie, the couple always there for a good time. Andreas for the companionship and help with french problems. Odnan for all the crazy times (my broom was never the same). In addition to all other PhD students that made me feel welcome from the beginning: Amine, Jean Philippe, Sebastian, Farzad, Jean-Romain and Loic.

A big thanks goes to all the people that I met through the student association StrasAIR and the CDE. Life during the pandemic would have been so much harder without everyone. Special thanks for Michael Okafor, Iryna Markachuk and Tatjana Gerasimova for so many great times together. But let's also not forget to thank: Wissal for all the great food; Filipa and Guido for so many events; Diana for making sure that I work in the final months and to lose the stress weight; Rokas for always being there, Thiago and Kelton for being the best brazilians, Gercelia for being the greatest help; Davi and Bea for being the greatest friends; and Priscila and Didier for being the cuttest couple.

But leaving the most important for last, I owe everything to my family. The support, mentoring and love is and always will be the foundation to everything I do in my life. I hope I will be able to repay in kind and with interest to all that you have done for me.

Abstract

Current deployments of IoT systems for smart cities and smart grids have a high density of devices. Which lead networks to encounter issues as deployment space and network infrastructure are limited. To tackle this, networks open their boundaries and share their sensors and services to others, allowing the other networks to access their internal sensors and services. Data speed meters in a highway, for example, can be further used for incident response and route recommendation services.

The resulting IoT ecosystem is complex due to the multiple operators and data owners that are involved. These require their networks to operate according to their own objectives and for their data to be used according to their consent. But result in datasets with high diversity as it includes data from multiple owners.

The focus of this thesis are the privacy concerns of producers as data is disclosed to other networks. Which are a consequence of the different consent levels that data is produced with. Medical data for example has very limiting usage consent from patients while data measure at public areas is much less restricting.

These restrictions are mainly enforced by dedicated devices among networks to manage the data disclosed. They employ access control strategies to disclose information based on the characteristics of consumers, producers and data itself. They also employ anonymization algorithms to limit the data disclosed and diminish the effects of data being further disclosed.

In this thesis we investigate how to provide privacy-aware data collection among different systems while preserving the constraints of each data owner. We focus on the usage of aggregation, that is, functions that describe groups of data, *e.g.*, average and minimum. These functions leverage scalability by reducing the network load and privacy by replacing detailed datasets with information on groups of entries.

For this objective we provide three contributions. We first provide a study of privacy-aware multi system ecosystem where we formally define data exchanged among them, and then we study how such a system scales with state-of-the-art communication technologies. Second, we provide a content-centric architecture for efficient aggregation of data streams from remote networks. And third, we expand our architecture to handle scenarios where networks offer data indiscriminately to each other and must correctly aggregate data. Through these contributions we offer the means to stream aggregated data in a multi-owner IoT infrastructure while respecting the restraints of involved producers.

List of Publications

Journal paper under review

1. Renato Juaçaba Neto, Pascal Mérindol, and Fabrice Théoleyre (2022). "Enabling Privacy by Anonymization in the Collection of Similar Data in Multi-Domain IoT". In: **Computer Communication**. Under review.

Conference papers

2. Renato Juaçaba Neto, Pascal Mérindol, and Fabrice Théoleyre. "Data Aggregation for Privacy Protection of Data Streams Between Autonomous IoT Networks". In: ***Symposium on Computers and Communications (ISCC)***. IEEE, 2021, pp. 1–6. DOI: 10.1109/ISCC53001.2021.9631401.

3. Renato Juaçaba Neto, Pascal Mérindol, Antoine Gallais, and Fabrice Théoleyre. "Scalability of LPWAN for Smart City Applications". In: ***International Wireless Communications and Mobile Computing Conference (IWCMC)***. 2021, pp. 74–79. DOI: 10.1109/IWCMC51323.2021.9498698.

4. Renato Juaçaba Neto, Pascal Mérindol, and Fabrice Théoleyre. "Transformation Based Routing Overlay for Privacy and Reusability in Multi-Domain IoT". in: ***International Symposium on Network Computing and Applications (NCA)***. IEEE, 2020, pp. 1–8. DOI: 10.1109/NCA51143.2020.9306709.

5. Renato Juaçaba Neto, Pascal Mérindol, and Fabrice Théoleyre. "A Multi-Domain Framework to Enable Privacy for Aggregated IoT Streams". In: ***Conference on Local Computer Networks (LCN)***. IEEE. 2020, pp. 401–404. DOI: 10.1109/LCN48667.2020.9314825.

We live on an island surrounded by a sea of ignorance. As our island of knowledge grows, so does the shore of our ignorance. (John Archibald Wheeler)

Contents

Chapter 1

Introduction

Internet of Things (IoT) has become part of the life of many people in the past few years and there is still expectation for the market to grow even further [5]. These IoT systems permeate modern life, from house appliances to work environments and public space systems. Generally, IoT devices can manage and be imbued to systems such as heating, electrical appliances, lighting, and much more. At industrial and office environments, IoT systems can also monitor work hours of employees with presence sensors and manage their access to different areas. Smart devices may also be employed in public areas such as roads and public monuments to monitor their usage and maintenance. The areas of agriculture, health, logistics, and environment protection also have their own applications and usages for smart devices [6].

Microsoft performed an interesting survey that further illustrates the scale and coverage of IoT [7]. It included 3000 large scale companies (1000+ employees) and stated that around 90% of these have an active IoT project under it, although only 25% of these are in use (completely finished development and trial). Thus, most people either already are or will soon be involved in some IoT system as a ***monitored subject*** or as a ***user***. In other words, they will either be actively feeding data on their daily lives and actions to an IoT system or will be the ones using such data.

Within this vast and growing market, we expect the deployed IoT systems to overlap each other both geographically and in functionality as they coexist in the same infrastructure. In the context of smart cities for example, many systems coexist within the same communication and power supply infrastructure in order to manage and monitor waste, air quality, noise, parking, lighting, etc [8]. The situation complicates if we consider different networks can sense the same variables to offer competing services. Indeed, we expect the emergence of a complex infrastructure with several coexisting IoT systems offering a wide range of services (***multi-IoT***).

These systems spontaneously form commercial relationships as opportunities emerge to sell and acquire data from each other. This is motivated by reduced deployment costs and richer datasets due data diversity [9]. Additionally, it also helps the infrastructure scale as a whole since this should mitigate the occurrence of redundant deployment of sensors. The collection and processing of data from smart devices among IoT networks will be a key feature for such scenarios. That

is, they issue queries for the data of sensors from other networks, and these will disclose some of their data in order to answer queries.

These require the stream of timestamped data samples from sensors in order to: (i) always utilize updated information, (ii) reduce the required storage resources at low-end devices, and (iii) give controllers access to the time variation of monitored variables [10]. Time series allow systems to model and categorize different states of monitored variables, such as the intrusion of individuals in a secured area with a presence sensor or track rush hours in a city road with speed cameras [11]. We denote any data generation entity as a ***producer***: mainly sensor devices that measure variables from the environment. These push their data to ***consumer*** entities that use this information for decision-making. We also denote the set of data chunks transmitted by one or more streams as a ***dataset***.

These datasets have associated ***metadata*** attributes which describe their information [12]. Attributes such as measurement location (*e.g.*, GPS coordinates, room, borough) and data type (*e.g.*, temperature, power usage, speed). These are important in order to match consumer's queries with available data [13]. This matching is done according to each query's criteria that describe the data of interest. A power supplier may issue queries to collect the power usage from households in specific neighborhoods.

At any rate, collecting raw datasets can be both more verbose than needed and too costly due to the volume of data that needs to be transmitted and stored. Which imposes the requirement to preprocess the information to adapt it to the needs of the consumer [14]. Consumers will include desired operations in issued queries. We can cite two examples of these operations: filtering (*e.g.*, limiting datasets to attributes or periods of interest) and aggregation (*e.g.*, averaging measures based on time windows or common attribute values).

It is also common to only disclose datasets that have been anonymized with privacy-enabling operations. These operations remove details to boost privacy [15]. They may, for example: remove attributes and entries, replace exact values with approximations and provide aggregated values instead of individual values. The level of privacy achieved by these operations is measured with privacy metrics such as k-anonymity that limit the probability of deanonymization.

Then, in the context of multi-IoT, the data that networks disclose must be controlled to maintain the ***restrictions of the different producers***. Networks will screen queries in order to avoid the disclosing too much information and ensure that they comply with the privacy requirements set by data owners and monitored subjects. For example, queries must not request and access identifiers (*e.g.*, social security number) or they must request for averaged data instead of individual samples. If queries are not compliant with such requirements, they are simply dropped, and the consumer should issue a novel query.

Special attention must be taken to disclosed data since we expect these networks to act as brokers to each other. That is, after acquiring data from other domains, they may further disseminate the information to other domains. We expect some systems to be deployed with this objective, but also expect any other system to use this as a way to earn back some acquisition cost.

The inclusion of these brokering systems implies that ***data can be disseminated in an unrestrained manner***. Which means that a producer's data can be

disseminated to systems completely without its consent and knowledge. Suppose a smart home's power consumption information is collected by its power supplier, then disseminated to a private company. That private company may be able to extrapolate more information from that home by merging different datasets with information from that home. It has been show that it is possible to identify people from datasets with attributes such as zip code, date of birth, gender and race [16]; and the same can be applied to any kind of dataset. Since the producer has no knowledge of the destination of their data, it is impossible for it to detect the leak.

In summary, three main issues arise from using data from different IoT networks: interoperability, scalability and privacy. This thesis focus on the efficiency and privacy problems as stated by the main research question tackled in this report:

Main Research Question

How to provide ***in-network aggregation*** of IoT ***streams*** from multiple independent ***services and owners*** while respecting each of ***their privacy constraints***?

Scalability is key here since we are dealing with large and dense multi-IoT scenarios. Systems are expected to contain hundreds of devices each and the incorporation of novel systems should be facilitated. Aggregation also plays a central role since we can apply it as data travels the network to lower network load and bring privacy. Thus, we investigate the means to apply aggregations as close as possible to data producers in order to decrease the amount of information forwarded.

Interoperability takes a secondary role in this thesis as we do not focus on it. However, we maintain the condition that each IoT system operates with different devices, protocols and semantics [9]. This enforces the need to normalize the means to communicate among systems to identify and exchange information.

Then, *security* is greatly affected by dealing with a multi-owner environment due to differing objectives, creditability and restrictions of each service. For example, a network may wish to disclose information only to non-competitors, to disclose only to a specific list of others or to disclose only data from specific time windows. An inter-IoT infrastructure must account for this diverse set of conditions and enforce the requirements of each producer network.

We provide three main contributions in this thesis that diverge a bit from usual approach of security among domains. Our approach provides inter system communication by taking a data centric approach where data is anonymized as it is disseminated. Anonymization is a powerful tool to:

- enforce each producer's requirements through existing parameterizable privacy metrics such as k-anonymity and ϵ-differential;
- protect data against further dissemination of data after initial disclosure;
- motivate reuse of data streams to answer multiple queries without complex key management services.

This thesis is structured in six chapters. In Chapter 2, we provide the necessary background knowledge to understand our research and also insights about the state of the art of proposals for private-aware inter-IoT communication. Chapter 3 describes our first contribution, an evaluation framework for multi-IoT networks which both describes data exchanged among privacy-aware IoT services and realistic traffic generation for these services. Additionally, we take advantage of this framework to study how LoRaWAN behaves in multi-IoT networks. In Chapter 4, we present an architecture for efficient inter-IoT communications while respecting privacy constraints based on processing functions such as filtering, aggregation, encryption, and anonymization primitives like generalization and noise addition. Chapter 5 extends the architecture presented in chapter 4; it introduces a mechanism for IoT networks to discover data from others and correctly collect their data while applying an aggregation function that complies with privacy requirements. Lastly, in Chapter 6 we present our conclusions over the research done during this doctoral program and discuss insights on interesting research paths in the field of privacy-aware inter-IoT communication.

Chapter 2

State of the Art

Contents

In this chapter we discuss the different areas of research that we focused on to answer the research question of this thesis. We first go through the state of the art for IoT streams and the subarea of Named Data Networking (NDN) streams for IoT. Then we introduce multi-owner IoT and the subtleties that it brings for secure streams.

2.1 Large scale data streams

We investigate continuous transmissions in infrastructures composed of thousands of producer devices. Scalability is a major issue as the system must account for all producers in order to correctly and efficiently answer queries.

2.1.1 IoT Data Streams

In IoT ecosystems we find devices with different purposes and sensors are the main ***producers*** of data. We find these in large numbers in order to monitor many of its characteristics like temperature and air pressure. The report[1] provided by Ericsson

[1] https://www.ericsson.com/en/reports-and-papers/mobility-report/reports/november-2021

in November 2021 points to nearly 15 billion IoT devices connected in 2021 (with the forecast doubling that amount by 2027).

These producers ***push*** their measurement data into the network in order for controllers, actuators and other entities decision-making entities to utilize (***consume***) it for their various objectives [17]. A single sample from a producer provides very restricted information to these consumers. Thus, producers will continuously ***stream*** timestamped measurements to allow consumers to be in possession of constantly updated data and store full timelines [13]. These timelines also allow consumers to extract more knowledge due to the time dimension such as the extraction of behavior patterns and detection of events [18].

Anomaly detection is a common type of application for IoT data streams. From the acquired timelines, statistical and machine learning models are applied in order to detect different states of the monitored system. In smart cities, we can cite the example of this type of application where anomalies in monitored road speeds can indicate the presence of ice or collisions on roads [11].

Producers will push their data, through the network, to consumers that ***subscribe*** to their streams: either in a timely manner or based on the occurrence of events [19]. The former type pushes novel samples at a given fixed ***periodicity*** while the latter pushes novel data when specific conditions are true. An example of application with time-based streams is weather monitoring that frequently and consistently (daily, hourly, or even every minute) updates information like humidity, temperature and wind speed [20]. On the other hand, an example of event-based stream application is automated security where sensors push a notification as doors open and close or when sensors detect the presence of an individual in an environment [21].

Streams are mainly assumed to remain enabled until the battery of sensors run out. That is, consumers that subscribe to a stream will receive updates for as long as a sensor remains in activity. Emphasizing the need to conserve the battery life of devices to maximize operation time [22]. But is it also possible for streams to be enabled for limited amounts of time.

When producers are not directly associated to consumers, *i.e.*, they are unknown to each other and do not always provide/collect data to/from the same entities, the networks requires a middle entity to mediate communication. We denote these entities as ***brokers*** and producers will ***publish*** data to brokers while consumers ***subscribe*** to them [23]. This communication model is known as the ***publish-subscribe model*** (pub/sub) and is widely used in IoT networks [24, 25, 26, 27, 28]. Several market ready solutions are also available for the pub/sub archetype, *e.g.*, MQTT [29] and CoAP Observe [24].

Message Queuing Telemetry Transport (MQTT) adopts a client/server architecture where the server acts as a broker and the client can both publish and subscribe to brokers. This application protocol employs small header size and limited payload size to be used by resource restricted devices. It also supports three levels of delivery guarantee, the first without any guarantee, second guaranteeing delivery but with possible replication and last with guarantee but without replication [29].

Constrained Application Protocol (CoAP) is another application protocol based on HTTP that initially only offered a single query / single response. However, it was extended to allow consumers to subscribe to receive constant updates from

producers. It also offers proxy functionality that allows broker behavior as the proxy answers requests for multiple producers [30].

While CoAP has a lower overhead than MQTT, the latter shows lower delays for low packet loss conditions [31]. Both protocols are very similar, so these are deployed based on the preference of network operators.

The publish-subscribe paradigm decouples producers and consumers as direct subscribing does not occur. It brings independence among these and more privacy as they also have no knowledge of each other. And thus, consumers will issue ***queries*** to brokers that select producers based on descriptions of the data of interest [14]. Indeed, consumers subscribe to a type of data from the pool available at the broker. We go into further detail on how these queries work in the following section.

2.1.2 IoT Queries

As producers subscribe to brokers, they describe their information in order to be matched with queries. These descriptions are provided in the form of ***metadata attributes*** and describe information such as the type of data, location of measurement and other descriptions of the monitored ***subject*** like patient identifiers, employee tags and room numbers [13].

The current main language to write IoT queries is SPARQL [32] as several industry-ready platforms support have support for it. However, this language is quite complex and difficult to understand due to its syntax and specific semantic. We even have efforts to create translators to facilitate its use [33, 34, 35]. Instead of fully introducing it, we employ a SQL-like language later in this thesis to facilitate understanding.

These queries are issued to known brokers that check the registered producers metadata and enable streams according to the ***metadata criteria*** given. Thus, the result is the desired disassociated relationship among producers and consumers. Which allows easy inclusion of producers and consumers in the network [23].

Complex models of metadata description are necessary due to disparities in the syntax and semantics of data among producers. These may store data in different storage formats [36, 37] or different data schemes. Brokers must account for situations like when different attribute names and values have similar semantics. For example, values such as “government employee” and “public employee” can be considered the same value for a “job position” attribute. However, we exclude this from this thesis and consider that such a method is in place to match queries with producers descriptions [38, 12].

We rather focus on the processing requirements of such queries. Instead of consumers demanding the raw data of producers, they will demand the data under some computation function, *e.g.*,:

Filtering *i.e.*, limiting results like to a specific time period or to a subset of attributes;

Categorization *i.e.*, classifying measurements into categories like different age groups or detect abnormal measurements [11];

Aggregation *i.e.*, grouping samples and applying descriptive functions such as average, minimum and maximum [39].

These are important to provide consumers with customized (more valuable) data. Additionally, as consumers may enable several streams with a single query as networks scale [10], streamed data can quickly become a heavy burden on the network that has to provide for multiple queries. Consequentially, administrators introduced different means to handle the large amount of traffic and relieve network resources. Data aggregation is one of such means [13] and we focus on aggregation algorithms in the next section.

2.1.3 Aggregated Data Streams

The most basic definition of aggregation functions defines them as functions that when applied to a set of input, produce only one output with information describing the input set. These can be ***lossless or lossy***, that is if the original data can be extracted from the result or not, respectively. We focus on lossy aggregations such as statistical descriptions that provide a summary of the input data while making it difficult to extract the original data. These operations can also be ***duplicate sensitive or insensitive***, which are functions where the final result changes depending on duplicated data or not. Minimum is a duplicate insensitive function which, but the average function is not [39].

Lossy aggregation functions are commonly used with large datasets due to the verbosity of data which makes unnecessary the acquisition of the full dataset. By acquiring aggregated data from producers and brokers, instead of whole datasets, consumers save network bandwidth and storage while acquiring preprocessed information [13]. Bandwidth is also saved by aggregating data as devices forward it from producers. These act as privacy enabling operations that both preserve the utility of data and veil sensitive information.

The strategies to apply these aggregation functions in IoT networks are classified in three big groups: ***centralized, clustered and tree-based*** [40].

Centralized aggregation is the conventional way where all data is collected at a sink, gateway or consumer (destination) to be aggregated. It allows maximum flexibility as raw data can be aggregated in multiple ways to answer multiple queries while already aggregated data may be impossible for another query. For example, weekly averaged data can be used to answer a query for monthly averages but not for daily averages [41].

Centralized aggregation is still used by some systems, specifically those that push all data to the cloud and use its resources to process it. We however discard this method as it does not bring the benefits of lowering bandwidth and storage costs.

For clustered strategies, devices elect a cluster head from surrounding devices and use it as the node to forward its information. Forming a group (*cluster*) of devices with a single cluster head to transmit the data aggregated from that cluster. That way, each non cluster head only transmits over short distances and the cluster head aggregates data from its cluster before transmitting further into the network. This assumes that devices spend more battery to transmit over long distances due to the higher power necessary for successful reception. Thus, by making most devices

transmit over short distances and rotating the cluster head that has higher power consumption, the network as a whole lasts longer. The biggest examples of this strategy is the Low-Energy Adaptive Clustering Hierarchy (LEACH) protocol and the family of protocols derived from it [42].

In the original version, devices assign a priority between 0 and 1 and nodes with high priority become cluster heads. The equation that determines the priority of devices simply gives equal probability of nodes being cluster heads which results in devices spending equal time being cluster heads and equally consuming their battery [43]. However, the probabilistic nature of the priority function makes it possible for devices to be "unlucky" and the distribution of devices in clusters to be unequal, which results in unequal battery usage among devices.

Then, several other variants were proposed with different priority functions and methodologies of selecting cluster heads but with the same aggregation strategy. An exhaustive list of variants is of no interest here since this thesis is not focused on clustered aggregation (see [42] for an exhaustive list). But we can give the example of Improved-LEACH [44] as it is one of the main variants and illustrates well all others. Here, cluster heads are still selected using random numbers but the number of neighboring devices, the distance to the receiver, and the remaining energy of the devices alter the probability of being the cluster head. This results in better average battery life among devices in the network.

The original LEACH results in low height trees where two hops are enough to collect data from all sensors. Other variants forward messages through other cluster heads which lower the distances of transmissions to save battery life [45].

Other clustered strategies use similar mechanisms of cluster election and forwarding to the destination, *e.g.*, changing the metrics of cluster head election or by routing data among cluster heads [46]. A comprehensive survey of this area of research can be found in [47].

Clustered aggregation can be seen as trees where the destination device is the root, cluster heads are intermediate nodes and other devices are the leaves. But this is not the only way to aggregate using such structures, we move on to discuss these others.

Trees-based strategies are the major method of local aggregation of IoT data as they enforce that each producer data is only acquired once and only requires each vertex to know its parent in the tree [39]. By forming a tree of devices from the interconnected devices rooted at and directed to the consumer, internal vertices of the tree can aggregate data hop by hop and only transmit the aggregated result, instead of individual samples.

Conventionally, devices will create a tree via the broadcast of the query by the consumer. The query message includes the level of the tree where the parent is and vertices can use this information to select their parent in the tree [39]. The Routing Protocol for Low Power and Lossy Networks (RPL) [48] and the TAG [49] protocols are simple examples of this kind of algorithm.

On the other hand, PEGASIS [50] builds trees in the form of chains. Neighboring devices elect neighbors in the chain and a leader of the chain. Devices will forward data to the leader through the chain and data is aggregated at every hop. The leader forwards the resulting aggregation to the destination and is periodically replaced in

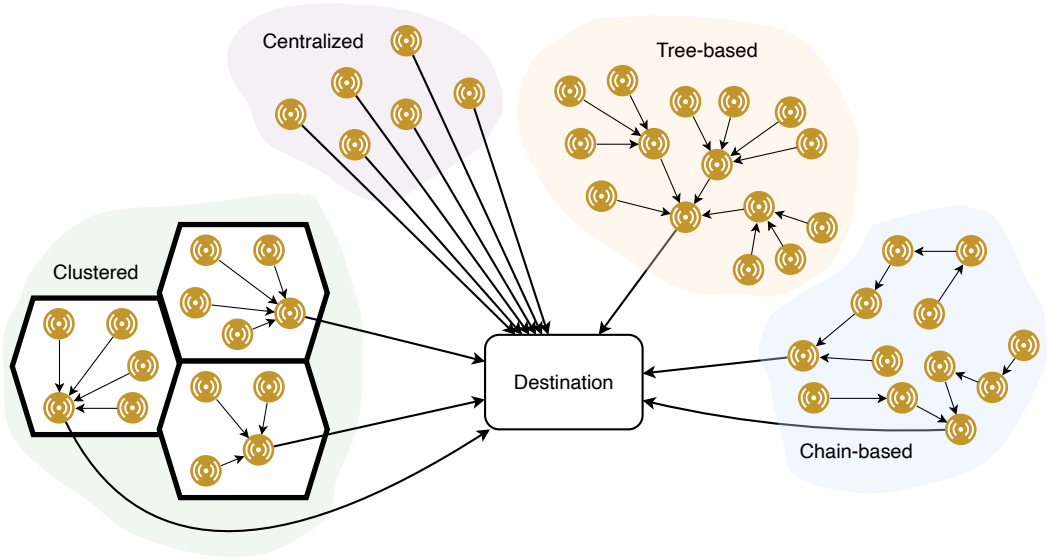


Figure 2.1: Types of aggregation

order to equalize energy consumption.

In order to improve reliability of these protocols, the usage of multipath aggregation has been proposed [51]. This is achieved by maintaining multiple paths to the destination of aggregation, *e.g.*, picking more than one parent during the construction of an aggregation tree to allow the device to forward data to any active parent. Then, in case of failure of a device, nodes can use alternative paths to the root of the tree.

Privacy then becomes a key issue for in-network aggregation as the aggregator entity needs access to data in order to apply the aggregation operation. ***Homomorphic encryption*** was then proposed to solve this issue as mathematical functions can still be applied when this type of cipher is used [52]. Formally, Given a plaintext space $Plain$ and an encrypted space Enc, an encryption function $f : Plain \rightarrow Enc$ is homomorphic if it maintains the following for two operators $\circ$ and $\diamond$:

$$\forall a, b \in Plain, f(a \circ b) = f(a) \diamond f(b) \tag{2.1}$$

Here, the $\circ$ is an operation over plaintext space such as addition or multiplication, and the $\diamond$ operation is an operation that can be applied to the encrypted data such that, when the result is decrypted, it will be equal to the result of the other operation (sum, multiplication) [53].

For example, given that the $\circ = +$, $Plain = \mathbb{R}$ and $E, \diamond$ are such that the homomorphic property is valid (Eq. 2.1), we can use this to calculate sum aggregation privately. In smart grids, smart homes can encrypt their individual consumption using f, then the building can apply $\diamond$ to collected values and transmit the result to the power supplier that can decrypt the result to bill the building.

However, a limited number of sets of such variables has been found to hold this property [54]. Thus, it cannot be applied in every situation, so we chose not to consider this type of solution in this thesis.

At any rate, we see a vast amount of work has been done to aggregate data

from IoT networks, to optimize collection, extend battery life and provide privacy. And we clearly see that ***the paradigm shifts to be content oriented*** instead of connection oriented. In the content centric paradigm, consumers describe the data that they want instead of issuing request to producers. This matches well with semantics of streaming data in IoT networks since we are not any more interested in individual producers.

Among the proposals for content centric networks, we highlight the Named Data Networking (NDN) [55] stack as one of the main proposals. In this thesis we use NDN to more efficiently disseminate information and reuse the content centric mechanisms that it adopts. We shall go into details of it in the following section.

2.1.4 Named Data Networking

NDN [55] is a clean-slate approach to the current Internet stack, which fits very well the needs of IoT applications to handle and directly forward pieces of data. It is one of the main proposals for content based networks due to its adoption by the academic community and its open source code base with a ready to use daemon.

As per the content centric mentality, consumers will query the network by describing the data requested. Routers will then forward these queries in the direction of producers that answer it.

In NDN, routers issue queries via ***interests*** packets with the name of the requested data. These names are hierarchical URL-like strings such as: */someproducer/measurements*. These names replace numerical addresses and enable semantic requests, prefix-based routing, and route aggregation. The Forward Information Base (FIB) of NDN routers associates the prefixes of data and the next hop for packets for that prefix. This forwarding scheme leverages semantic forwarding. For example, by adding the prefix */local* to the FIB, routers can disseminate information locally.

Another feature of NDN is the signature of interests and data packets, which allows for the authentication of consumers and producers of information. These both may include the signature and the name of the certificate of that issued it. The certificate is then acquired via the transmission of an interest to its name and the authenticity can be verified.

As any entity may verify the authenticity of data, every router can cache all data it forwards in its *Content Store (CS)*. That way, a router can use this cache to reply directly to any novel interest that matches the same piece of data. A router cannot forge a chunk of data by its own.

Figure 2.2 describes the forwarding scheme of NDN. Here we see that as interests arrive, the router checks its CS to answer the requested data if available. Otherwise, the novel interest is registered in the *Pending Interest Table (PIT)* and the request is forwarded using the FIB.

Each entry in the PIT of this table accounts for one or more interest packets for a given data name. When an entry already exists for a name, the PIT entry is simply updated to account for the novel interest, instead of forwarding a novel interest for the data. Then as data packets arrive, routers check the PIT to forward information to all requesting consumers. Take notice that all data forwarded is also cached in the CS of the router.

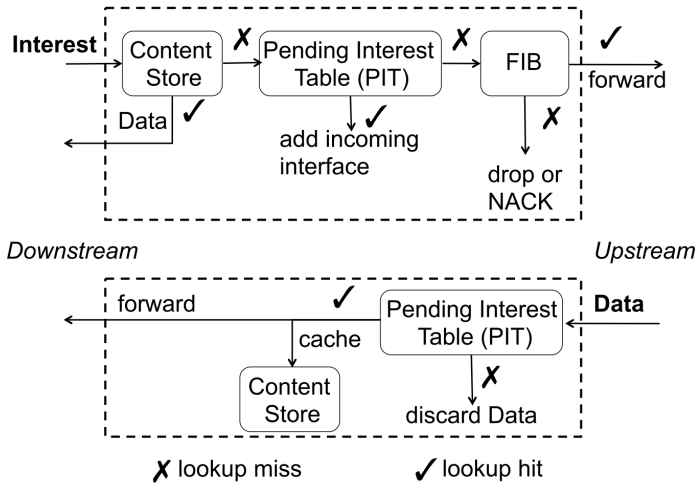


Figure 2.2: NDN Forwarding Scheme (extracted from [55])

Unfortunately, ***NDN is not directly applicable in IoT*** due to missing features and IoT's limitations [56, 57]. For instance, the packet authentication takes a large toll on the battery life of resource restricted IoT devices due to all ciphers involved. Caching is another issue as most devices are restricted on their storage and remain inactive for long periods to save battery [58]. We now focus on some proposals to provide the necessary IoT features to NDN. Please refer to the citations in this paragraph for a more complete list.

Subscription to streams are one of such mechanisms that are not supported as per default there is a 1-to-1 association between interests and data messages due to the forwarding behavior of deleting PIT entries as they are answered. Which leads consumers to constantly transmit interests packets for novel samples under a given name, instead of the producer pushing novel data as it is available. Such behavior enforces congestion control since a congested network will also drop interest packets that would further congest the network [59].

Mechanisms to make PIT entries persistent allow for data streams in NDN [60]. That is, instead of discarding entries when data is forwarded upstream, the entry remains until a given number of packets is forwarded or a duration expires. This allows the association of multiple data packets of a given stream to the same interest since the reverse paths are kept and producers can push data as needed.

Other solutions to insert data entries in interest packets in order for it to be pushed or to utilize the FIB to also forward data packets [61]. The first takes advantage of the fact that NDN packets do not have a size limit and add data in the parameter section of interest messages. The other changes the downstream routing scheme (see figure 2.2) to also route data packets using the FIB in addition to the PIT. However, these two last proposals break the semantics and allow for malicious routers to flood the network by inserting phoney data.

the publish-subscribe communication paradigm is also not natively supported by routers. In other words, NDN does not have a mechanism for consumers to be

notified on novel data. Its scheme is considered a pull scheme since consumers must issue an interest packet to acquire data.

COPSS [62] was one of the first to introduce broker like entities called rendezvous nodes. Producers publish data to these nodes by issuing publish packets (a novel data packet type) with */rendezvous/* prefix added to its data name. That prefix takes data to the rendezvous node that strips the prefix and forwards the data to subscribers.

A name convention is also adopted to select data in a rendezvous node where the hierarchy of a data name can be used to describe the desired granularity. For example the data name */temperature/public/courtyard* to denote measurements of a public courtyard can be selected through a interest for /temperature to select any temperature measurement */temperature/public* to select temperature measurements for any public area.

Mobility support can also be employed pub-sub for vehicular NDN where vehicles that subscribe to remote rendezvous nodes via road side infrastructure. Here, the covering roadside base station changes as vehicles move, which make PIT entries obsolete, thus making vehicles resubscribe as they change base station. [59].

Distancing from the method of using a broker to forward data, we may also introduce a subscription manager that receives interests and answer with an authorization that can be used to directly subscribe to producers [63]. On the other hand, Psync [64] periodically notifies the consumer with novel names for a stream. The consumer will issue a query to a Psync enabled router that will periodically answer with the data names of data packets to answer that query and the consumer will directly request these names.

Caching with restricted devices is challenging due to limited available storage and due to the devices alternating between active and sleep modes to save battery [57]. The default policy of caching everything provides high redundancy on data that improves cache hits but storage fills quickly and can be ineffective due to constant rotation of data in the CS.

Thus, several cache policies have been proposed for restricted devices [65]. These are based on:

content type where nodes decide to cache based on predefined data of interest, for example when networks have stricter quality of service for specific data types;

content popularity where nodes keep track of data popularity based on the amount of requests for data names [66];

node characteristics data is stored based on characteristics such as distance to producers and remaining battery [67]; and

In-network computation is introduced to the NDN paradigm by Named Function Networking (NFN) [68] that proposes to integrate lambda-calculus in interests, where each NDN router may process the data before sending the reply. Similarly, [69] proposes to use named unikernel functions that run on top of a virtualization layer and may be moved between routers to improve delay or bandwidth. An

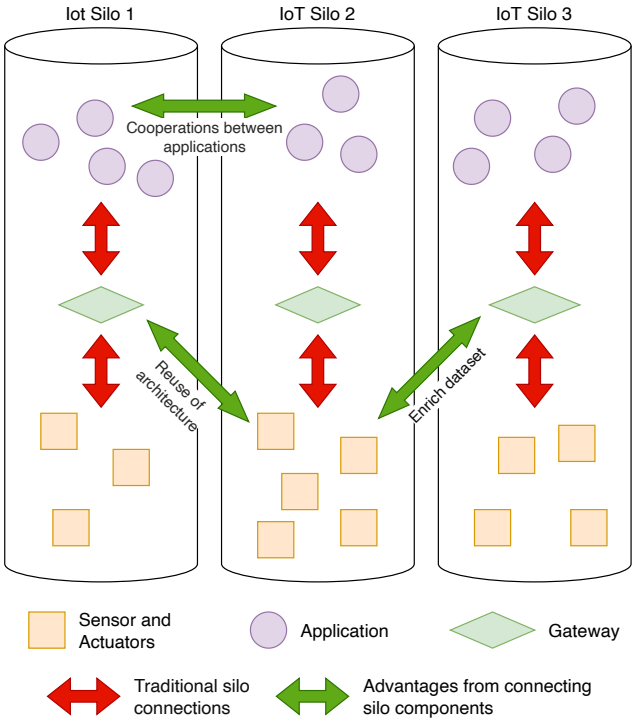


Figure 2.3: Example of IoT Silos (Based on [38])

extension has also been proposed to cope with edge computing architectures to preferentially select resources which are closer to producers [70]. [71] builds on the concept of named functions by providing another service allocation strategy that consider the available resources of devices in addition to the proximity to data sources.

NDN brings several benefits to IoT networks and several extensions have been created to accommodate its requirements. However, NDN is not applicable throughout the whole IoT ecosystem due to its limited resources. Namely, packet authentication requires considerable processing that reduces battery life and caching is restricted due the reduced available storage and to devices being frequently inactive to save battery [57].

2.2 Secure Streams in Multi-Owner IoT Systems

IoT networks are naturally segmented as each service deploys its own set of devices and infrastructure for their objectives. These monolithic type of deployments are known as IoT ***silos*** (Fig. 2.3) [38]. It allows each network to operate using the set of protocols, hardware and behaviors to match the expertise of its operator and its objectives. However, multiple advantages appear by breaking down these silos in order for them to interconnect and cooperate [9].

However, as the density of networks increases in scenarios like smart cities and smart grid, redundant deployments cause scalability problems. And the deployment of multiple IoT silos naturally bring about this occurrence [9].

Thus, networks have begun to break apart their silos and expose internal resources to other networks. Allowing other networks to query their sensors and/or utilize their transportation network. Aside from reducing redundant deployments, services improve data diversity by querying multiple data owners and lower deployment costs by using existing resources [72, 73].

Unfortunately, security issues and interoperability issues arise from interconnecting IoT structures.

2.2.1 Diverse IoT Systems

Heterogeneity is a key characteristic of IoT networks due to all the different devices that compose them. One has to consider heterogeneous infrastructures that support multiple protocol stacks. We can exemplify this with the CoAP [30] and MQTT [74] protocols that offer similar features to IoT applications.

To facilitate integration, a proxy may help to interconnect producers and consumers [75]. A network will place this at its edge to answer remote requests for local data. However, existing solutions propose to rely on access control strategies to control the dissemination of sensitive information [76]. Alternatively, we may apply an approach based on middleboxes, to interconnect different services. Such as the concept of semantic gateways, in charge of relaying annotated data [38]. However, middleboxes do not address the privacy concerns of a multiservice scenario since their purpose is to enable interoperability, not enforce a set of privacy constraints.

An overlay of CoAP servers is able to build a federation of sensor networks [77]. However, their focus is rather on wrapping sensors, actuators and other entities, instead of the privacy concerns raised by such multiservice architectures. Similarly, content proxies serve as a central, stable entity to connect publishers and subscribers, but still, a multiservice environment is not considered [78].

Heterogeneity can also be found in the communication protocol. Among the main communication technologies for IoT, Low Power Wide Area Networks (LPWAN) allow very scalable connectivity to low-end devices due to long ranges and capacity to handle thousands of devices per base station. NarrowBand IoT, Sigfox and LoRaWAN are the three leading technologies for LPWAN [79].

Sigfox [80] is a proprietary technology of UNB technologies and operates in the unlicensed bad at a maximum of data rate of 100bps. NarrowBand IoT [81] is a standard defined by 3GPP, it reuses some of LTE technology and may be deployed within or outside existing LTE deployments. Finally, LoRaWAN [82] is a standard created by LoRa-Alliance and, due to its signal design, it can achieve high ranges and channel multiplexing. We use LoRaWAN in our first contribution to exemplify how much infrastructure a scenario of coexisting IoT networks requires. In the following section we explore the details of such networks.

But the diversity of IoT ecosystems goes beyond heterogeneous devices and protocols [9]. Services will have different objectives, owners, semantics and requirements. They can be deployed for data collection, distribution, decision-making or actuation.

Then, privacy turns into a specially sensitive requirement as data is streamed among services with different owners, *e.g.*, from a data collecting one to a decision-making one. Maintaining privacy is not trivial since data is collected under different

levels of consent from monitored subjects. It is necessary to maintain mechanisms that stream and compute data while complying with the privacy requirements of each producer involved. We focus next on the security mechanisms that are used in such scenarios.

2.2.2 Middleboxes & Access Control

Security solutions must be aware of the limited resources of IoT devices since costly operations jeopardize battery life due to longer processing and extra network exchanges. Essential operations such as encryption and radio transmissions are specially costly [83, 22].

Thus, IoT networks deploy dedicated devices at their edge in order to manage secure channels and access to internal resources [84]. These take charge of costly security operations since they have access to more resources when compared to sensors and actuators [85]. Operations such as authorization checking and granting in addition to key management are examples of these operations.

They verify queries and distribute necessary credentials (*i.e.*, encryption keys) to the issuers of these queries. This verification can be performed based on different access criteria [86]:

Discretionary allows data owners to directly specify the consumers that can access data;

Mandatory is based on clearance level where data owners give a specify the minimum level that consumers must have to access data;

Role-based group access permissions under several pre-defined roles and consumers must have the correct role to access data;

Attribute-based assumes that data, consumers and producers have metadata that describe them. Then, data owners give access depending on these attributes.

Attribute-based access control provides the most granularity and is the closer to the metadata-based queries. Access consents are mapped from the consent agreements between the monitored subjects and the network of sensors. For example, patients will disclose their information to their doctors but not to other users of a healthcare institution [87]. Then, a trusted third party will distribute credentials (access keys) if a query and its issuer comply with policies. This approach requires several exchanges with a middle entity in order to acquire the access keys to enable streams [88]. After credentials are given, these resources may be queried directly by external entities.

The presence of a trusted third party is a strong assumption for multi-owner infrastructures. Recent works have introduced blockchains as a means to remove the necessity of a third party that manages validates queries and manages credentials due to the strong trust requirement [89, 90]. Blockchains are distributed databases used to store transactions among unreliable entities. All entities in the system have access to this database and transactions in the chain are immutable This allows anyone to verify the history of transactions to validate the state of the system.

Additionally, recent blockchains provide the feature of ***smart contracts***, where entities place code that may be run by any entity upon request. These are called smart contracts and allow for anonymous and unrelated entities to request, execute and collect data in the blockchain in a verifiable manner.

The basic behavior of blockchain based access control is:

1. Consumers will publish their attributes on the chain;
2. Producers will publish their data and policies on the chain;
3. When a consumer decides to query data, it will invoke smart contracts to acquire credentials;
4. Smart contract will check published policies to publish a response transaction with compliant data names and credentials;
5. With the credentials, the consumer can collect data directly from consumers (not through the blockchain);

This scheme of access control is a good way to remove the trusted entity and offload the computational cost of key management and distribution. However, the privacy of all entities is hindered since all attributes, policies and queries become public. Malicious entities can use the attributes of consumers to identify entities even if these are anonymous and profile their interests from their queries.

Dual blockchain schemes can be used to lower available data by keeping one chain public and one chain private [91]. In this case, encrypted attributes are stored in the private chain and all else is recorded in the public chain. The only ones with access to the private chain are attribute managing entities, but these can only decrypt the attributes of a subset of producers that share their keys. When a query is introduced in the public chain, these attribute managers check attributes and distribute credentials as previously described. This secures the privacy of the attributes of producers and consumers.

At any rate, the access control schemes discussed above can correctly provide access, however data can still be leaked by the destination. Thus, data must still be "protected" to prepare for further dissemination. Which is the objective of anonymization operations, it removes unnecessary information from data to protect the producer while maintaining its utility to the consumer [92].

This is specially important in our multi-owner infrastructure where maintaining consent policies as streams occur among different data owners and operators is a major requirement. Data can be redistributed between systems after acquisition from the original producers without the intervention of these. Indeed, we consider anonymization a requirement in this infrastructure, and we discuss the details of it in the following section.

2.2.3 Stream Anonymization

Anonymization provides privacy by decreasing the precision of data. Aggregating data from multiple sensors hides the specific values of each sensor while providing information on the global population [93]. We may also filter and mask sensitive information from data until privacy requirements are reached.

Normally, datasets attributes are classified in three categories [92]:

Identifiers attributes uniquely identify some entity like social security number or employee number;

Quasi-Identifiers attributes can identify some entity when used together. For example, if the age, gender, height and weight of someone is known, and the same values are found in entries of a dataset, it is likely that those entries are of that person;

Sensitive Information these are the measurements and metrics that are listed with the other attributes. These are highly sensitive since they disclose information on the subjects behavior and condition.

During anonymization, the identifiable attributes are suppressed or replaced with random identifiers to remove direct association. Quasi-Identifiers are then altered in order to decrease the probability of associating subjects with their sensitive information. Sensitive information may be left as is, categorized (*e.g.*, changing exact age into age group) or added controlled noise to occlude exact values.

This process is done until privacy metrics such as ***k-anonymity*** [94] and ***ϵ-differential*** [95] are reached. They quantify levels of privacy in terms of the probability of deanonymizing information. They measure how similar data entries are from each other and how whole datasets are similar to each other.

A k-anonymity dataset mandates that at each entry is indistinguishable from k-1 other entries. In other words, every combination of quasi-identifiers values has at least k entries in the dataset. Even if a subjects' information is known to be in a dataset and the identifiable attributes are known, the probability of identifying it is $1/k$. Several algorithms have been proposed to reach the k-anonymity [96, 97, 98] that focus on clustering data entries in order to minimally change quasi-identifiers to maximize utility of data.

A ϵ-differential dataset denotes that a dataset does not change significantly from adding or removing any single entry. For a given function K that produces ϵ-differential datasets from any dataset, $\forall S \subseteq Range(K) \wedge \forall$ datasets D_1, D_2 that differs on at most 1 entry:

$$Pr[K(D_1) \in S] \leq exp(\epsilon) \times Pr[K(D_2) \in S] \tag{2.2}$$

For every possible result of the K function, *i.e.*, for every ϵ-differential query answer, the addition or removal of a single entry will not significantly change deanonymization probability. Several algorithms have also been proposed for ϵ-differential [99, 100, 101]. These algorithms focus on adding enough noise to make entries similar enough. Distributed noise addition is also possible within limitations [102, 103]

In a multi-owner IoT infrastructure, producers are likely to have different levels of consent for collecting data. For example, patients data will have very strict anonymization requirements while data collected in public areas will have laxer restrictions. Imposing the same privacy level to every producer would mean that

the most strict level must be applied, which would result in increased difficulty of acquiring data. Thus, complying with the requirement of each producer as closely as possible is necessary.

In [104], authors have considered a privacy model based on ϵ-differential. Privacy requirements are modeled based on the parameter ϵ, and each producer has its own privacy requirement. The group of devices of a producer applies a stochastic model to anonymize and disclose information. This model is frequently updated gradient descent algorithm in order to limit information loss.

2.3 Summary

Streaming data from different data owners in a large IoT infrastructure allows networks to both enrich their services with diverse data and to lower deployment costs by reusing existing resources. However, it exposes scalability issues as thousands of devices are involved in the enabled streams and security issues as ownership boundaries are crossed.

The problem of inter-system communication is still an open subject. Translators are able to translate protocols and data formats but are limited when dealing with conflicting semantics. And security middleboxes proposals focus on blocking specific connections and providing attribute based access control. Concerning privacy, the larger portion of the literature on data streams focus on access control instead of anonymization. However, this is necessary for inter-system communication to limit the leaks when data is further disseminated. Additionally, little has been done in scenarios where data is acquired and aggregated from heterogeneous and untrustworthy sources. The problem is specifically complicated due to different levels of trust among services.

The consent from monitored subjects must be respected while data is forwarded and aggregated via untrusted networks. The different levels of privacy of these subjects must also be closely observed in order to make the most data available to others.

Our work investigates all these areas to provide privacy aware disclose of information among services. We focus on providing privacy by anonymization so that it can be disclosed among domains without complex encryption and access control mechanisms. That way we maximize the utility of data and facilitate dissemination and reuse of information. Through NDN we leverage semantic requests and efficient forwarding then extend it to reuse data streamed among services to answer multiple queries.

Chapter 3

Case Study of Multi-IoT Services under LPWAN network

Contents

The plethora of smart city services highlight the various benefits that IoT can bring to the population, companies and local authorities [105]. A collection of sensors is typically disseminated in a smart city, and monitors a large set of environmental parameters such as traffic speed, air quality, weather, noise, etc.

Smart cities also expect that the different systems cohabit with each other and exchange data while protecting sensitive data. Isolation has a cost [106]: the same radio resources have to be shared among a larger number of flows, which is particularly prejudicial in the ISM band.

In this chapter we investigate a scenario of Intelligent Transport System (ITS) where distinct services share data to offer a globally efficient and privacy aware transportation system for smart cities. We seek insight on a realistic scenario where multiple IoT systems cooperate for their objectives. More specifically, we want

to better understand the data streams among services, what kind of processing operations are applied to data and how privacy is maintained.

Existing system models for IoT focus on a single application or service and the infrastructure necessary for it [107]. While our scheme is a formally defined ecosystem of services that may both provide and consume data to and from other services. Our ecosystem is also defined in a way that privacy is provided by design by disassociating producer and consumers and by carefully selecting the data collected.

We take advantage of datasets of two real services from large cities to propose a system model that generates packets mimicking a realistic scenario, with all the heterogeneity (*e.g.*, downtown vs. suburbs) and temporality of large scale ITS. The traffic model we derive from these datasets is useful to realistically assess the communication needs in smart cities.

Finally, we exploit our construct to answer a secondary question: can LPWAN be used as the means of communication for multi-IoT services? How should these services be deployed in order to handle this traffic? We rely on a LoRa model capable of inferring the Packet Error Ratio (PER) of an arbitrary deployment of LoRa gateways to estimate the scalability of such LPWAN solutions in realistic deployments.

3.1 Proposed Multi-Service Model for Intelligent Transportation

We limited the services in our model to those that have considerable network complexity and actors in common. That way we have considerable interest for network applications and data dependency among services. The services which compose our multiservice model are:

Ride hailing where platforms connect passengers with vehicle drivers (*aka.* on-demand taxi). Passengers will subscribe to multiple platforms to search through several price options. These options are updated in real time to mirror changes on availability and other conditions. On the other side, vehicle drivers subscribe to platforms in order to receive available rides that best fit their preferences and schedule. Thus, allowing drivers to make the decision on which would be their next ride. These rides would also be constantly updated as several drivers concurrently pick rides. In a smart city, this service could be provided instead of a global platform.

Smart parking where sensors monitor occupied parking spots, registering when vehicles arrive and leave spots. Vehicles in transit query these sensors to find available parking. Consequentially, drivers spend less time randomly looking for parking spots which decreases road congestion. The system will additionally bill vehicles with the information sensors send to the company that owns each area.

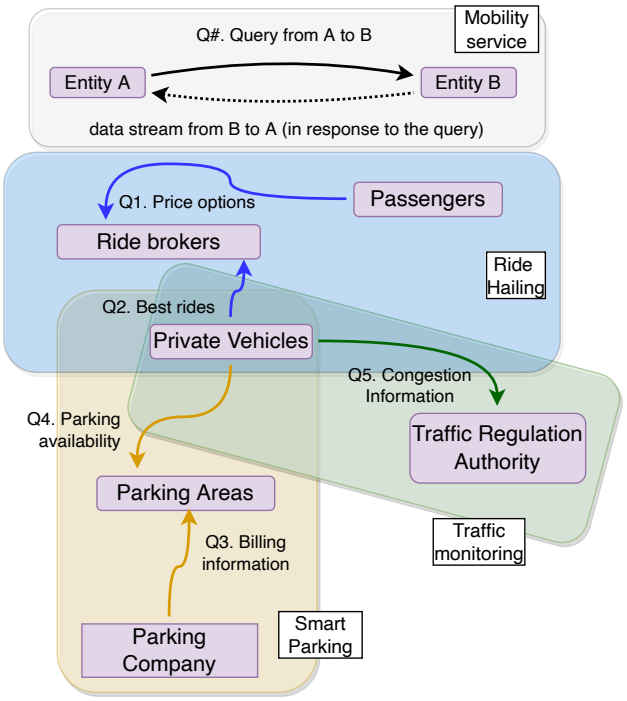


Figure 3.1: Overview of services

Traffic monitoring where the ITS measures the average speed for each road segment by capturing snapshots of the speed of vehicles. Snapshots are a key aspect here since full traces of vehicles would generate too much information on the habits of drivers. Simple snapshots of several (private and public) vehicles already provide enough information on road segments. This information is then also used by other systems for traffic regulation (*e.g.*, vehicle routing problem). We decided to exclude these other systems from this since we could not realistically model them.

Figure 3.1 illustrates our scenario, where the different applications and entities share information with each other. There you can see the different information exchanged among each component of these services. We formally define these components by listing the queries used to enable streams. In the following sections we define the query language used and then the queries themselves.

3.1.1 Query Language

We rely on a slightly extended version of SQL to explicitly define the queries (*i.e.*, the arrows in Fig. 3.1) that trigger the streams of data between entities (in the inverse direction of the queries). While other languages exist for IoT queries are available, like SPARQL [108] and CQELS [109], we chose to slightly extend SQL to be more didactic. These other languages have specific syntaxes and semantics that will needlessly complicate our explanations. Furthermore, we map IoT data to the relational database paradigm by considering data sources, samples and attributes acting as tables, rows and columns, respectively. We then simply add the following

keywords in our query language extending SQL to describe the IoT streams:

COMPUTE indicates a list of computations to apply to the data in sequence. That way, data may be transformed with in-network processing [110]. This requirement is necessary to adjust the verbosity of data and enrich data before it is accessed by consumers.

EVENT describes one or more conditions that may activate/deactivate a stream (*e.g.*, local sensors measurements and remote messages).

EVERY indicates when to generate new data to stream. This clause describes an activation condition via events (as defined just above) or constant time intervals.

UNTIL indicates the condition to stop the stream. Analogous to EVERY, events or specific durations describe this clause. Without this keyword, the query is assumed to hold indefinitely.

In the following sections we use our extended SQL language to describe each of the queries listed in figure 3.1 in order to formally describe the information exchanged among services.

3.1.2 Ride Hailing

Mobility services that aim to provide passengers with on-demand rides from private vehicles already exist for years [111]. In terms of traffic management, this service provides a way for passengers to share vehicles, decreasing their numbers on the streets.

Recently, the portion of taxi trips handled by private cars has steadily increased. A common platform serves as broker between passengers and drivers, so that all passenger requests can be fulfilled. We detail here streams 1 and 2 illustrated in Fig. 3.1.

Query 1 – Ride selection for passengers

Passengers request multiple platforms (*i.e.*, brokers) for the best quotes to perform a ride by specifying the time of departure, and the geographical location of departure and arrival. The reply data stream includes the name of the platform and the cost to the destination (here represented by the `cost_to` function and the `$destination` parameter). The query remains active until the passenger has made a decision on one of the proposed quotes. These quotes change depending on real time availability of drivers and demand of passengers.

```
SELECT ride_platform, cost_to($destination) as ride_cost
FROM ride_platforms
ORDER BY ride_cost DESC
LIMIT 5
EVERY minute
UNTIL EVENT client_decision
```

Query 1: Passenger request sent to a set of brokers.

Query 2 – Ride selection for drivers

For the other part of the service, drivers must select their next ride. More precisely, drivers register with platforms to receive unsatisfied rides when they are about to complete their current fare. We assume here that platform implement a scoring function (`ride_score`) which ranks rides according to driver preferences, proximity, and expected payment. According to this function, the list of the best rides is constantly updated with the response stream that stops when a driver decides which quote to take.

```
SELECT ride_begin, ride_end, estimated_price
FROM ride_platforms
ORDER BY ride_score(*) DESC
LIMIT 10
EVERY 5 minutes
UNTIL EVENT driver_decision
```

Query 2: A driver asks for the available passengers.

3.1.3 Smart Parking Service

Searching for a parking spot is particularly expensive in terms of time and fuel [112]. Thus, smart parking solutions help drivers find an available parking spot, and reduce their wasted time and fuel searching for one. We can also make the system more efficient and convenient with automatic billing by having parking areas forward information on the usage of the spots to the company in charge of billing.

Query 3 – Smart Billing

Here, A vehicle is automatically billed when exiting its parking spot. We consider that a sensor is capable of detecting the occupation of one or multiple parking spots and then sends notifications whenever a vehicle enters or departs of a spot. This notification is sent to the company in charge so that the vehicle can be charged accordingly. These notifications include what event occurred (arrival or departure), timestamp of the event, and identifications of vehicle and spot.

```
SELECT event_type, time_stamp, car_plate, parking_spot_id
FROM parking_parking
EVERY EVENT park_begin, park_end
```

Query 3: Tracking vehicles entrances.

Query 4 – Guidance of cars to available parking

Drivers ask for the availability of nearby parking areas to find available parking. A filter selects areas with a sufficient number of available spots (`THRESHOLD_VALUE`), else the available spots may be occupied by the time the driver arrives to the location. Then, results are ordered by proximity and limited to prevent verbose responses. The continuous query 4 holds from the time of departure to the time of arrival (end of trip), and the list is updated every minute to account for changing availability.

```
SELECT parking_area_id,  available_count, distance(arrival_location
    , parking_location) as geo_distance
FROM parking_areas
WHERE available_count > THRESHOLD_VALUE
ORDER BY geo_distance DESC
LIMIT 10
EVERY minute
UNTIL EVENT end_of_trip
```

Query 4: Parking availability.

3.1.4 Smart Road Traffic Routing Service

Traffic jams induce increasing costs in most major cities, both in terms of time, money and pollution [2]. Not to mention the frustration caused to drivers and consumers and thus the ensuing damages it may lead to.

Vehicle routing solutions help drivers to find better routes [113] to mitigate these effects. It requires road surveillance in order to identify the level of congestion of each road segment.

Query 5 – Real-time Road Speeds

A smart mobility service may collect the location and speed measurements from a large collection of vehicles (*e.g.*, bus, private cars, taxis) to profile congestion in real time. Smart cities may exploit this information to interpolate and infer congestion in each road segment. The real identity of each vehicle should be hidden (*e.g.*, with a hash) for obvious privacy concerns.

```
SELECT hash(car_plate), latitude, longitude, current_speed
FROM consenting_vehicles
EVERY 3 minutes
```

Query 5: Road congestion information.

3.2 Open Mobility Datasets

Several datasets for IoT applications are available [107], they include traces of messages and other information regarding these applications. These traces vary depending from a dataset to another, while some may include raw network traffic [114], others will only provide sensor samples [115]. These provide insight on real life applications and may be used as a starting point when designing novel systems.

In this section, we detail how we exploit two public datasets to model the global smart mobility service we envision. We rely on this data for our evaluation of how an ITS in these realistic conditions can scale while relying on LoRa gateways.

3.2.1 Datasets Description

We exploited the two following datasets (cf. Table 3.1). These were chosen due to being from the same time period, having significant volume of data and the good

[2] https://inrix.com/press-releases/2019-traffic-scorecard-us/

Table 3.1: Datasets used in this study.

Service	Parking usage	Private-driver hailing
Location of sampling	Melbourne, Australia	New York City, USA
Number of samples	37.7 million samples	234 million samples
Sampled Period	Jan 2019 to Dec 2019	Feb 2019 to Dec 2019
Data available	Parking occupations	Taxi rides by private cars

match with the services being defined.

Ride hailing: we use a dataset of rides performed by the companies Uber, Juno, Lyft and Via in New York City (NYC)[3]. The city provides data since 2009 on trips provided by yellow taxis, green taxis and vehicles for hire. We only use data of the latter type of vehicle as to mirror our system that accounts for private vehicles (those with more privacy concerns). Each entry describes the pickup and dropoff information (*i.e.*, timestamp and geographical location) of a trip, in addition to payment information and company identification. To respect privacy concerns, the city employs pre-defined zones and zone identifiers instead of exact geographical coordinates. We model the vehicles here as mobile devices that transmit and receive information;

Parking usage: the city of Melbourne provides measurements of its large deployment of parking sensors[4]. Each sample of this dataset describes individual occupations of parking spots in the city and include timestamps of arrival and departures in addition to multiple identifiers of the parking spot. We have here static sensors with event-based traffic.

Some adjustments are necessary for these two dataset to be compatible with each other and our model. We see in the following section the pre-processing performed.

3.2.2 Dataset Preparation

The first necessary change is to adapt geographical locations since the service described by these two datasets are deployed in different cities. Unfortunately we could not find equivalent datasets of systems deployed in the same geographical area. Thus, we need to merge both to mimic a global service running in the same city. We map Melbourne's dataset locations to NYC locations since more information is openly available for the latter which would allow our system to be further expanded later[5].

We gather a list of parking lots in NYC with the aid of the TomTom API[6]. Our search unfortunately did not gather enough parking lots in NYC to perform one-to-one mapping between those in Melbourne and NYC. Consequentially, we

[3]https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page

[4]https://data.melbourne.vic.gov.au/Transport/On-street-Car-Parking-Sensor-Data-2019/7pgd-bdf2

[5]examples are further explained in the conclusion of this chapter

[6]https://developer.tomtom.com/

first cluster the 4,876 parking lots in Melbourne that we have from the dataset into 1,212 clusters (the number of parking lots we found in NYC) using the k-means algorithm according to geographical location.

With this, each cluster represents a virtual parking lot with the data from its members, and we can look for a mapping of these clusters in Melbourne to parking lots in NYC. This mapping must be made based on geographical and network traffic to realistically model the network load generated. However, this information is not available in the information gathered from NYC. Thus, we assume that parking lots closer to Manhattan (commercial center of NYC) are likely to have more demand. Which allows us to map our virtual parking lots of Melbourne with most traffic into the parking lots of NYC which are closer to Manhattan.

Then, to mimic vehicle displacement from query 5, we expand the NYC dataset by computing the trajectory between pickup and dropoff locations for each trip. For the sake of simplicity, we consider here straight line trajectories under constant speeds. In our case, an approximated geographical location is sufficient to model the traffic since pick up and dropoff are already non-exact.

3.2.3 Dataset-based Network Traffic Model

We describe here how we model the network traffic from the datasets. Table 3.2 summarizes the variables used in our models in addition to the origin of the value used for that variable.

Observe that a fair amount of the variables listed here are not defined via datasets as we could not find this information in any of available datasets. However, we argue that these variables may be empirically defined without loss of generality as future users of this framework can adjust these as needed.

Ride Hailing

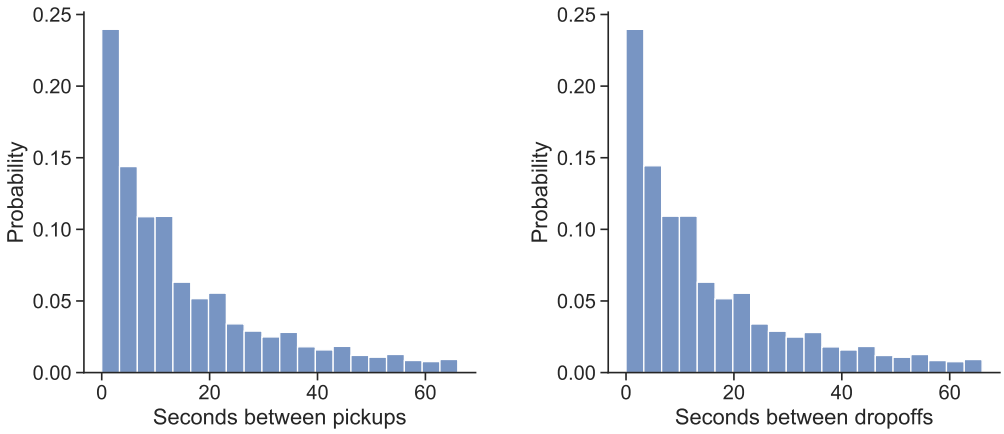
We model queries 1 and 2 based on the fact that passengers request price options right a trip and drivers request new rides right as they finish one. We use the time and location information of taxi pickups and dropoffs in NYC, respectively for these queries. Thus, each trip record triggers these queries at time and location of pickup and dropoff.

We have in figure 3.2 the distributions used to generate messages for the two queries mentioned above. Since the dataset do not identify vehicles in its entries, we measure time between events for each pickup and dropoff location to understand the message periodicity of each area. We see here that the streams of these queries have considerably frequent updates.

The other variables of these queries are not available in any dataset we have found and may be defined empirically without much loss of generality for our model. We define the duration of the query with a normal distribution and define the update periodicity (interval between updates) as constant values (see Table 3.2).

Smart Parking

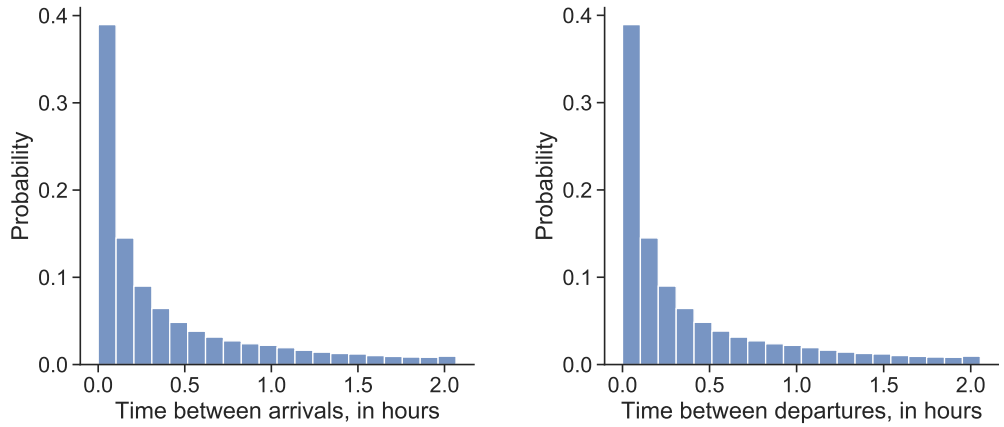
Here, arrivals and departures in parking spots trigger messages. Thus, we model the `park_begin` and `park_end` events of query 3 with the arrival and departure data



(a) Time between pickups per location (b) Time between dropoffs per location

Figure 3.2: Distributions used from NYC taxi dataset

observed in the Melbourne dataset. More specifically, for each record, at time of arrival and time of departure, one message is sent from the parking lot location.



(a) Time between arrivals per spot (b) Time between departures per spot

Figure 3.3: Distributions used for parking message generation

Figure 3.3 shows us these distributions as they appear in Melbourne's parking dataset. We also see that the resulting periodicity of sensors can scale up to hours which shows the sporadicity of this stream.

Then, we model the inter-query time of query 4 by assuming that there was a search for available parking every time a parking spot is taken (see figure 3.3a). Since available parking can change constantly, one message is sent every minute until the car parks. Similarly to the queries in the previous section, we again rely on a normal distribution to model the time to find a parking spot and the update periodicity (see Table 3.2).

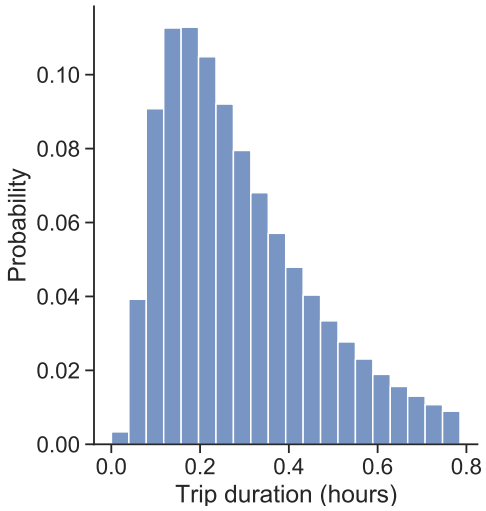


Figure 3.4: Distribution of duration of trips in NYC dataset

Smart Road Traffic Routing

In query 5, we consider that a subset of vehicles sends its speed information (*e.g.*, a subset of vehicles is equipped for it or that consents to have its data used). More precisely, we consider that only 10% of the dataset of taxi trips is considered by this sub-service. Messages are sent periodically along the trajectory we generated for each trip.

From figure 3.4 shows us that most trips are between 10 and 20 minutes which makes so that most streams transmit less than 6 measurements with the chosen periodicity (3 minutes).

3.3 Scalability Analysis with a LoRa deployment

We discuss here whether a LPWAN technology like LoRa could be used for such Smart Mobility services. LoRa was chosen for this study as it is one of the leading technologies and a simple simulation model was available through [116]. We assume that LoRa will be the only means of communication for queries the defined queries and their answers. We have considered including NB-IoT or Sigfox, but we decided not to extend it too much since the evaluation is not correlated with objective of the thesis. The next section provides a quick primer on LoRa and details how we simulate them.

3.3.1 LoRa Analysis Methodology

LoRa is a cellular network where devices connect to base stations (*aka.* LoRa gateways). Traffic is assumed to be mainly uplink as low-end devices push data to the base station in order for it to be forwarded to its destination. Its signal design is based on Chirp Spread-Spectrum modulation technique, that uses pulses to encode information in order to improve the robustness and resilience of the signal. The spreading of the signal is a key feature in this technique. Transmissions on different

Table 3.2: Query variables.

Query	Variable	Value	Source dataset
1	Location of query	Trip pickup locations	NYC
	Time of query	Trip dataset pickup times	NYC (Figure 3.2a)
	Update periodicity	1 minute	–
	Passenger decision	Normal (μ=1, σ=1) min	–
2	Location of query	Trip dropoff locations	NYC
	Time of query	Trip dropoff times	NYC (Figure 3.2b)
	Update periodicity	5 minutes	–
	Driver decision	Normal (μ=5, σ=3) min	–
3	Location of query	Parking locations	Mapped NYC parking (Section 3.2.2)
	Update periodicity	Parking arrival and departure times	Melbourne (Figs. 3.3a and 3.3b)
	Time of query	Beginning of times	–
4	Time of query	Parking arrival times	Melbourne (Figure 3.3a)
	Update periodicity	1 minute	–
	Limit	10	–
	Trip duration	Normal (μ=8, σ=3) min	–
5	Vehicle Trajectory	Straight line between trip pickup and dropoff locations	NYC
	Trip durations	Taxi trip duration	NYC (Figure 3.4)
	Periodicity	3 minutes	–
	Vehicles subset	10% of available cars	–

Spreading Factor (SF) have different characteristics such as air time, bit rate and reception range. Pairs of transmitters/receivers have to select a spreading code, also designated as a SF in LoRA in order to communicate.

As displayed in figure 3.5, devices may be able to transmit to base stations up to a dozen kilometers away from it and take advantage of bit rates in the order of kilobits per second. By selecting the right SFs, devices can use the highest bit rate adapted to the necessary signal strength. More importantly, the SFs are mutually orthogonal, that is, transmissions on different SFs never collide. However, handling a very large number of devices (even generating only a few packets each) with few antennas is challenging [118]. Collisions are frequent, and network capacity may be rapidly reached.

To conduct our scalability analysis, we consider the LoRa analytical model described in [116]: for a given amount of network traffic, it predicts the PER of each

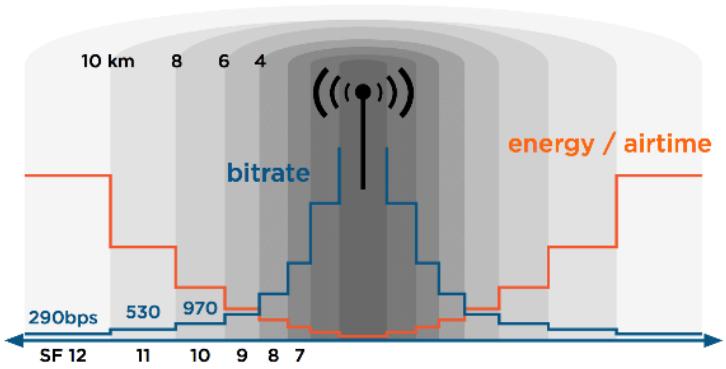


Figure 3.5: Lora spread factor characteristics [117]

device associated with a LoRa gateway. It measures it by calculating the chance of two devices overlapping transmissions while also accounting for the possibility of retransmitting the packet in case of collision[7]. In our calculations we use the EU 863880 MHz ISM band, with the receiver sensitivity of the Semtech SX1276 LoRa receiver[8].

Since the services we study consist of dense urban deployments, devices need to be close to their LoRa gateways (else, the network capacity is reached too fast). In other words, if gateways serve an area too large, they will not be able to adequately serve devices. The question we want to address is: if feasible, how and how many, Lora gateways should be deployed such that the link quality between devices and gateways does not turn into a limitation after a certain degree of density.

We employ two different methods of SF allocation:

Distance-based: some devices are too far from the LoRa gateway, and we must consider their signal strength when selecting a SF. Thus, we allocate to each device the lowest SF with respect to its signal strength. A lower SF means less signal robustness and higher bit rate and shorter airtime, thus this method greedily chooses the SF with higher data rate available for the perceived signal strength.

Airtime-based: with very dense deployments, devices are very close to gateways and signal strength is not a constraint anymore. In that case, we balance the load for each SF with respect to airtime and equally distribute the probability of collisions among SF. More precisely, we assign the data rates so that all the data rates have the same cumulative air time. The devices with the largest SF consume more energy to stay awake longer, but we reduce the overall number of collisions.

We use the density of base stations deployed 5 kilometers apart as the threshold to use the airtime-based allocation, *i.e.*, if devices are closer than 5 kilometers from the base station, they use it, otherwise it will use the other method. We indeed

[7] All formal details are given in [116].
[8] https://www.semtech.com/products/wireless-rf/lora-core/sx1276

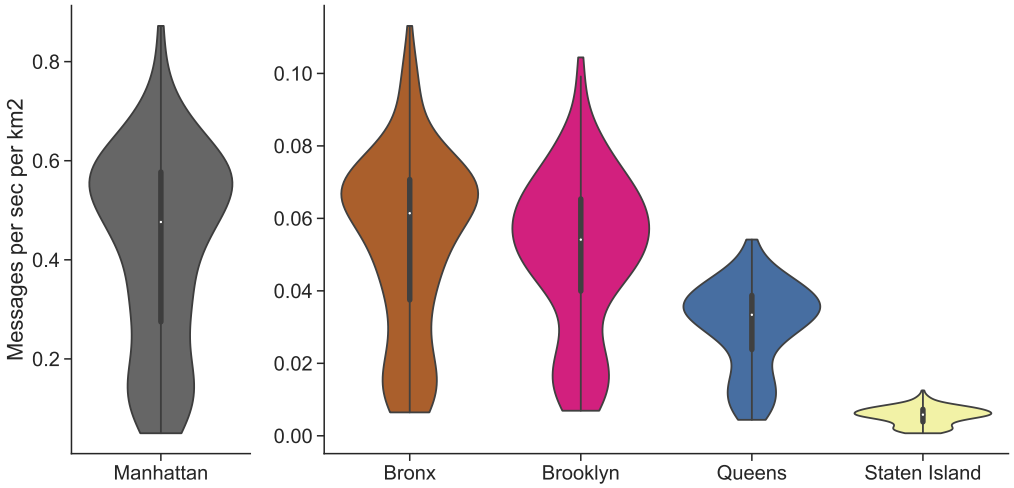


Figure 3.6: Message density per neighborhood.

observed that at this point, devices are close enough to the base station to use airtime-based allocation.

To further simplify our study, we also assume that any message can be sent within a single LoRa transmission. The payload of LoRa packets may go up to 256 bytes which is plenty for the messages in question. Furthermore, compression techniques can be applied to ensure that messages fit in this limited space [119].

We can now rely on the ITS we design along with the real datasets introduced in sections 3.1 and 3.2. To account for time fluctuations in network traffic, we pick 200 random time intervals of one hour, and use the average number of messages per second during each of these intervals. More specifically, for each randomly picked interval, we count the number of messages sent by each device and infer the average number of messages sent per second by each device. By considering independent enough time samples, we are able to investigate the scalability of LoRa for these smart mobility services.

Then, we position our base stations in a grid format. Machine learning models have also been proposed in order to discover optimal base station positions [120, 121, 122, 123], which are interesting for scenarios with non-uniform traffic. But for this work we consider uniform grid deployments for LoRa gateways for a simpler deployment that maximizes coverage. To assess the scalability of such LoRa deployments, we investigated several grid sizes by varying the density of base stations.

Overall, we consider each borough separately since each area generates different amounts of traffic, which allows us to focus on the geodistribution of traffic.

3.3.2 Results: Sufficient Adaptation of Deployment Density

We measured the intensity of traffic per mobility service and per borough in *figure 3.6*. The graph presents the distribution of the average number of messages per second for all 200 time intervals. The violin plots are stretched out as the number of messages obviously varies a lot between peak hours and off hours. Besides, if we count the sum of messages, we can see that Manhattan (only 7.5% of NYC area)

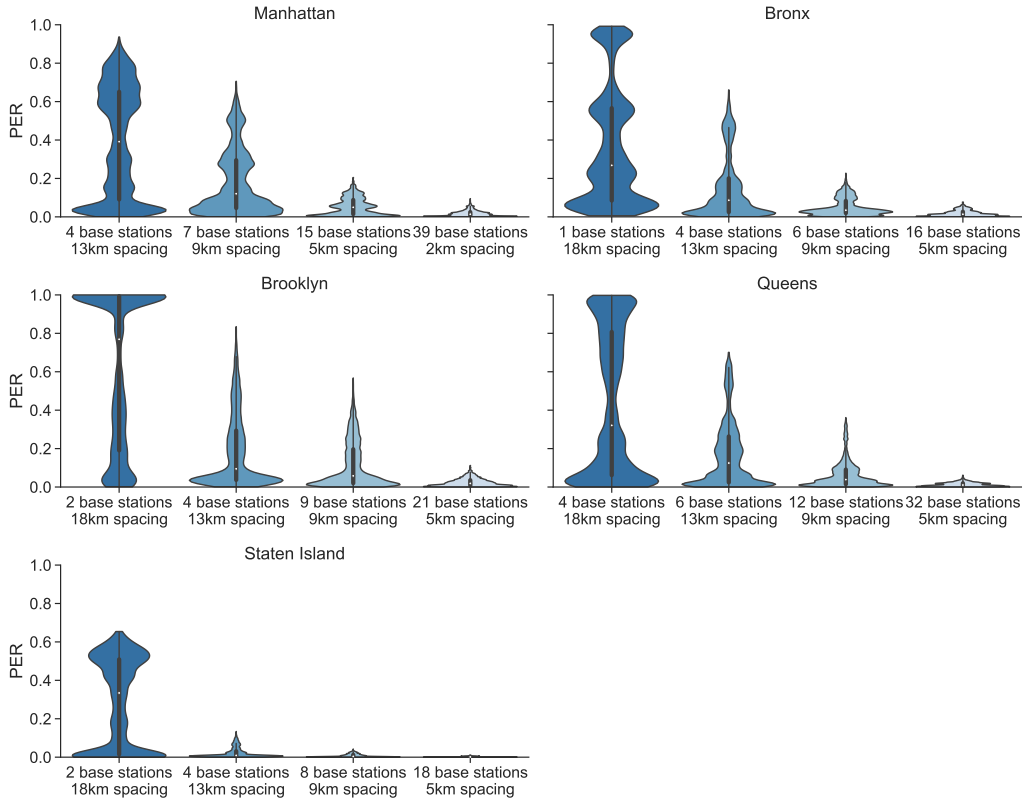


Figure 3.7: PER of streams under different base station densities.

concentrates most of the traffic (notice the different scale on the vertical axis) while others have much less variation among them. For instance, 65% of ride haling messages are generated in Manhattan while the other boroughs only stretch over 7.5% of NYC.

Relying on this data, we can now compute the distribution of the PER independently for each borough (*Figure 3.7*). While each gateway is theoretically capable of serving nodes that are up to 9 km away, the traffic is too high and the resulting network becomes congested. We must place more base stations in the same area to provide a reliable service.

For instance, we can see that two base stations are not enough to efficiently cover Staten Island. However, 4 base stations prove enough to cover the whole area, and to also decently limit the number of collisions. Similar patterns can be seen for other boroughs, except for Manhattan.

For Manhattan, the volume of packets being specially large, many collisions occur. We see that only very dense deployments of LoRa gateways are capable of providing an acceptable PER. As a direct result, devices are very close to the LoRa gateways, and the lowest SF may be used by all devices, creating congestion if all devices are configured to use it. Thus, we need a more balanced allocation among the different SFs, which motivates a further evaluation of the two SF allocation schemes.

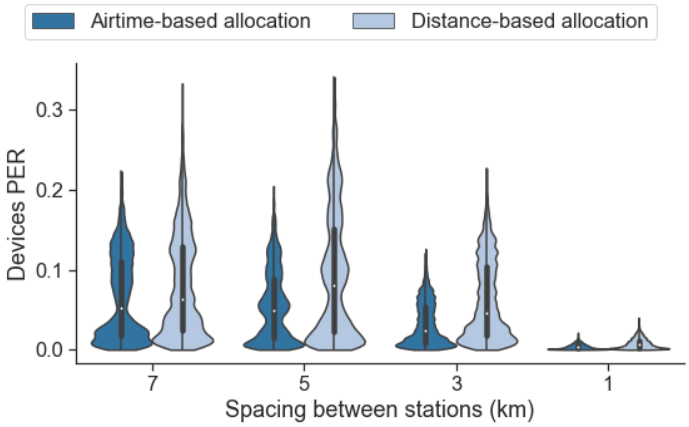


Figure 3.8: Impact of the SF allocation strategy in Manhattan.

Figure 3.8 shows how our two allocation strategies behave under various network densities. For very dense deployments (*i.e.*, one gateway every kilometer), the number of devices associated with a given gateway is quite low and collisions seldom occur: the PER is minimum. For all the other densities, the airtime-based strategy is more efficient: devices are well balanced among SFs, and the system minimizes the number of messages lost (*i.e.*, PER).

Finally, we look at the geographical distribution of the PER value (*Fig. 3.9*). More precisely, for each borough, we select a grid size to obtain for each device an acceptable reliability. Thus, we consider 5 km spacing for Manhattan, 13 km spacing for Staten Island and 9 km spacing for all other boroughs to achieve the PER displayed in figure 3.7 for each area. We represent the results with a quad tree where the size of each square is selected so that each square has the same number

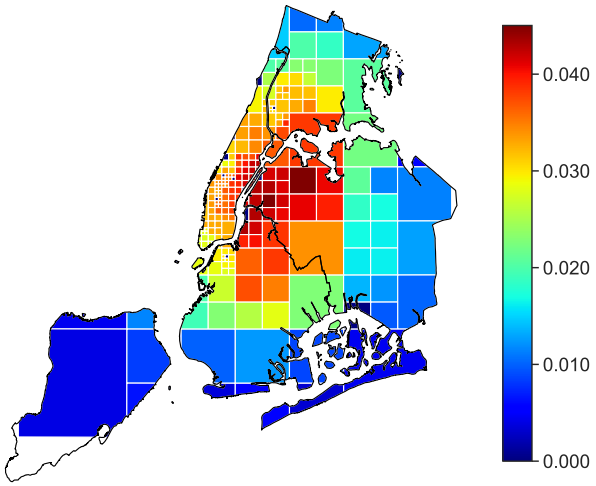


Figure 3.9: PER under mixed deployment.

of devices. Then, each square is colored according to the average PER obtained by the corresponding devices. As we can see, the resulting PER reaches a maximum of 4% in the center of the map: squares are smaller, denoting a huge number of devices that generate traffic. On the contrary, sparse grids are enough for Staten Island, since the traffic is very small in regard to Manhattan. If the PER has to be decreased even further, a localized optimization scheme should be used to deploy extra base stations in the congested hot spot areas (red ones).

3.4 Conclusion & Perspectives

We defined here a global smart mobility scenario, where several services cohabit. We then emulated the traffic generated by such scenario by using publicly available datasets, collected respectively in NYC and Melbourne. In particular, we consider both event-based traffic from static devices and periodical traffic from mobile ones. Finally, we analyze how a LPWAN deployment (LoRa in practice) can be provisioned to support these mobility services. While our scalability analysis tends to indicate that relying only on LoRa is enough and so conceivable for a unified ITS deployment, we note that the number of LoRa gateways should be carefully tuned according to the traffic. Overall, the network capacity is reached fast, and can create scalability issues for large-scale scenarios.

It is interesting to explain that we initially experimented with a non-uniform (non-grid) method of base station allocation to account for message generation hotspots. Let us use the following notation to explain this other method:

- $PER(L)$ as the predicted packet error ratio of a LoRa gateway with load L
- c_i as the coordinates of LoRa gateway i
- L_d as the network load of device d
- D_i as the set of devices which LoRa gateway i is the closest

Then, we try to acquire the best LoRa gateway positions for each of the picked periods and a given number of gateways n using:

$$\underset{c_1,\ldots,c_n}{\arg\min} \sum_{i}^{n} PER\left(\sum_{d\in D_i} L_d\right) \tag{3.1}$$

We obtain consequently the n optimal positions for each time window. We input this into the BFGS minimization algorithm [124] by using the concatenation of the list of coordinates $c_1, \ldots, c_n$ as the parameters input vector. Then, since each time window may result in different positions, we have to find n positions that would be good for all time windows. For this purpose, we use k-means to find the patterns in the placement of LoRa gateways for each number of LoRa gateways. The centroids of each cluster of base stations found by k-means provide us with a position that should work well even with the variation traffic.

Unfortunately, this method is not working well due to our minimization not converging properly. Even after a very large amount of iterations and hours of

processing, the overall PER observed at each base station was very large, which makes the problem untractable. Thus, we decided to abandon this non-uniform deployment method and use simpler grid deployments as the objective for this thesis is not to investigate deployment strategies.

In a future work, it would be interesting to investigate how to refine the placement of the LoRa gateways. We propose here a simple heuristic, using a grid density adaptive to each borough, but we should go further by identifying more accurate and favorable positions for base stations, depending on the location of hotspots. Unfortunately, the location of gateways being a continuous 2D variable, the global optimization becomes almost intractable. It is worth to investigate if it would be enough to simply discretize the deployment space and create a specific algorithm to minimize overall PER by iteratively move base stations in the direction of hotspots. Perhaps a genetic algorithm would adapt well here, similarly to the one proposed in [123]. One would also do well to include collisions among base stations in the LoRa model used to evaluate placements, at the time, we have not included it do to limitations in the time allocated for this contribution.

We may also explore how different technologies (medium vs. long range) may be complementary in such situations, handling differently static and mobile devices. Typically, smart parking concentrate many devices in restricted areas that can result in congested hotspots and a localized short range technology would perform better here.

The insight from this work exposes the needs of multiservice and multi-owner scenarios. We have to pay attention to:

- the requirement of decoupling producers and consumers of data in order to enforce privacy;
- the need to process data as it is disclosed to enforce that it is private enough; and
- the necessity for efficient transmission protocols to handle the large volume of data.

Regrettably, we do not use this framework in the evaluations of our following proposals. We rather focus on larger scale infrastructures and this is too limited to stress our proposals. Thus, we chose to take more generic and random topologies of connections among services, so that we are able to scale and tackle more complex structures.

Our next proposal is an architecture that enforces privacy enabling computations as data is disseminated among services. While the queries among services include transformations by design, our architecture enforce the anonymization of data and offer in-network processing.

Chapter 4

NDN Overlay for Privacy and Reusability in Multi-Domain IoT

Contents

As complex IoT systems rely on services from several owners, these may accept to share *some* data to other trusted systems. While networks with different owners can cohabit in the same shared infrastructure (multi-owner infrastructure) they must select the data they want to share [125].

Enabling solutions rely on complex access control mechanisms to ensure to export data and prevent leaks [126]. Unfortunately, maintaining credentials for consistent access control rules is very challenging in multi-owner applications [127].

Additionally, since these IoT streams shared among different services can account for a large volume of data, there exists an opportunity to filter and pre-process it directly in the network to mitigate the overall network load while still offering

the same application benefits to consumers. Such as when customers desire simple operations like applying an average function, we may apply these operations inside the network rather than letting customers perform them on their own [39].

In this chapter we tackle the problem of finding a method to bring scalable and privacy-aware communication to large scale IoT infrastructure composed of devices from multiple owners deployed for different objectives. We propose a virtual interconnection (overlay) of multiple IoT networks based on provider/client relationships where NDN routers take charge of handling communication among networks. These exchange anonymized datasets on the basis of a simple policy-based inter-system communication protocol.

Such policies define control plane rules that describe which data stream can be exported and how it has to processed first. We define how to implement in-network transformations in the data plane which we assume to make data private enough to disseminate it. These transformations are expected to be functions such as filtering of sensitive information, averaging of samples and noise addition.

Our method provides privacy without the complex ciphering mechanisms usually employed in such scenarios. Privacy is rather enforced by the imposition of owner-defined transformations that make data anonymous enough to be disclosed.

We implement our transformation based NDN overlay in NS3 to evaluate its benefits and performance. In particular, we show how the opportunity to transform, aggregate and reuse popular data allows improving scalability while enforcing privacy by design.

In the following section we more formally define concepts that aid us in the definitions of the problems and solutions we discuss in the remainder of this thesis.

4.1 Domains, Queries & Aggregation

IoT networks can naturally be delimited by virtual boundaries defined by criteria such as ownership, application, and deployment objective. For instance, we may delimit velocity meters of a smart city by the company in charge of its management and/or by road segments. We denote these groups of devices as ***domains***.

4.1.1 IoT Domains

The following properties illustrate well how networks can be delimited in practice:

Geographic: devices located within the same room / building / neighborhood;

Application: devices in charge of a given system. For instance, it can be a parking lot grouping a set of presence sensors, lights, security locks and the controller;

Owner: devices that share the same owner and thus are managed by the same entity which has complete access.

According to this definition, any device may freely exchange data since data is of the same owner, or it was given consent to be used by that application. Indeed, data can be exploited by any application part of a domain. Thus, we do not have any privacy issues with data transiting within a domain.

Two main benefits are brought by an infrastructure partitioned in domains:

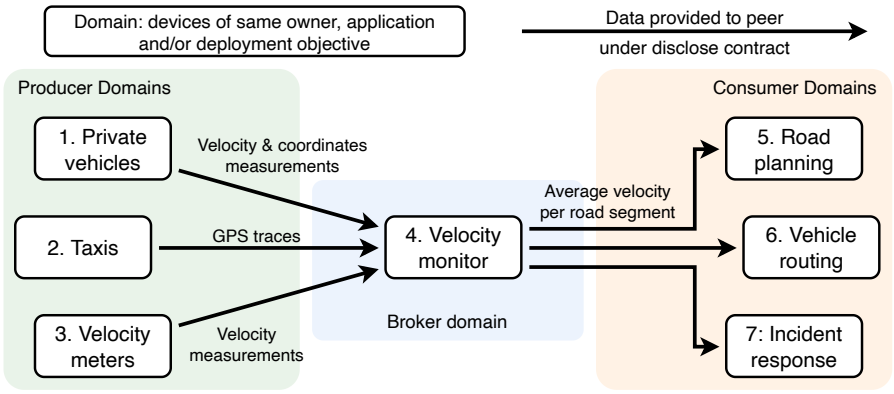


Figure 4.1: Smart city transport example

Reuse of infrastructure sensors can supply multiple applications, resulting in less redundant deployments;

Data diversity applications can take advantage of data from multiple owners to answer make decisions with a much broader view.

Moreover, the emergence of ***broker domains*** that can distribute data they collected (locally or from others) leads to the detachment of consumers and producers. We expect that large trees of brokers may form as these collect data from each other to answer queries.

Take for instance the logical topology illustrated in *Figure 4.1* derived from our model in Chapter 3. Domain 4 (velocity monitor) is a ***subscriber*** for the data streamed from the three domains on the left, and ***publishes*** an aggregated stream for the three domains on the right. Here, data can be gathered from available velocity meters, but in areas where they lack, vehicles such as taxis and private vehicles (upon agreement) can be used to produce information on current road speed. Taxi companies are known to keep location traces of their vehicles in order to keep track of their usage [128] and private vehicles may agree to offer sporadic location and velocity samples.

Different taxi companies can be seen as different domains and private vehicles can also be seen as individual domains. Once collected, this data can be further shared to other domains such as road planning, vehicle routing and incident response in order to, respectively, plan road modifications, direct traffic to areas with faster flow and detect collisions and other incidents.

While *some* information need to be exchanged among the different domains, it is critical to respect the privacy concerns of each domain. In our example, private vehicles provide individual velocity and location information instead of complete GPS traces like taxis provide.

To define how information is exchanged among domains, the following subsection explains the type of queries issued among them.

4.1.2 Query Model for Inter-Domain Data-Streams

In our model, a consumer is not interested in the data of specific producers, instead, it is interested in a type of data. Thus, a query describes the desired data based on information such as type (*e.g.*, temperature, velocity, wattage), cardinality (the number of samples and sensors), frequency and location. For example, a query may consist in asking for the humidity of soil in farms of a specific region, or in the power consumption from at least 100 households every day. Such a description is referred as the ***metadata criteria*** of a query.

When a domain receives a query, it has to find data that match its criteria. That may involve the data from multiple producer devices, and in our case, it may include the data from producers in different domains.

Additionally, consumers normally require preprocessed data instead of the raw sensor samples. The desired computations normally include some form of aggregation function, *i.e.*, a function that summarizes information on a group of values, as a means to lower the verbosity of datasets. Descriptive statistics such as average, minimum, histograms and density functions are examples of common aggregations that strength privacy [39]. Queries might also include other operations such as encryption, noise addiction, and filtering. We will use the generic term ***transformation*** to denote these computations.

As a typical query demands aggregated values (both spatially and in time), the domain handling it is both in charge of retrieving the data and applying such transformation to provide the desired granularity to consumers. For instance, the energy consumption may be averaged monthly, and this simple transformation should be applied as close as possible from the producer to save bandwidth.

Indeed, it is not only about scalability (by transmitting the resulting value, the network load decreases), aggregations also enable privacy by masking specific samples. When possible, a domain may apply such operations by itself or trust others to do it.

We now take some time to discuss more formally the datasets and aggregation functions that we consider in this thesis.

4.1.3 Datasets and Aggregation Functions

Aggregations are operations that provide us information on the population of a random variable through multiple samples. We model these datasets as multisets denoted $\mathcal{M} = (A, m)$ where A is the set of distinct data values observed and the multiplicity function $m : A \to \mathbb{N}^+$ denotes the number of times the multiset contains each value in A. We also define the sum operator to represent the merging of two datasets as $M_i + M_j = (A_i, m_i) + (A_j, m_j) = (A', m') = M'$ such that $A' = A_i \cup A_j$ and $\forall x \in A', m'(x) = m_i(x) + m_j(x)$. Notice that merging two datasets is an anonymous operation since values in M' have no indication whether it originally is from M_i or M_j.

We assume here that all datasets have a set of descriptive attributes. Take for example the dataset of parking sensors deployed in Melbourne to monitor available parking spots (see section 3.2.1). This dataset includes geographical location of more than a thousand sensors that monitor bays of multiple parking lots. We denote as $meta_p$ the set of metadata of the dataset offered by producer p.

Table 4.1: Example of aggregations functions (minimum, average, and histogram functions)

$D1$		$D2$		$D1 + D2$	
$x \in A$	$m(x)$	$x \in A$	$m(x)$	$x \in A$	$m(x)$
3	2	7	3	3	9
4	1	8	5	7	6
7	3	3	7	4	1
8	4	1	2	8	9
				1	2

$min(D1 + D2)$		$avg(D1 + D2)$		$hist(D1 + D2)$	
$x \in A$	$m(x)$	$x \in A$	$m(x)$	$x \in A$	$m(x)$
1	27	5.4	27	1	12
				5	12
				10	0

We say that a function $f : \mathcal{M} \to \mathcal{M}$ preserves the sum operation of multisets iif $f(f(\mathcal{M}_i) + f(\mathcal{M}_j)) = f(\mathcal{M}_i + \mathcal{M}_j)$. Commutative and associative functions satisfy such a preservation of the result. Thus, applying such a function to merged or unmerged inputs (and whatever their ordering) does not change the final value.

Finally, we denote an aggregation function as such if it preserves the sum operation and also conserves the cardinality of the resulting dataset while decreasing its level of detail. More specifically, for a given aggregation function f and a set of n input datasets $\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_n$ such that, for easier notation, $\sum_i^n \mathcal{M}_i = \mathcal{M} = (A, m)$ and $f(\mathcal{M}) = (A', m')$.

$$\sum_{x \in A} m(x) = \sum_{y \in A'} m'(y) \text{ and } |A| \geq |A'| \tag{4.1}$$

Average, maximum and minimum are simple statistical functions that match this definition of aggregation functions with $|A'| = 1$. Such operations result in a multiset with a single value, *i.e.*, $f(\mathcal{M}) = (A', m')$ such that $|A'| = 1$ and $m'(y \in A') = \sum_{x \in A} m(x)$.

Table 4.1 shows examples with two datasets $D1$ and $D2$. Observe that both multiplicity functions resulting from the minimum (min) and average (avg) functions indicate the number of input samples used as required in equation 4.1. This displays the number of samples initially used and hints to the statistical significance of the result. In the case of the average function, it is also necessary in order to correctly average two already aggregated datasets by using the cardinality as weights.

Such functions may however have different granularities as data can be aggregated according to available attributes. A monthly average (timestamp-based aggregation) would, for example, result in $|A'| \leq 12$, *i.e.*, at least one value for observed months. With this method, one can indeed extract controllable approximations describing the most basic statistical indicators such as the standard deviation,

median, quartiles or any of such descriptive information. We present the histogram (*hist*) function in table 4.1 as an example of such a function. Here, we implement this function with the value in the resulting set is the lower bound of the bins of the histogram and the value of the multiplicity function is the number of samples in the given bin.

Our objective here is just to exemplify the implementation of such functions. It also serves to illustrate the need to maintain the sum of cardinalities as defined in equation 4.1. In addition to the need to quantify the statistical significance of an aggregation, we see that the implementation of these functions require it. For the average function instance, applying the average function must account for the values in m in order to correctly weight the number of samples in an aggregated dataset.

The explanations of this section are the basic building blocks that we use for future definitions of the problems we tackle in the remainder of this thesis and also the solutions to these problems.

4.2 Problem Statement: Privacy Aware Inter-Domain Streams

We assume that ***domains are curious but honest***, *i.e.*, they may read and take advantage of all information they have access but will not stray from the protocol. We consider this model as adversaries that modify or falsify data can be handled by the NDN authentication and integrity verification that we use in our proposal. Additionally, domains that stray from the protocol by leaking data indiscriminately can be identified by watermarking operations [129].

Encryption would be the direct mean to hold back these curious domains. First an access control system would have to be put in place to grant access to data. That in itself would be a burden as this would have to be implemented through a trusted third party [130, 131] or through a distributed database [89] which would respectively grant power and information to an entity or would result in excessive public information. Then, in-network aggregation would force domains to decipher, apply transformations and finally recipher the result. Homomorphic encryption would solve this issue, but we disregard it since it is limited to some transformations [54].

Thus, we seek to provide privacy without relying necessarily on encryption. Domains apply transformations instead in order to make data private enough for neighbors to access. And then, additional transformations are required to disseminate further the data to untrusted domains.

Transformations akin to aggregations and filtering can be exploited in order to provide levels of privacy similar to that of anonymous datasets. Anonymous data strategies rely on a tradeoff between utility and risk: aggregation decreases the level of information but also makes de-anonymization harder [132].

We assume that peering domains trust each other, *i.e.*, one domain can expect its peers (neighbors) to respect the privacy policies they agree on. This trust can be enforced with tools like watermarking [129] to verify that their peers correctly respect defined policies. When a leak is detected, watermarking indeed allows identifying the faulty peer in the chain. That is why we expect that watermarking

may typically be part of the critical and minimal set of pre-transformations applied before sharing any data stream.

In the following sections we detail our proposal for an architecture based on NDN to provide multi-domain communication while respecting transformation requirements among domains. This structure both provides interoperability via border routers that work on the same protocol and provide privacy by enforcing transformations in streams among domains.

4.3 Proposal: A NDN Multi-domain Architecture

To enable the exchange of datasets between multiple domains, we propose to construct an overlay of ***border routers***. These routers are in charge of defining what data can be exported to respect a set of privacy requirements (defined by the owner or administrator of the domain). The overlay connecting them is a logical topology, built on the top of the trust relations between domains. Typically, a border router defines which data it accepts to share with its peers: it defines the ***exporting policies*** attached to each of the dataset it exports. We model privacy requirements (*aka.* exporting policy) as a collection of (anonymization) transformations that must be applied to the exported dataset. For instance, a border router may filter and aggregate a dataset before it exits the domain. It consists, respectively, in removing personal data that may can lead to some kind of identification and sending only the aggregated value to obfuscate precise data. We also make a distinction between direct peers, that trust each other, and the rest of the network, reached transitively, where additional transformations may be enforced.

Note that transformations are at the core of our multi-domain architecture. Aggregations specifically not only hide individual data streams to respect privacy

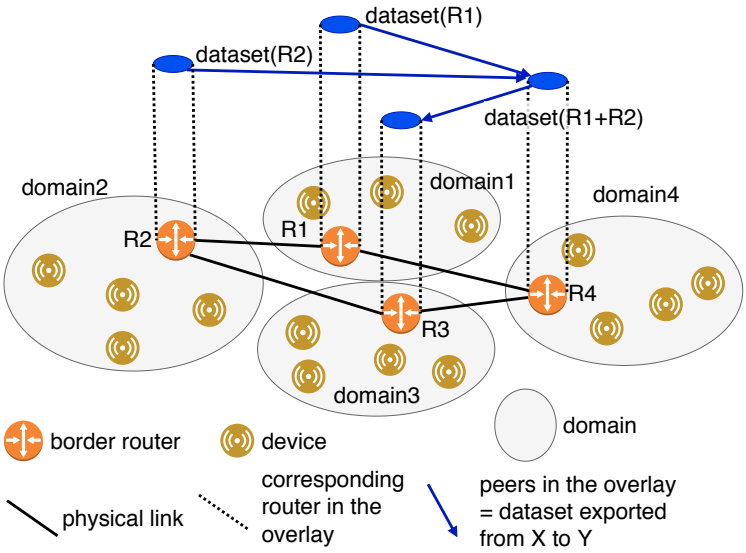


Figure 4.2: Overlay on top of multiple domains. This overlay forms a logical tree for a given dataset (here customers in domain4 can directly access the entire set of producers).

but also allows for scalability. Let us consider *Figure 4.2*: each domain has a border router in charge of exporting and importing data from/to other domains. The overlay takes shape after the relationships among domains as they provide data to each other, *e.g.*, domain 4 is a consumer for domain 2, and a producer for domain 3. Here R4 aggregates the data from two domains (domain 1 and domain 2 being here leaves of the tree).

In this chapter we assume that domains form trees in the overlay. A tree structure prevents by design the occurrence of loops in the acquisition and transformation of data. Indeed, a mesh topology may result in inconsistencies such as averaging the same data multiple times. *We tackle mesh topologies in the next chapter.*

A tree is enough to capture well existing commercial relationship among domains. Indeed, domains will acquire data from other domains which likely bough data from other domains, thus forming tress as this repeats. For example, smart homes send their electricity consumption to their electricity provider, that re-sells the anonymized dataset to a broker. Finally, the broker aggregates the data from different electricity providers to sell a large and diverse dataset to R&D / marketing / commercial entities.

We choose to implement this overlay while relying on the Named Data Networking (NDN) paradigm. This clean-slate network stack is particularly relevant for IoT applications as a NDN router manipulates chunks of data and not any more opaque packets (see Section 2.1.4). Data can be cached in the network to reduce the volume of packets to transmit when multiple consumers are interested in the same popular data [60]. Typically, two or more consumers subscribing to the same dataset exported by a domain, will receive the same continuous flow of data in a multicast fashion. We also expand NDN cache mechanisms to detect overlapping dataset. Indeed, the sequence of transformations for two exiting data streams may be partially overlapping. For example, if one query requests the monthly average of the year and another requests for the monthly average of the first trimester, the data used to answer the first query can simply be filtered in order to answer the latter. In that case, the border router detects the overlaps and uses its Content Store to save transmissions.

Our overlay defines the underlying control plane of our NDN routing scheme. When a query (NDN interest) arrives at a broker, this broker is likely to request more data to answer it, and thus a fork occurs as the broker issues queries to acquire this data. For example, when the necessary number of samples (the dataset cardinality) cannot be achieved considering only local data. We let the definition of more flexible and sophisticated routing schemes, using graph construct less strict than a single directed tree per dataset, for our next chapter. The main challenge being about avoiding data duplication without directly identifying the producers.

Next, we detail how the different building blocks of our solution interact. In particular, we explain how the data is transformed during the forwarding process to preserve the privacy constraints while improving the network performance.

4.3.1 Border Routers

In each domain, at least one border router is in charge of defining what data may be exported, where and how. In other words, each border router is a door for any data stream to/from the outside. It represents the key location to control privacy but also to enable re-usability. To fulfill its functions, each border router maintains peering (point-to-point) connections with a selected set of other border routers, which are part of different domains, *i.e.*, its ***overlay peers***.

To favor flexibility and enable the incremental deployment of relations between domains, border routers form an overlay on top of the physical network, as illustrated in *Figure 4.2*. This NDN overlay is a logical topology deployed on top of the physical one. Border Routers can use any secure tunneling mechanisms for domain inter-connection.

First, a domain decides whether it aims to expose a portion of its data to its peering border routers based on it's service needs, commercial contracts, etc. In other words, it accepts (or not) to reply to NDN interests that it receives from its peers. A peering border router can aggregate several datasets to create a novel one, exported to its peering domains. Globally, this overlay forms a directed tree that may not match the physical topology. In Figure 4.2, an arrow ($A \rightarrow B$) in the overlay represents the dataset exported between domains A and B. The direction of a link describes the relationship between domains: A provides a (possibly aggregated) dataset with a given semantic, granularity, and cardinality to domain B. For instance, let us assume that R2 and R4 are peers: while they are connected with a tunnel which consists in the physical path ($R2$, $R3$, $R4$) (one of the two best shortest path between them), $R2$ provides its dataset to $R4$ that can then share it with $R3$ (along with the dataset of $R1$ – having the same semantic).

To summarize, a border router executes the following tasks:

1. it collects data from its domain, according to internal protocols, and only externally exposes parts of streams that the owner of the data desires to export;

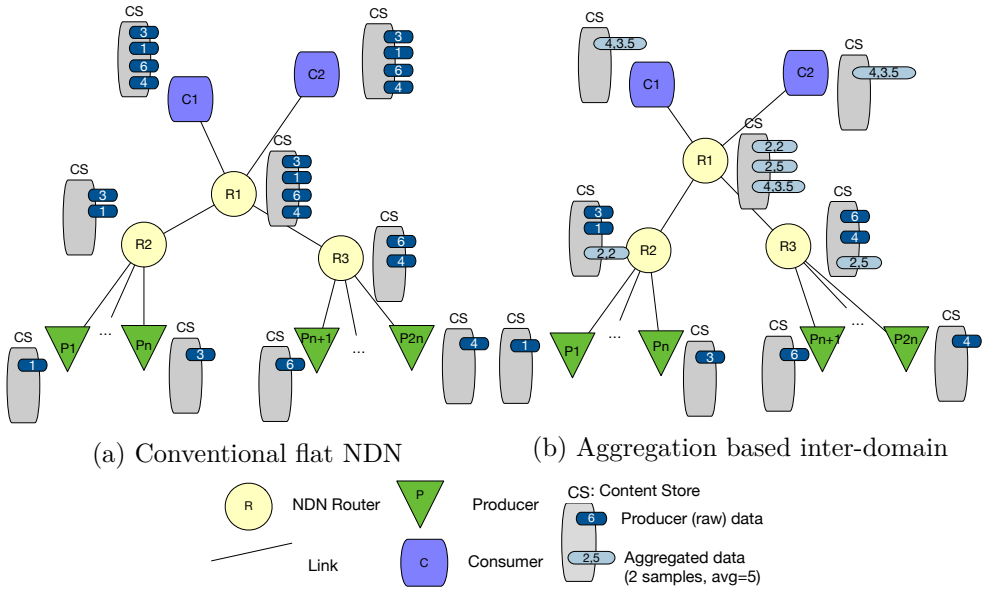
2. it verifies that the interests it receives are compliant with the defined policies. Else, they are silently dropped;

3. it constructs a reply, possibly aggregating and transforming data before it exits the domain. It uses directly its content store if the data is present, otherwise relaying the sub-interest(s) to its peers.

In the next section we discuss how to illustrate how this overlay can be used to aggregate and transform data.

4.3.2 Illustrating the Capacity of Aggregation

Let us consider a simple scenario to compare a flat approach (*Fig. 4.3a*) to our inter-domain aggregation model (*Fig. 4.3b*). The figures here show the content store (cache storage) of NDN that records all data that is passes through routers. From it we can see the data that is forwarded and that is



(a) Conventional flat NDN (b) Aggregation based inter-domain

Figure 4.3: Topology with 2 consumers and $2 \times n$ producers.

In a flat NDN approach, each producer sends its raw stream directly to the consumers. This strategy is efficient only if several consumers (here $C1$ and $C2$) are interested in the same data, *e.g.*, if several consumers are monitoring the same sensors. An intermediary NDN router (*e.g.*, $R1$) may have already the popular chunk of data (measurements) in its cache to reduce bandwidth consumption. However, consumers need to apply by themselves the aggregation function relevant for each of their applications. It also disseminates raw data throughout the infrastructure, which is prejudicial to privacy if not ciphered since intermediate routers may take advantage of their contents.

On the other hand, *our aggregation-based strategy* processes the datasets directly inside the network. For instance, $R2$ is both a consumer and a router as it consumes the streams from its domain, and generates the average value (2) over two samples; this dataset is then re-exported to $R1$. $R3$ behaves similarly, only exporting aggregated data. As with the flat approach, the data can also be cached efficiently, but here Content Stores contain only the aggregated values, not each sample individually. While less flexible (data cannot be reused for queries with other transformations) it not only brings scalability but also privacy. Final consumers never access and process the individual raw data and cache sizes are much smaller.

Figure 4.4 provides an overview of our solution. It shows the different engines that our border router uses to process a query. Here, a consumer in domain 4 creates an interest to a dataset offered by domain 3 that is the result of the aggregation of the data from domains 1 and 2. The NDN routing engine (see section 2.1.4) is responsible to find the dataset given its name. Then, the policy engine enforces the transformations that are applied by the transformation engine. In the following sections e detail this figure and the engines of our architecture.

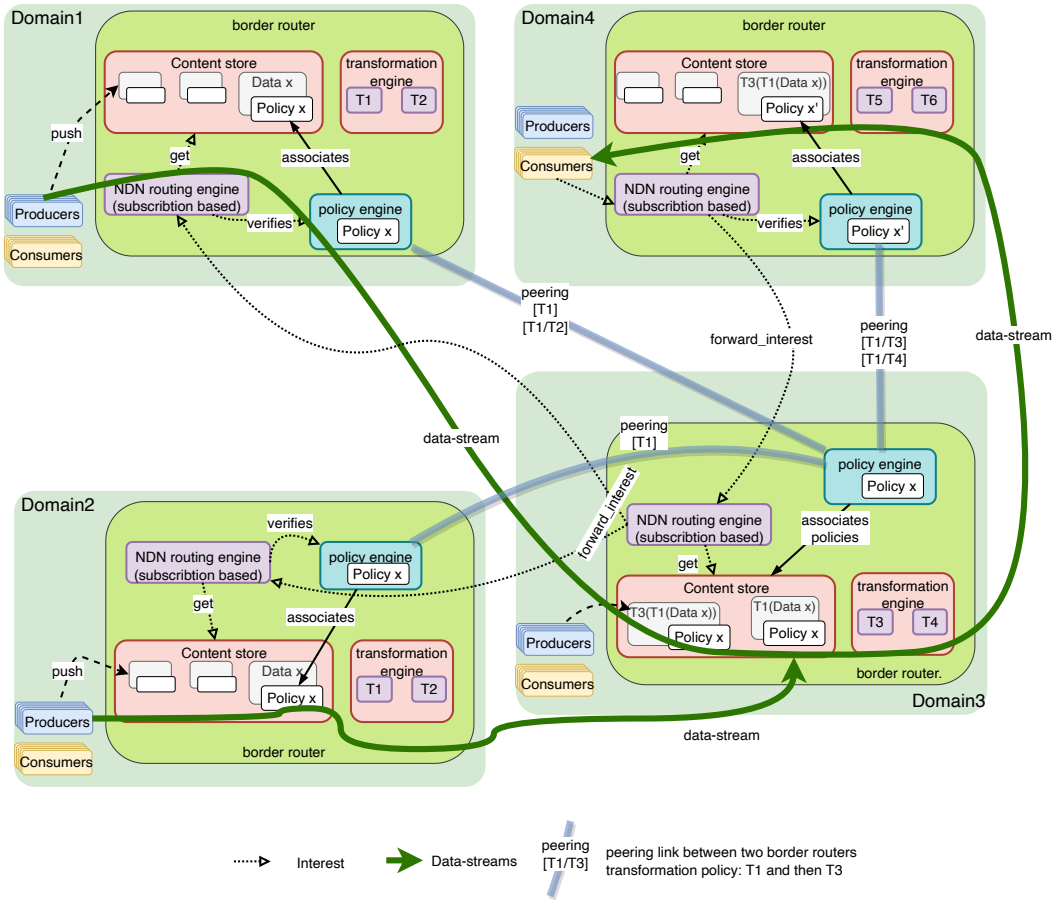


Figure 4.4: An overall illustration of our multi-domain overlay solution with all its components.

4.3.3 Policy engine

Each domain selects what data it accepts to export, and with which (trusted) peer domain(s). The data that exits a domain may have been generated locally or received from another domain, indistinctly. In particular, a border router may aggregate several data streams for anonymization purposes.

Each border router relays the interests of consumers. Inversely, at the other end of the chain, interests received by border routers are forwarded up to the concerned producers (border routers do it only if the data is not already present in their content store). Then, the data stream flows in the reverse direction of the interests, from producers to consumers, populating content stores of relaying border routers in the meanwhile. This way, the same transformed and aggregated streams may be re-used for similar interests of other consumers.

The policy engine is in charge of defining policies and exporting them to peering domains. More precisely, each border router associates a set of transformations to each of its datasets (*e.g.*, all temperature measurements). Then, border routers push each policy to its peers, denoting characteristics of the datasets that can be shared, *e.g.*, their size, the nature of their content, and the applied and requested

transformations. This exchange between border routers of adjacent domains is the basis of the control plane. It is indeed enough to install the required export rules in each policy engine of the overlay (see peering links in Fig. 4.4 with the requested transformations). When a border router later receives an interest, it has just to parse the policies it received from its peers to verify if it can send a reply or not. Thus, an interest is typically forwarded by the different border routers, so that the corresponding domains form finally a logical (sub)tree, rooted at the consumer, and where the leaves correspond to the producers.

The policy engine maintains a peer-to-peer connection with one border router for each domain with which it accepts to exchange data (export or import). The policy engine is in charge of constructing the exporting policies. These are composed of:

Peers list: list of domains with whom to maintain a connection to export/import data.

List of usages: usages that are authorized by the owner (commercial, market, research, etc). Nowadays, most data owners consent to only some specific usages of their data [15]. Usages correspond typically to keywords (see [133]), appended in the metadata.

Blacklist: for the same reason, a list of domains may be prohibited. These domains cannot access concerned data as long as the trust between peers is not violated.

Transformations: how the data should be transformed before being exported (*i.e.*, which mathematical transformation with which parameters). The applied sequence of transformations provides privacy guarantees.

Typically, the policy engine associates an exporting policy to each available dataset (see Figure 4.4), and the border routers enforce the policies. In other words, a peer receives an interest only if its dataset complies with the policies defined in the interest.

When the border router manipulates a chunk of data, it enforces the associated exporting policy. We consider here two types of enforcement:

Source transformations are applied when the data exits its producer domain;

Sticky transformations are applied by the peer receiving the data before it re-exposes the data in its turn (forwarding).

Indeed, we consider that the privacy concerns increase for an indirect peer, and a domain may force the insertion of additional transformations. We assume that a border router trusts its peers: they will respect the privacy constraints it defined (sticky transformations, usage, etc.)

4.3.4 Routing engine

The overlay of border routers is used when external data is required to be exchanged with other domains. For example, when an application needs a large volume of data, *e.g.*, information from numerous producers aggregated in a large stream, multiple domains will be solicited.

4.3.4.1 Interest and subscription through the overlay

The routing engine (which is implemented in the border router) is in charge of receiving and handling interests (Fig. 4.4). In our proposal, each interest is composed of the dataset description with the desired chain of transformations. In addition, the usage and the consumer domain are appended to the interests' parameters to check compliance with privacy policies. We also append transformations and parameters natively to the name of the interests so that (transformed) data-streams can be distinguished from each other. We encode this by embedding each transformation into a name component, *e.g.*, ***/ProducerName/DataSet/$average?every=5minutes***. The exact naming for datasets, transformations and parameters are left as an open issue, but some form of unified semantics has to be used.

If the data to answer a given interest is not available in its content store, the routing engine has to ask a peering border routers. First, it asks the policy engine all the datasets it knows (characteristics and imported policies). Then, it selects a sufficient collection of datasets to answer the interest (that maps to a collection of peers). Interests are issued to this collection of datasets in order to enable streams to answer the original interest.

In Figure 4.4, the consumer (in Domain4) asks its policy engine for the list of datasets it knows. Here, the border router forwards the interest to Domain3, which has exported the policy corresponding to the requested data. The border router of Domain3 receives the interest, and solicits its policy engine: the two datasets from Domain1 and Domain2 are required to form an aggregated data stream (to respect for instance a minimum number of measurements). Thus, in such a case, it forwards the initial interest to its two peers via their border routers. Note that it generates two novel interest packets to subscribe to the data from domains 1 and 2.

4.3.4.2 Policy enforcement

When the routing engine receives an interest, it has to verify that it is authorized to answer. In other words, it has to verify that the resulting data stream respects the exporting policy. Thus, it asks the policy engine the list of policies in its domain and verifies that a dataset is compliant with the interest.

Formally speaking, an interest is accepted if the requested transformation policy is at least as restrictive as the exported one, and if the usage is authorized:

$$\begin{aligned} \exists t \in \mathcal{F}_{src} \mid t \subseteq \mathcal{F}_{int} \wedge U_{int} \in U_{pol} \\ Pfn_{int} \notin BL_{pol} \wedge \exists t \in \mathcal{F}_{stik} \mid t \subseteq \mathcal{F}_{int} \wedge U_{int} \in U_{pol} \end{aligned} \tag{4.2}$$

Where Pfn_{int} denotes the prefix name of the interest's issuer, BL_{pol} the blacklisted domain prefixes for this policy, $\mathcal{F}_{src}$ and $\mathcal{F}_{stik}$ respectively the sequences of transformations of the source and sticky policies, $\mathcal{F}_{int}$ the sequence of transformations of the interest, U_{int} the usage specified in the interest, and U_{pol} the set of usages allowed by the policy.

4.3.5 Transformation engine

The border router has to apply transformations to the data-streams that exit its domain. To both improve the privacy and performance in terms of network load, transformations, and aggregations, in particular, should be applied as close as possible to producers. The transformation engine is in charge of executing these transformations. More precisely, the routing engine sends chunk(s) of data to the transformation engine, so that it can execute a specific piece of code, and send back the result to the routing engine. To be generic, we may rely on unikernel functions [69], which are functions retrievable by name (like NDN data packets) and run on top of lightweight virtualization. We let the exact definition of a global naming scheme of these transformations to future works.

4.3.6 Cache management and re-usability (routing engine)

Different interests issued by different consumers may re-use the same streams, at least partially. Thus, our solution identifies this overlap, to avoid forwarding the same data twice. If partial computations can be re-used as they are cached, we can rely on the NDN cache without any further actions. A border router has just to cache additional transformations (if required).

If an interest requires a chain of transformations which is a superset of the chain of an interest already being answered, the cached content is directly re-used. For instance, an interest may specify a set of transformations {T1 T2 T3} that is more specific than an existing cache entry (*e.g.*, {T1 T2}). Formally, the routing engine can re-use data in cache if the associated set of transformations matches exactly the first transformations of the interest:

$$\forall i \in [0, k_c], \mathcal{F}_{cached}(i) = \mathcal{F}_{int}(i) \tag{4.3}$$

With $\{\mathcal{F}_{cached}(i)\}_{i\in[0,k_c]}$ the sequence of transformations for the cached data, and $\{\mathcal{F}_{int}(i)\}_{i\in[0,k_q],k_q\geq k_c}$ the sequence of transformations specified in the interest (k_c and k_q respectively denoting the number of transformations, applied on the same prefix name, for cached data and the interest). Basically, the routing engine relies on longest prefix lookup for IoT interests such as enforced in Eq. 4.3 but only if Eq. 4.2 is verified. Then, it applies the transformations that are not present in the cached data (*i.e.*, $\{\mathcal{F}_{int}(i)\}_{i\in]k_c,k_q]}$ if $k_q > k_c$).

To improve caching efficiency, we force each border router to cache systematically all the chunks after the minimal set of transformations as specified in the exporting policy. This sub-chain corresponds to the smallest sequence in common for all the interests that match the dataset. Optionally, the border router may also cache the data which has undergone a subset of the transformations defined for popular interests. This extension may help to save computational resources in the border router if many interests overlap homogeneously.

Let us consider the example illustrated in Fig. 4.4. Domain3 is in charge of transforming the data between Domain4 and Domain1/Domain2. It collects the data, which has already been transformed with operation `T1`, and stores it automatically in its cache. That is, Domain3 stores the raw data received from Domain1/Domain2, `T1(data x)`. Since its exporting policy specifies that the transformation `T3` has also to be applied before the data exits the domain, Domain3 can also put this

additional transformed data in the content store to save computational resources (if another interest asks later for the same data).

4.4 Performance Evaluation

In order to evaluate the benefits of our solution, we assess its performance with simulations. We consider queries where consumers are interested in the average value of a collection of measurements. This transformation helps to preserve privacy by hiding the individual values. Let us denote $S_i =< c_i, v_i >$ a sampled value S_i that consists of a number of measurements c_i and its value v_i. Transformation function tf takes as input a collection of samples $S_{i\in[1,k]}$ and returns:

$$S_{tf} = \left\langle \sum_{i\in[1,k]} c_i, \frac{\sum_{i\in[1,k]} v_i \cdot c_i}{\sum_{i\in[1,k]} c_i} \right\rangle \tag{4.4}$$

We may implement similar transformations such as $Min()$, $Max()$, or more complex series transformations based on wavelets [134].

4.4.1 Simulation setup

We implement and compare the two following approaches:

Conventional NDN subscriptions: consumers directly subscribe to multiple data-streams to reach the desired sample size. Thus, all computations are executed by the consumer. This is the best comparison we can muster since non NDN approaches do not directly handle pieces of data, and transformations would not be so straightforward to implement;

Transformation overlay: our solution that exploits an overlay of border routers. These border routers implement the transformations, so that only aggregated data is forwarded across the network (cf. section 4.3).

At the time of this writing, this was the best comparison we could provide since privacy by enforcing anonymization and aggregation is not largely studied by the research community. While we could define a middleware entity that collects streams and employs NFN [68] or NFaaS [69] to provide in-network aggregation, it would be a contribution in of itself, and we chose not to go into that direction.

We extended the ndnSIM simulator (`https://ndnsim.net/`) to support all border router features and stream-based subscriptions. Our implementation is freely available[9].

To display our results we plot violin plots in the following figures. They allow us to display the distribution of data instead of simple average values. *Observe that some of the vertical axis are not continuous.* These are needed due to the large difference in scale among distributions. Our axis display the following metrics to evaluate our solution into multiple dimensions:

[9] https://icube-forge.unistra.fr/rcaminha/nanoas-proof-of-work

Table 4.2: Simulation parameters for the random topology

Parameter	Value
Simulation time	2 hours
Repetitions	30
Interests sampling period	uniform $\in \{1, 2, 4, 8\}$ min
Physical Network Topology	
Number of domains	20
Additional Edges	10
Network links	10Mbps, 10ms delay
Logical Topology	
Number of Producers	250
Logical trees branching factor	uniform distribution, $\in [2, 5]$

Size of content stores measures the amount of data in the content store of each node (*aka.* the cache size). We rely on unlimited content stores to analyze the total amount of data required for ideal re-usability conditions;

Network load measures the sum of data transmitted by all devices, to quantify both bandwidth requirements and battery consumption utilized by streams;

Normalized setup delay measures the time between the transmission of the consumer interest and the arrival of the first data packet for the corresponding stream. The value is normalized by the sampling interval of each consumer;

Hop count is the average number of hops in the physical network topology between a consumer and the producers which have provided the data for its interest;

Data spread counts the number of NDN routers that store each chunk of data. A large data spread means that a private chunk may be largely disseminated.

Our topology generation proceeds as following (see table 4.2 for parameter values used in our topology generation):

1. We generate random physical topology of domains (*i.e.*, network topology). To control the density and ensure connectivity, we first construct a random tree of domains. The 20 domains are put into an ordered array and each is connected to a random one from those after it in the array (uniformly). Then, we add a fixed number of edges between random pairs of domains;

2. we construct an overlay of domains (*i.e.*, which border routers are logical peers). A random root is selected, and we recursively expand the tree, by connecting its current leaves to a random number of non-connected domains until all domains have been picked. We control the random number of children with a branching factor parameter (see table 4.2);

3. producers are evenly distributed in the leaf domains;

4. we place the consumers uniformly in all domains. Each consumer generates an interest where the number of requested values is chosen uniformly between the minimum size of a domain (*i.e.*, its number of producers), and the size of the whole sub-tree.

It is worth noting that, the overlay uses longer routes since physical routes must follow the links of the overlay. Paths in the conventional NDN approach are shorter since data is disseminated via shortest path in the physical topology.

4.4.2 Simulation results

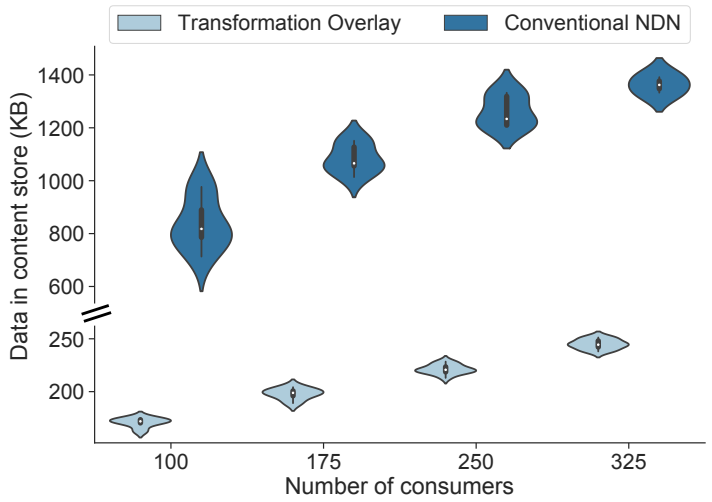


Figure 4.5: Content store usage of Border routers: our transformation overlay does not require to collect and spread all the raw data.

We first measure the size of the content store for both solutions (*Figure 4.5*). The re-usability provided by the conventional NDN cache engine is remarkable, many consumers re-use the same data, and the size of the content store does not increase drastically with the number of consumers. However, retrieving all the raw data is expensive. In other words, although the same chunk of data may be re-used, requesting a large volume of raw data increases the need for large content stores. On the contrary, our transformation overlay based solution allows each NDN router to store the transformed chunks of data. Thus, the volume of data to store in caches is at least 4 times lower in our simulations.

Then, we also measure the network load for all the routers (*Fig. 4.6*). Again, while each chunk of data may be efficiently disseminated in conventional NDN, it is not enough to limit the network overhead. On the contrary, the transformation overlay, which is first designed for a privacy purpose, is also able to reduce significantly the network load. Our proposal can greatly improve scalability. Note that both approaches reach a plateau: with the number of consumers increasing, the interests start to necessarily present a strong overlap, and the cached data becomes sufficient for any novel interest (*i.e.*, the content stores already have all the data).

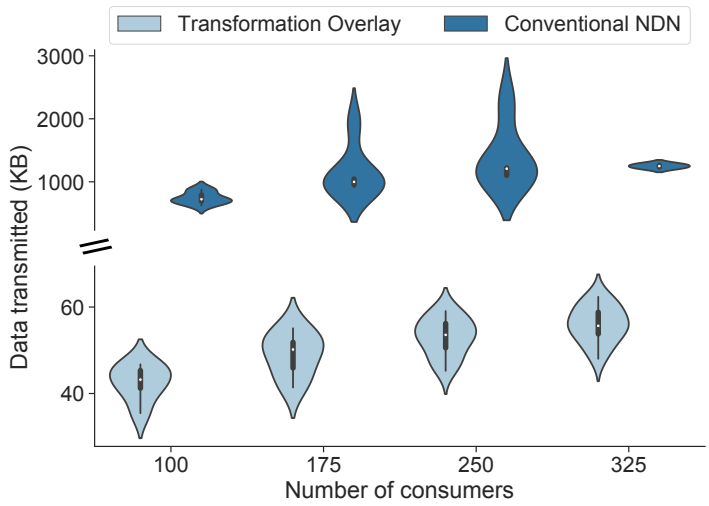


Figure 4.6: Data transmissions on Border routers: our transformation overlay scales well regarding the network load for large number of consumers (and requests).

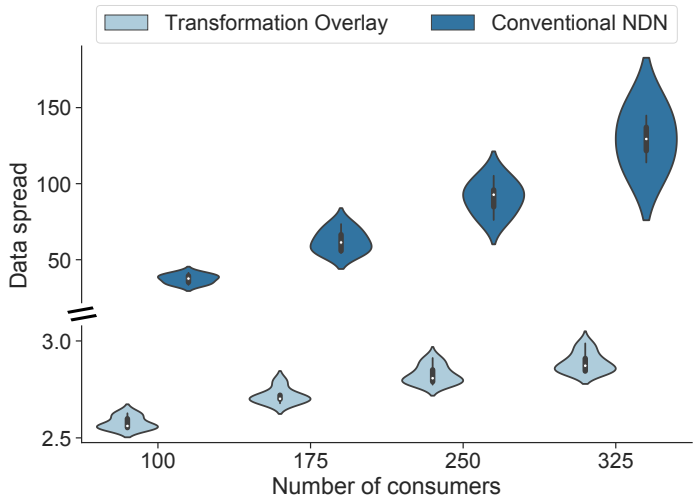


Figure 4.7: Spread of content in the network

Figure 4.7 focuses on the privacy characteristics. The data spread measures, for each dataset, the number of nodes whose content store has at least one raw sample of the given dataset. Basically, a chunk of data is produced and forwarded inside the domain, but the spread is limited by our transformation based overlay that transforms it before sending it to the consumer or the next level of the tree. The conventional NDN approach spreads the raw data in all the networks, the spreading being significant even if shortest routes are used. While data may be ciphered, it results in the need for complex access control schemes, which can become very challenging in complex multi-domain situations. Besides, the consumer still knows the identity of the producers since they get directly their data. We argue that it leads to a possible privacy leak if no further mechanism is here implemented to

anonymize the data.

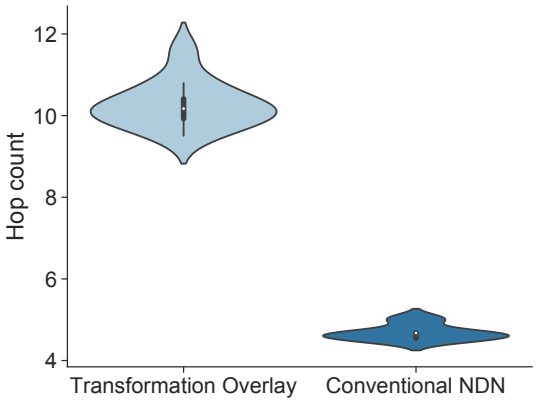


Figure 4.8: Path size between consumers and producers

Finally, *figure 4.8* illustrates the average physical hop count from the producers to the consumers. Indeed, our transformation overlay has a cost: domains have to forward data through the overlay, leading to suboptimal, longer paths. We can see that our transformation based overlay constructs physical paths twice as long as with the conventional NDN solution. It is the price to pay for enabling transformation based privacy: a domain does not trust blindly any other domain, and data has to be forwarded through detoured paths in our transformation based overlay. This increase is highly dependent on the number and locations of producers; if domains collect data from far away domains, this increase will be much larger, but if data is collected from directly connected peering domains, this increase may be much lower.

4.5 Conclusion and Future Works

We have presented a novel privacy-aware solution to safely exchange streams of private data in multi-domain IoT networks. We proposed an overlay of NDN border routers which forward the anonymized data that producers accept to export. More precisely, the control plane we propose leads to a tree structure that exploits exporting policies describing which data can be exported, at which extent, for which usage, and after which transformations. Data is transformed and cached before exiting the domain so that privacy concerns are respected while enabling re-usability using smart content stores looking at overlaps between interests among transformed datasets. Our simulations highlight the scalability of our solution. It is both able to enable fine-grained privacy rules along with efficient large scale multi-domain data exchanges. In particular, applications interested in aggregated data acquisition can strongly benefit from our proposal.

It would be interesting to incorporate NFN in our architecture as a way to define transformations or a way to enforce sticky policies. The named functions could be implemented as containers that both apply transformations and sign data. If data can only be signed through containers defined in the sticky policy, it would

be necessary to apply it in order to prove authenticity to other domains.

In the next chapter we relax the assumption that our overlay naturally forms a tree. We provide a publish-subscribe scheme to correctly aggregate data from arbitrary graphs of domains while respecting aggregation requirements.

Chapter 5

Privacy-Compliant Routing for Inter-Domain Aggregation

Contents

In situations with high density of devices, *e.g.*, smart cities, we envision that an infrastructure of interconnected systems (domains) will naturally emerge. We also expect domains to share resources such as sensors and communication infrastructure with each other in order to remain scalable.

Such partitioned architecture enables the aggregation of data from multiple owners through a shared infrastructure [135]. Consumers issue queries to other domains describing which data should be collected in terms of its descriptive metadata, *e.g.*, location, data type, and ownership. The data that matches the criteria is then collected and aggregated to answer the given query.

This implies interoperability and creates security issues: a domain needs to be able to acquire and correctly aggregate data from multiple domains, while respecting privacy constraints. In particular, we need to avoid data leaks, and let other domains decide their own privacy constraints.

In this chapter, we go further in the direction of the previous chapter by dropping the pre-existing tree-like topology (Figure 4.2). Domains then may peer with any other domain as they please. The overlay then turns into a mesh as domains provide data to each other.

This represents a challenging routing problem: the same anonymous data should not be aggregated several times as domains forward and aggregate it. Indeed, data may be forwarded through cycles or data may arrive to the same destination through alternative routes. Then, as domain collect, aggregate and disseminate data anonymously, it is possible for the same data sample to be aggregated multiple times and the result of the aggregation to become biased.

Aggregating data requires exploring non-intersecting datasets (see Sec. 4.1.3) while verifying minimal privacy requirements, here based on k-anonymity. A border router needs to identify datasets to publish, *i.e.*, set of producers (with their metadata) from which an aggregated stream may be constructed.

Instead of conventional aggregation methods of creating trees of domains to avoid repeated aggregation, we propose here to explore more flexible aggregations where data may transit until aggregations comply with privacy requirements. We tackle the problem by having producer domains publish their datasets and then all domains incrementally create aggregations from sets of non-overlapping datasets, and then, domains use the published aggregations to subscribe to queries based on metadata criteria. We will see here that we achieve greater aggregations by having data loop around in the overlay (without overlapping datasets) in order to acquire enough producers to reach minimum aggregation requirements and be further disseminated into the network.

The following section provides a better definition of the challenge tackled. It is important to remind that here we also are discussing a problem using concepts discussed in the previous chapter (see Section 4.1).

5.1 Problem Statement: Correct and privacy-compliant aggregation

To provide a global interconnection, peering domains may exchange locally produced data. Some domains act as ***publishers*** while the others may be ***subscribers*** to them. A domain may also forward data that was generated elsewhere if it is able to respect privacy concerns (based here on k-anonymity). In that case, a domain may be both subscriber and publisher.

5.1.1 Meshed overlay structure

We model the relationships among domains as a directed graph $G(V, E)$ where each vertex is an IoT network and an edge $u \rightarrow v$ exists if network u is a provider of data to its peer v. In real systems, an edge represents a commercial relationship among domains. These relationships bind receivers to respect the restrictions of providers (privacy, metadata, etc.). In particular, a receiver can forward a data stream only if it respects the k-anonymity property as defined by each of the producers. These entities are able to act as ***brokers*** between other entities by disseminating information acquired from others. Indeed, as a broker disseminates data acquired from its own data providers, it provides data to entities without any relationship with the data's producer. Due to the lack of guarantees between the producer and the consumers of a broker, the broker is the last entity capable of protecting the data.

5.1.2 Privacy requirements: minimum aggregation

We consider that data is private enough to be disseminated into the network when, according to the requirements of its producer, it is sufficiently aggregated. More specifically, direct peers may acquire data without aggregation, because of mutual agreement. However, these peers are not authorized to forward the data until it meets the privacy requirements of the producers.

Our privacy requirements are based on k-anonymity which determines that data cannot be distinguished from k-1 other samples. Producers reach privacy guarantees similar to k-anonymity by aggregating data with at least k-1 other producers.

When a dataset is composed of several producers, a border router must respect the highest privacy requirement among all of them. Let P be the set of producers part of a dataset and $req(p)$ be the **minimum aggregation requirement** of each producer $p \in P$. The border router must respect the following minimum aggregation requirement:

$$req(P) = Max_{p \in P}\left(req(p)\right) \tag{5.1}$$

While other privacy metrics may be implemented such as l-diversity, t-closeness and ϵ-differential [93] we chose to model our privacy settings over k-anonymity due to its simplicity. A producer can immediately understand the level of privacy acquired by k-anonymity and adjust the k parameter accordingly. Other metrics such as ϵ-differential require elevated levels of knowledge on statistics that make the choice of producers more obscure.

It is worth noting that respecting the minimum aggregation requirements creates constraints on the forwarding scheme. In particular, a producer may specify a very high minimum aggregation requirement, that prevents any peering domain to aggregate it. For example, in figure 4.1, if any producer has a requirement larger than three, the domain in the middle is not able to aggregate the data streams. In conclusion the topology of domains must be sufficiently dense, and/or the minimum aggregation requirement sufficiently low to enable a global dissemination. However, we are convinced that's the cost to pay to respect privacy.

5.1.3 Properties of Valid Aggregations

We target to create valid aggregations in a multi-domain environment. We propose the following three properties must be preserved when computing aggregated data.

P1 Only data of interest must be involved: Subscribers describe their data of interest in their query. Our subscription scheme must identify the relevant set of Matching Producers (MP), *i.e.*, the producers whose metadata match the query.

P2 Enforce non-intersecting anonymous datasets: the aggregation is applied recursively to form a stream. Obviously, the same producer must not appear several times in this aggregation: we need an aggregation tree. Else, considering the same sample value multiple times leads to bias.

P3 Aggregation requirements must be preserved: the aggregation must respect the minimum privacy requirement of each producer. This way, we enforce k-anonymity datasets [136]. We assume here that we trust peering domains, *i.e.*, they will respect our privacy requirements defined in our contract.

In the following section we describe our proposal to discover datasets and valid combinations of datasets that may be collected from other domains.

5.2 Proposal: Anonymous pub-sub of unbiased aggregated data

Our proposal builds on top of our NDN-based architecture (Chapter 4) by adding the mechanisms for domains to correctly aggregate data. Now considering a mesh of interconnected domains in the overlay, we propose here a publication and subscription mechanism to enable the detection of overlap and to construct globally consistent streams. Our publish/subscribe scheme behaves in two phases:

1. **Publication**: routers disseminate the valid combinations of producers. Each combination forms a dataset that respects the properties defined in Section 5.1.3;

2. **Subscription**: the consumer is able to identify the producers that match a query, and to construct a valid set of streams with the combinations discovered during the publication stage. Possibly, the union of combinations found may be a subset of the matching producers, when some producers have *e.g.*, privacy requirements impossible to respect.

5.2.1 Producer IDs and metadata

Our method maintains correct aggregation through the use of ***producer IDs*** that identify the data used in aggregations. These IDs are locally created by each producer via *e.g.*, the use of the hash of the network address and the metadata. Thus, other domains, apart from peers, cannot associate producers with their data but are

still able to identify overlapping datasets. Through their use we provide both properties **P1** and **P2**. If the hash function is sufficiently well-chosen, we can neglect the probability of collisions. If a collision occurs, it means that, unfortunately, the two corresponding producers cannot be merged in the same stream, which doesn't seem such a stringent drawback.

A producer associates its producer ID and descriptive attributes (*i.e.*, metadata) to the data it generates. In particular, the subscriber exploits the metadata to identify the data of interest of each query, forming the MP set. The descriptive metadata of producers forms a descriptive space, where each dimension of this space consists of all possible values of the attributes of producers. Similarly, the metadata criteria of queries identify areas of this descriptive space. Thus, matching metadata to criteria is simply checking whether a given producer is inside the area of the criteria.

Highly descriptive metadata may turn possible the association of producers and IDs. Thus, we assume that some level of anonymization is applied, such that real entities cannot be associated with IDs. For example, even if the ID is anonymous, a precise address will give away the identity of a producer.

5.2.2 Metadata similarities

When a border router has to construct the lists of datasets to publish, it must merge some datasets. Else, the number of offers would increase exponentially. We propose to merge datasets that contain *similar* producers. More precisely, we define the ***metadata similarities*** as the distance in the descriptive space. Such distances are easy to measure for continuous attributes (*e.g.*, geolocation). For categorical attributes (*e.g.*, device types), we need another quantification function [137]. We assume here a similarity function Δf exists. It quantifies the similarity between two datasets.

In our example from Figure 4.1, domains 1, 2, and 3 produce data with metadata containing their geolocation, as well as the measurement and sensor type, its accuracy, etc. Domains 1 and 2 have very similar metadata: a border router will merge preferentially domains 1 and 2. Domain 3 will likely be merged with more similar domains.

Figure 5.1 illustrates this similarity. We see the overlapping areas of interest of two queries, one selects the space of any speed measurements and the other selects vehicles speedometers. Observe that the datasets 1 and 2 are identical on this descriptive space and any query for one will also select the other. This serves to show that an offer created with similar producers will be frequently used. We detail the creation of offers in the next section.

5.2.3 Publication phase

Each border router has to identify the datasets it can publish, *i.e.*, share with its peers. An aggregation offer is a pair $o = (AS, r)$ where AS denotes the set of producer IDs that are being offered in offer o and r denotes the minimum requirement of these producers in offer o. Such an offer denotes that any combination of producers $C \subseteq AS$ may be enabled as long as it is compliant with the minimum requirement, *i.e.*, $|C| \geq r$.

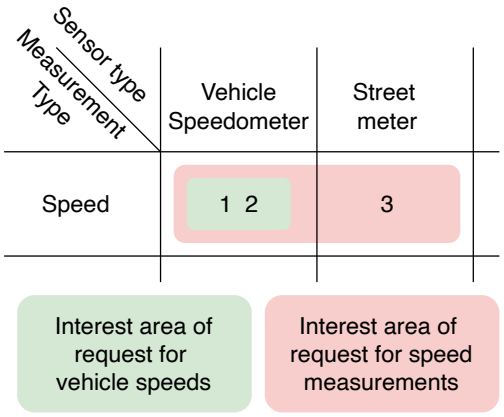


Figure 5.1: Illustration of vertex location in the descriptive space of our smart city transportation example of figure 4.1

On a high level, ***publication*** consists in filtering and/or transforming ***aggregation offers*** from incoming neighbors (***input offers***) and exporting aggregation offers to outgoing neighbors (***output offers***). Let us consider in *Figure 5.2* the use case described in Figure 4.1 (page 41) with a focus on offers. This figure describes the output offers of domains at different rounds. For example, at round 1 (Fig. 5.2a), the domain 1 exports an offer $(\{1\}, 2)$, meaning that it exports data from the producer ID 1, that has to be merged with at least 1 other producer before being exported (to respect a minimum aggregation requirement of 2).

The minimum requirement is implemented as a sticky policy in our multi-domain architecture. Domains collect data directly with any source transformation defined by producers but here we impose that an aggregation is also added as sticky policy. Any aggregation may be applied here: for example average, minimum, and histogram.

We assume, for simplicity, that the publication happens in a synchronized manner, *i.e.*, messages are exchanged at the same time and instantaneously. Thus, we consider a discretized time, counting ***rounds***. In figure 5.2, we specify the first round for which a given offer starts to be published. At the beginning of each round, a border router receives offers, process them, and publishes them at the beginning of the next round.

5.2.3.1 Completing offers to respect privacy requirements

Producers initially disclose their dataset as an offer with AS containing only their local ID and r being its own requirement (see round 1 of figure 5.2). Such offers naturally have a requirement larger than the available number of producers, *i.e.*, $|AS| < r$, which we denote as ***incomplete offers***. Analogously, we denote offers which already have enough producers to comply with the requirement, *i.e.*, $|AS| \geq r$, as ***complete offers***.

Complete input offers are directly disseminated (published) as output offers. They already are compliant with the minimum requirement of all producers, and thus must be Oppositely, incomplete offers cannot be disseminated as they do not

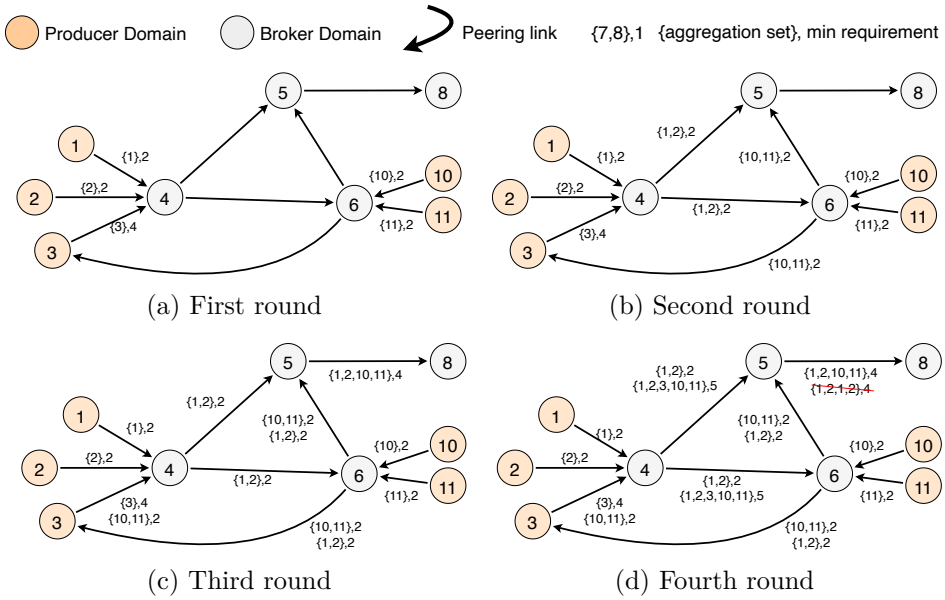


Figure 5.2: Extended smart city transport example and offers published

respect the privacy constraints of producers. The border router must extend these incomplete offers before publishing them.

We extend these offers by using producer IDs from other offers. Border routers find non-intersecting sets of producers IDs that can be acquired from different offers, *i.e.*, compliant with producers requirements. Then, the union of these sets of producers IDs can be merged into a novel offer. Essentially discovering data that can be aggregated together to fulfill privacy requirements.

The minimum aggregation requirement of the novel offer must be equal to the exact cardinality of the union of the sets found to enforce that each set is collected in full to ensure compliance with requirements. By this we also enforce that the new offer's requirement is larger than each producer requirement, as each set respects the minimum of each of their producers. As this is done several times among the border routers, the aggregation requirement of offers will be way above the requirements of producers.

In conclusion, we guarantee properties **P2** and **P3**. When offers are merged, the peer cannot extract where data has been aggregated. In particular, it hides the logical topology used to construct the aggregated set. We also hide the privacy requirements of each producers. This anonymity represents one of the strengths of our solution.

In figure 5.2 the broker 4 receives 3 incomplete offers during the first round. Thus it creates the novel offer $(\{1, 2\}, 2)$ to be published during the second round (Fig. 5.2b). Similarly, the broker 6 can combine two incomplete offers into $(\{10, 11\}, 2)$. The broker 4 cannot construct, during the first round, a valid complete offer with the producer 3: its minimum aggregation requirement is too high. It needs to wait for the round 4 (Fig. 5.2d), to combine it with the (complete) offers $(\{10, 11\}, 2)$, $(\{1\}, 1)$, and $(\{2\}, 1)$.

5.2.3.2 Avoid duplicated aggregated sets

Classical routing protocols use the concept of loop: a cycle exists in the routing topology that prevents the packet to be delivered. Concretely, the same router must not forward twice the same packet. In our architecture, a loop exists only when the data of the same producer is aggregated several times in the same stream. Thus, each router has to aggregate streams that don't overlap, to guarantee property **P2**.

Besides, a router considers only offers that can complete existing offers in a better way. More precisely, a router considers only the first peer that announces a specific offer. If the same offer is received from another peer later, it is just discarded since it may be a forwarding loop for this offer. The combination of these two mechanisms (*i.e.*, non overlapping aggregated set and first received offer) avoid the creation of any loop in the routing topology.

Let us still consider figure 5.2. In round 3 (Fig. 5.2c), domain 5 receives the offers $(\{1,2\},2)$ from domains 4 and 6. Then, at round 4, it could generate the invalid offer $(\{1,2,1,2\},4)$ but it would be invalid due to repeated producer IDs. The same will happen at round 5 (not illustrated here) by round 6 since domain 6 could generate the invalid offer $(\{1,2,3,10,11,10,11\},7)$ In our algorithm, we enforce empty intersections when merging offers.

Authorizing cycles in the physical topology allows us to increase the number of possible offers. Let us focus on the producer 3 which has a large privacy requirement in figure 5.2:

1. Broker 6 can aggregate the stream from producers 10 and 11, and publishes this offer to its peers (step b);

2. Node 3 receives this offer, that it can forward to 4 (step c);

3. Broker 4 can then aggregate the data from individual producers 1, 2, and 3 with the stream $(\{10,11\},2)$. This novel offer $((\{1,2,3,10,11\},2))$ is sent to its peers 6 and 5 (step d).

It is worth noting that a cycle exists in the physical topology: node 6 first publishes the offer with only 10 and 11, which is augmented by node 4 with producers 1, 2 and 3. However, the data of the same producer is not aggregated several times in the offer: we guarantee property **P2**. A loop can be easily identified by any broker: two different offers cannot be merged if their aggregated sets overlap.

5.2.3.3 Algorithm to construct complete offers

Let us define the conflict graph $CG(V',E')$ where the vertices V' are the available combinations of all valid combinations of producers ($V' = C^*$). An edge exists in the conflict graph between two vertices if the corresponding sets intersect ($E' = \{(C,C')|\forall C,C' \in V', C \cap C' \neq \emptyset\}$). Figure 5.3 illustrates the conflict graph for border router 4. Creating novel offers from incomplete offers consists in exploring the complement of CG to find maximal cliques: all the offers in a clique are pairwise disjoint. Thus, creating offers is thus a NP-Hard problem as an exponential number combinations of offers is possible, and each offer can potentially possess an exponential number of combinations of producers [138].

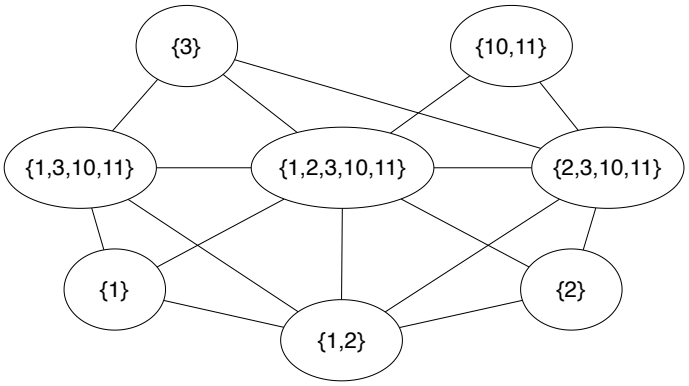


Figure 5.3: Conflict graph (CG) of the border router 4

An exhaustive exploration is too expensive for large instances of the problem. Thus, we rather propose a heuristic based on two criteria:

- We stop the completion as soon as we construct a given number of valid output offers nO_{max};
- We also stop after exploring the conflict graph a given number of times nC_{max} as the requirement may be too high to create a complete offer.

We propose to add as many IDs as possible to complete an offer. This greedy approach prevents easy de-aggregation of data due to similar combinations of producers being offered. Consider the offers for round 4 in fig. 5.2, in addition to $(\{1, 2, 3, 10, 11\}, 5)$, it is also possible to create $(\{1, 3, 10, 11\}, 4)$ and $(\{2, 3, 10, 11\}, 4)$. If these offers are published, a consumer may construct one stream for $(\{1, 2, 3, 10, 11\}, 5)$ and another one for $(\{1, 3, 10, 11\}, 4)$. In that case, the consumer may de-aggregate the two streams to infer the data produced individually by the producer id 2, which is clearly not suitable.

As already discussed in section 5.2.2, a border router should merge offers with similar metadata. Combinations of producers IDs also describe an area of the descriptive space. To use a given combination to answer a query, the combination should be fully within the area of interest of a query (Property **P1**). Thus, data is more likely to be selected by the same query if they are similar, *i.e.*, close in the descriptive space. During publication, we limit the creation of offers where the similarity among producers in AS are above a certain minimum threshold value, Δ_{min}, *i.e.*, $\Delta f(AS) \geq \Delta_{min}$.

Algorithm 1 details the actions taken at publication time:

1. We must first identify all producers IDs which are part of incomplete offers. In lines 1 to 5, we find the list of such IDs and their requirement. These incomplete offers are identified to be completed by other existing offers.
2. We initialize in line 6 the counters used to enforce the limit on offer creation and combination exploration. This way, we enforce the nO_{max} and nC_{max} limits: when these counters exceed their respective limit, we stop the offer creation for that round (lines 24 and 26).

Table 5.1: Table of notations

Variable	Meaning
O^-	Set of input offers of local node
O^+	Output offers of local node
O_n^+	Output offers of local node to node n
u	Local node
N^+	Outgoing neighbors of local node
Δf	Similarity function
Δ_{min}	Minimum similarity required for offer creation and query matching
$nO_{counter}$	Counter of offer creation
nO_{max}	Limit of offers that may be created per incomplete offer
nC_{max}	Maximum number of combinations explored during subscription
C^*	Set of all valid combinations from offers available to local node
$C_{val}(o)$	Set of valid combinations from a given offer o
AS	Aggregated set of given offer
r	Minimum requirement of given offer
$req(p)$	Minimum requirement of producer p
$\mathcal{I}(nP_{min}, criteria)$	A query for at least nP_{min} producers that match given metadata *criteria*
$meta_p$	Set of metadata from data shared by producer p
MP	Set of matching producers to a given query criteria
BS	Set of bootstrapper producers (r = 1)
k_{max}	Maximum aggregation requirement among producers

3. The main loop of our algorithm (lines 7 to 27) expands offers one by one. We first copy the initial set of IDs into variable AS^* and expand it by adding producers from incomplete offers one by one as the set complies with their minimum requirement. We also verify that for each producer p, there is no intersection, and it is similar enough to the others (line 11). After expanding the set AS^* as much as possible, we create an offer if any novel producer was inserted (lines 19 to 22). The novel offer must have requirement equal to the size of the set in order to prevent invalid combinations of producers.

4. The algorithm ends by returning an updated set of outgoing offers (line 28). These are the complete offers from O^+ in addition to the offer created with the domains local data. The set returned is expected to be used as input for the next execution of the algorithm on the next round of message exchange.

5.2.4 Subscription phase

A consumer needs to exploit the available aggregation offers received during the publication to answer a query. Our subscription algorithm takes a query, and the set of offers available as input. Then, it needs to explore the combination of offers

Algorithm 1: Generating output streams from input streams

Data: Input offers store O^-, Previous output offers O^+, Similarity function Δf, Minimum similarity for offer creation Δ_{min}, Limit of combination exploration nO_{max}, and Limit of offer creation nO_{max}.

Result: Updated output offers O^+.

$iAS \leftarrow \emptyset$ // Find producer IDs in incomplete single offers from neighbors and their requirements
foreach $(AS, r) \in O^- \mid r > 1 \wedge |AS| = 1$ **do**
  $iAS \leftarrow iAS \cup AS$
  $p \in AS, r_p = r$ // Store individual requirements
end
$nO_{counter} \leftarrow 0 \wedge nC_{counter} \leftarrow 0$ // Initialize counters of offer creation and combination exploration
foreach $(AS, r) \in O^+$ **do** // Begin expanding offers
  $AS^* \leftarrow AS$ // Copy starting set of producers to expand on
  foreach $p \in iAS$ ***in increasing order of requirements*** **do**
    if $|AS^*| + 1 \geq r_p$ **then** // If set is large enough for next producer ID,
      if $p \notin AS^* \wedge \Delta f(AS^*, \{p\}) \geq \Delta_{min}$ **then** // datasets are disjoints, and similar enough
        $AS^* \leftarrow AS^* \cup \{p\}$
      end
    end
    else
      break // Stop adding producers if set has not grown enough
    end
  end
  if $AS^* \neq AS$ **then**
    $O^+ \leftarrow O^+ \cup \{(AS^*, |AS^*|)\}$ // New offer from extended set of producers
    $nO_{counter} \leftarrow nO_{counter} + 1$
  end
  $nC_{counter} \leftarrow nC_{counter} + 1$
  if $nO_{counter} > nO_{max} \vee nC_{counter} > nC_{max}$ **then** // Stop iteration if any limit is reached
    break
  end
end
return $\{(AS, r) \in O^+ \mid r \leq |AS| \vee$ ***is local data***$\}$ // Only announce complete offers

that maximizes the number of producers it can collect data from.

5.2.4.1 Filtering offers of interest (matching producers)

A consumer can create a list of all known producers, which is the union of all the input offers. From this list, a border router extracts a set of MP that match the criteria of a query, and then the consumer needs to select a set of offers to be used to acquire the aggregation of producers in MP. The query specifies a minimum number of producers nP_{min} that have to be present in the answer. The objective of our subscription algorithm is to maximize the number of matching producers part of this aggregation to make the resulting dataset richer.

We first need to filter the offers of interest. Formally, the consumer needs to identify all offers $o = (AS, r)$ such that:

$$\exists C \mid C \subseteq AS \wedge C \subseteq MP \wedge |C| \geq r \tag{5.2}$$

For example, if we have the matching producers, 1 and 2 (Fig. 5.2), the offer $(\{10, 11\}, 2)$ cannot be used as $\{10, 11\} \not\subseteq MP$.

5.2.4.2 Identification of non-intersecting offers

The challenge is then to identify offers that don't intersect, and such that their union maximizes the number of producers. Indeed, we cannot have intersecting offers in the combination, else, the data from the same producer may be aggregated several times, creating statistical bias (property **P2**). Similarly, maximizing the number of producers in this combination allows the consumer to collect a richer dataset. Similarly to the publication problem, we must find maximal cliques in the conflict graph in order to maximize the number of producers offered to the consumer. Thus, this is also an NP-Hard problem as several combinations of producers may be possible.

Here we propose two stopping rules for our heuristic:

1. We stop when we reach the minimum number of producers required (nP_{min}) instead of continuing searching for the maximal number of producers;

2. We also impose a limit nC_{max} on the conflict graph at subscription in order to avoid excessive computation.

5.2.4.3 Exploration algorithm for subscription

Algorithm 2 details the actions taken at subscription time. It proceeds in the following way:

1. First, the consumer identifies the set MP that match the metadata criteria defined in the query $\mathcal{I}$ (lines 1 and 2). If the number of matching producers is too low, we must discard the query (line 3) as we are certain that no reply can be generated;

2. We identify the set of useful ID combinations (line 6). These are the ones that offer combinations of producers which are contained in MP (property **P3**). To simplify our pseudocode, we use the C_{val} function to denote all valid combinations from offer $o = (AS, r)$, *i.e.*, $C_{val}(o) = \{C \subseteq AS \mid |C| \geq r\}$.

Algorithm 2: Enabling streams to answer a consumer request

Data: Output offers O^+, Request $\mathcal{I}(nP_{min}, criteria)$, Similarity function Δf, Minimum similarity Δ_{min}, and Exploration limit nC_{max}

Result: Matching producer IDs and set of corresponding combinations to satisfy the interest $\mathcal{I}$, or two empty sets if the interest cannot be satisfied

1 $P_{avail} \leftarrow \bigcup_{o \in O^+} AS(o)$
2 $MP \leftarrow \{p \in P_{avail} \mid meta_p \text{ matches } criteria\}$ // Producers that match requested metadata
3 **if** $|MP| < nP_{min}$ **then**
4 | **return** $(\emptyset, \emptyset)$ // Not enough producers for $\mathcal{I}$
5 **end**
6 $C^* \leftarrow \bigcup_{o \in O^+} \{C \in \mathcal{C}_{val}(o) \mid C \subseteq MP\}$ // All usable producer combinations
7 $nC_{counter} \leftarrow 0$ // Initialize combination exploration limit
8 **foreach** $C' \subseteq C^*$ **do** // Extract disjoint combinations of producers that match the requirement $nProd$
9 | **if** $nC_{max} < nC_{counter}$ **then** // Verify limit of combination exploration
10 | | **return** $(\emptyset, \emptyset)$ // Stop if limit has been reached
11 | **end**
12 | **else**
13 | | $nC_{counter} \leftarrow nC_{counter} + 1$
14 | **end**
15 | $Ans \leftarrow \bigcup C'$ // Compute possible answer aggregated set and intersection
16 | **if** $\bigcap C' = \emptyset \wedge Ans \subseteq MP \wedge |Ans| \geq nP_{min}$ **then** // Make sure answer is of interest, disjoint and large enough
17 | | **return** (Ans, C')
18 | **end**
19 **end**
20 **return** $(\emptyset, \emptyset)$ // No matching was found

3. Similarly to algorithm 1, we limit the exploration of combinations via a counter per query (line 7). This counter is incremented for each failed attempt to find a valid answer to the query. The query is rejected if the limit is exceeded (line 9).

4. We search for a combination that respects the following conditions (lines 8 to 19):

   (a) it only combines offers with producers that match the metadata criteria of $\mathcal{I}$ (property **P1**);

   (b) it results in a valid aggregation by having an empty intersection (property **P2**);

   (c) the number of producers is enough to satisfy the query $\mathcal{I}$.

   If such a combination cannot be found, we reject the query (line 20).

5.3 Performance Evaluation

To assess the performance of our proposal, we evaluate it on dense topologies composed of numerous producers attached to several broker domains. Producers provide their information to multiple brokers that acquire the data and, in turn, disseminate the resulting aggregated information to other brokers (and their customers). This setup is challenging as the same piece of data can be provided multiple times via the meshed overlay. Resulting in artificially generate biased information (intersecting producer sets). We indeed aim to evaluate our proposal in a difficult scenario for performing the aggregation correctly.

For each new offer, our method selects the incomplete offers to merge based on similarities (see section 5.2.3.3). In practice for this evaluation, we use the geographical distance as a similarity metric. We assume here that the descriptive space of producers is simply the normalized physical coordinates (a square space with side equal to 1). Similarity among metadata is assumed to be the complement of the distance among producers (regarding their physical 2D locations) with respect to the eccentricity of the two-dimensional Euclidean space (*i.e.*, the maximal distance in our square in practice). Thus, the similarity between two sets of producers P_1 and P_2 can be described as follows:

$$\Delta f(P1, P2) = \frac{\sum_{\forall p_1 \in P_1 \wedge p_2 \in P_2} (D_{max} - d(p1, p2))}{|\{(p_1, p_2) | p_1 \in P_1 \wedge p_2 \in P_2\}|} \tag{5.3}$$

where $d(x, y)$ is the euclidean distance between producers x and y and D_{max} is the eccentricity of the considered Euclidean space ($\sqrt{2}$ in our example). With such a valuation, brokers aim to aggregate and offer data from producers that are similar to each other, *i.e.*, datasets which are sufficiently close in space to each other.

Besides, it is worth to notice that the minimum aggregation requirement of each producer may have a significant impact on the convergence of our scheme. Indeed, a large minimum aggregation value implies an increase in the number of incomplete offers, such that it becomes more and more challenging to complete them. To analyze the impact of such a requirement on the efficiency of our scheme, we simply rely on an uniform distribution to explore its effects: each producer selects uniformly its minimum aggregation requirement in the interval $[2, k_{max}]$ where k_{max} is a tunable parameter (limited to 10 in our simulations). We also model a given subset of permissive producers, i.e. having a minimum requirement of 1, so that the system can bootstrap. We denote BS this set of *bootstrappers*.

For each simulation, we generate 30 random graphs composed of two parts having the following characteristics:

1. 30 brokers belonging to a strongly connected graph. We first use the Erdös-Rényi method [139] such that each edge has a probability of 0.15% to exist and then check whether the resulting is indeed strongly connected (otherwise we generate a new one and so on).

2. 170 producers are attached to the graph of brokers. More precisely, we connect each producer to its 3 closest brokers (in space). This way, brokers offer data of similar producers according to the metadata considered.

Table 5.2: Evaluation parameters

Simulation parameter	Value unless specified
Number of graphs generated	30
Broker vertices	30
Producer vertices	170
Edge probability between brokers	15%
Producers metadata model	Location in 2D square with side equal to 1
Brokers connected to each producer	3 closest brokers in 2D square
Queries issued in each graph	50 to random brokers (uniformly picked)
Minimum similarity Δ_{min}	0.7
Offer creation limit nO_{max}	10
Combination exploration limit nC_{max}	100
Size of matching producer set $\|MP\|$	50
Number of bootstrappers $\|BS\|$	20
Maximum aggregation requirement k_{max}	10

Finally, we generate 50 queries randomly while we set the combination exploration limit nC_{max} of our scheme to 100 (we limit to 100 the number of combinations of offers to explore, cf. algo 2). Each query is generated by a broker randomly selected in the graph, and corresponds to picking collecting the data of producers from a given area in the 2D space. In practice for the evaluation, the selection criteria[10] (line 2 of alg. 2) of a query matches with the $|MP| = 50$ closest producers related to a randomly picked location. Note that, at the end of our analysis, we modify this static $|MP|$ value into a range to better understand its impact. Generally speaking, all the parameters used in our simulations are listed in table 5.2: we provide default values in use when we are not varying them.

5.3.1 Aggregation Upper Bound: Unlimited Walk

To analyze the efficiency and the limits of our pubsub mechanism, we construct here an upper bound as some queries may be with an optimal solution. Obviously, such an optimal approach is inapplicable in realistic situations since it requires a complete central knowledge of the topology, and potentially a complete exploration of each offer combination. Intuitively, an aggregation that respects the minimum requirements of each associated producer can be forwarded to any broker to be completed: the aggregation grows iteratively. Through an *unlimited walk*, we can

[10]While we try to remain generic in the pseudo-code, the matching criteria needs to be actually specified in the implementation, potentially according to the kind of criteria but it is left for future works.

Algorithm 3: Unlimited walk. Aggregation upper bound achievable for a given request in a given topology.

Data: Set of brokers B, Set of matching producers MP, Requirement r_p of each producer $p \in MP$.
Result: Set of producers that are collectable UW.

```
1 UW ← ∅                                   // Begin considering empty aggregation
2 ∀b ∈ B, P_b ← MP ∩ N⁻(b)                 // Find producers which are one hop away from
    each broker
3 while ∃b ∈ B ∧ ∃p ∈ (P_b \ UW) | r_p ≤ |UW| + 1 do   // While there are
    brokers that can still aggregate
4 |   while ∃p ∈ (P_b \ UW) | r_p ≤ |UW| + 1 do   // Aggregate data from novel
  |     producers
5 |   |   UW ← UW ∪ {p ∈ (P_b \ UW) | r_p ≤ |UW| + 1}
6 |   end
7 end
8 return UW
```

feed an aggregation as long as we identify at least one remaining broker that can complete it with some locally connected producers. It is worth noting that we have to run this algorithm for each topology, query and distribution of minimum requirements.

Our upper bound (Algorithm 3) maintains the list of producers that have already been aggregated. Recursively, we identify a broker that can complete this aggregation with a set of local producers such that, (*i*), each minimum aggregation requirement is respected, and (*ii*), all these local producers are part of the matching producers (*i.e.*, they match the query's metadata). Initially, the aggregation is build considering a single broker having a set of locally connected matching producers that can be aggregated together while respecting the minimum aggregation requirement.

5.3.2 Metrics

We compute the following metrics for each combination of parameters:

Normalized answer dataset size: the number of matching producers that can be constructed in response to a specific query by using available aggregation offers. Larger is better: the subscriber can acquire larger datasets than expected as long as the set of producers match the query (it may be included in a larger compliant set). This value is normalized by the number of producers matching the query issued ($|MP|$). We plot distributions in the form of violin plots with our publish-subscribe algorithm. With our unlimited walk algorithm we only need to plot lines as it does not exhibit a large variability as observed with our heuristic.

Offer table size: the number of IDs in the offers available at each broker. Assuming that each ID is stored as a hash, we calculate how many bytes would be

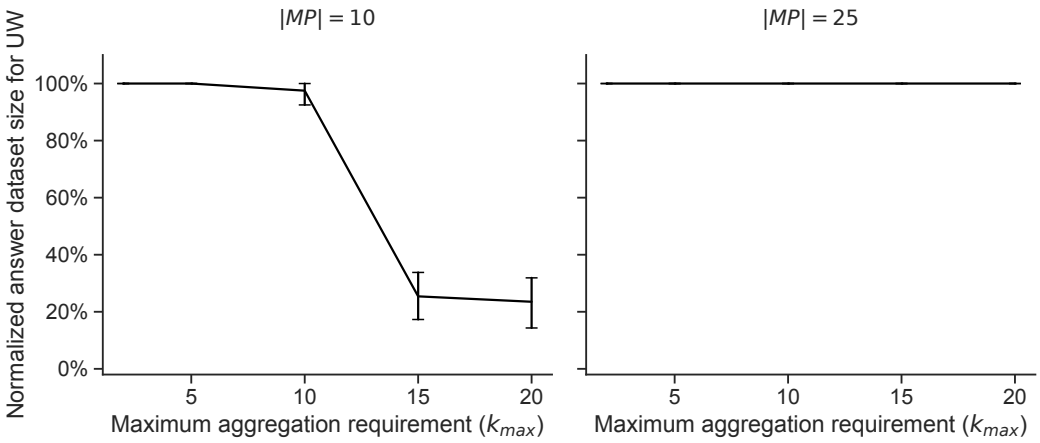


Figure 5.4: Limits of aggregation for very restrictive queries

required to store all available offers. We compute this sum using hash sizes for the md5 functions.

Processing time: amount of time taken by our algorithm. For the publication stage, we measure how long each message exchanging round takes. For the subscription stage, we measure how long it takes to find the set of offers to use to answer the query.

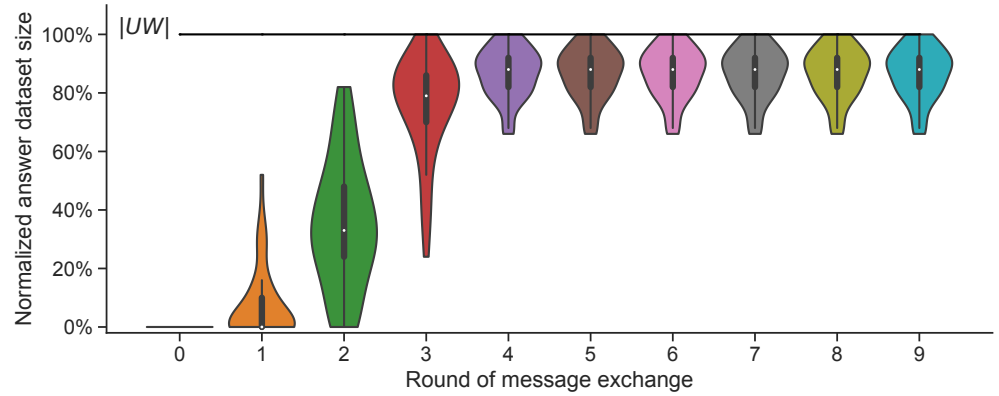
Our figures display these metrics through the use of violin plots. These plots show the distribution of values in the vertical axis. Additionally, our plots include a mini box plot in the middle of each violin to indicate the mean, quantiles and whisker intervals. These graphs should allow the reader to get insight of our population of samples.

5.3.3 Completeness of the subscription

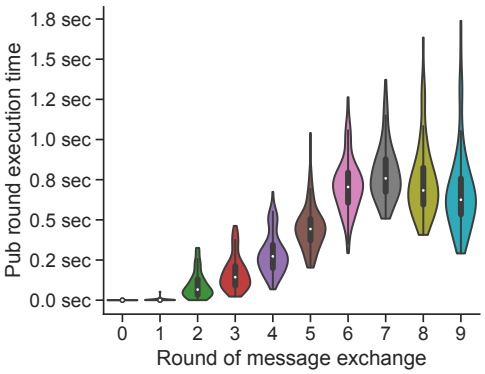
We measure the size of the normalized dataset according to the minimum aggregation requirement *(Figure 5.4)*. Using a small minimum aggregation requirement mitigates how the system is constrained: the consumers can then easily construct a subscription that collects the data from all the matching producers especially when the number of matching producers is high enough (e.g. 25).

With a large set of matching producers, we maximize the chances to combine the data generated by different producers. For instance, when we have 25 matching producers (*i.e.*, 15% of the producers), the subscriber can construct an aggregation with all the concerned producers, whatever the minimum aggregation requirement of each of them is.

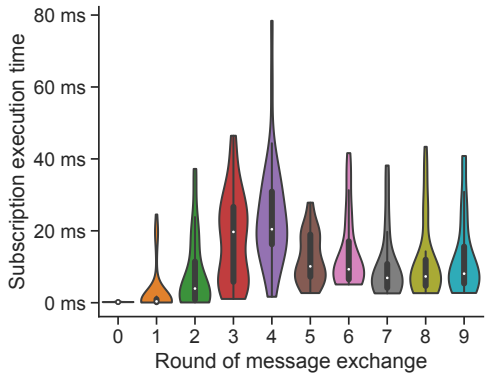
With a few matching producers (*e.g.*, 10, 5% of the producers), the minimum aggregation requirement starts to be a constraint. Since the minimum aggregation requirement is probabilistically chosen between 0 and K_{max}, the subscriber may start to collect a suboptimal set of matching producers only when K_{max} exceeds 15. In these conditions, the subscriber can only construct a subset of the matching producers that may be acquired with the ideal, upper bound algorithm. In practice,



(a) Aggregation capacity at different rounds



(b) Duration of each publication round



(c) Time to find streams to answer queries at different rounds

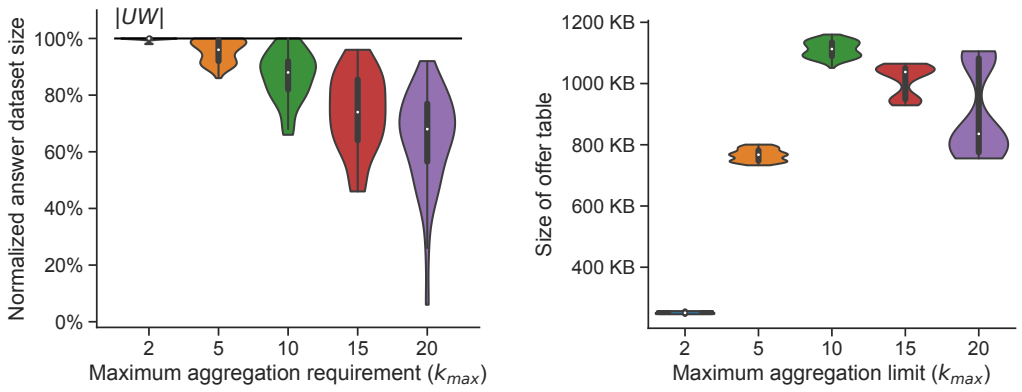
Figure 5.5: Convergence of the publication step

when such extreme situations do not occur (the ideal method is then successful), we will see that our method is able to perform pretty well.

5.3.4 Convergence of the publication step

We first study the convergence of the publication phase, by measuring the average size of the dataset at the end of each round (Fig. 5.5a). Our system looks to converge quickly: each broker is able to complete the incomplete offers. After only 4 rounds, the subscriber is able to construct a dataset that respects the privacy constraints by selecting a compliant set of offers. The number of rounds depends both on the eccentricity of the space (its maximal distance) and the minimum aggregation requirement, that may generate a larger set of incomplete offers.

Then, we measured the computation time per round for the publication step (Fig. 5.5b). The computation time increases with the number of rounds: each broker has more offers to filter and to complete. Mechanically, the computation time increases. However, it converges after only 6 rounds. The computation time even decreases slightly for more than 7 rounds: the incomplete offers are easier to complete thanks to the offers diversity. In any condition, a broker spends less than 2 seconds to compute the combination of offers. Since such dissemination is done



(a) (b)

Figure 5.6: Impact of k_{max}

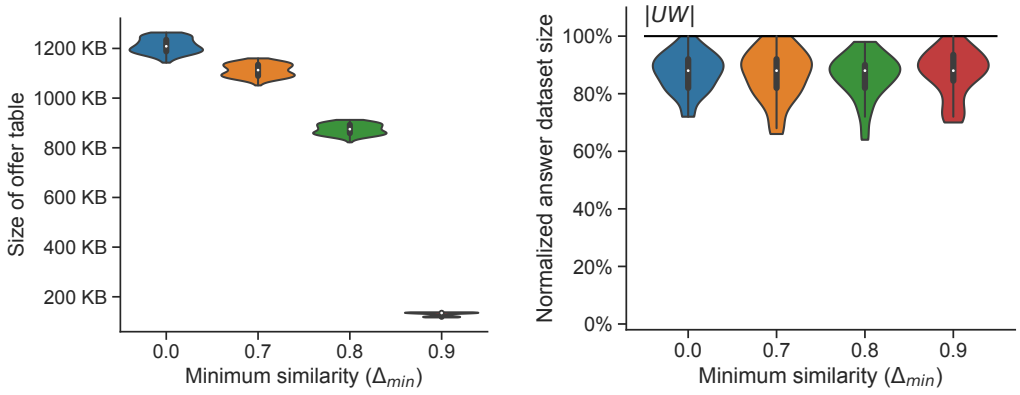
once (it is independent of the number of queries), this seems reasonable.

Finally, we measured also the computation time for the subscription (Fig. 5.5c). While it increases at the beginning since the subscriber has more offers to filter, it converges fast and the computation time even decreases. More offers are available, and the subscriber can efficiently the whole dataset by parsing the most promising set of offers. Thus, our subscription algorithm is efficient to construct a privacy-aware set of producers matching the query. A few dozens of milliseconds are at most required to identify which offers to combine to construct a valid dataset.

5.3.5 Impact of the privacy requirements

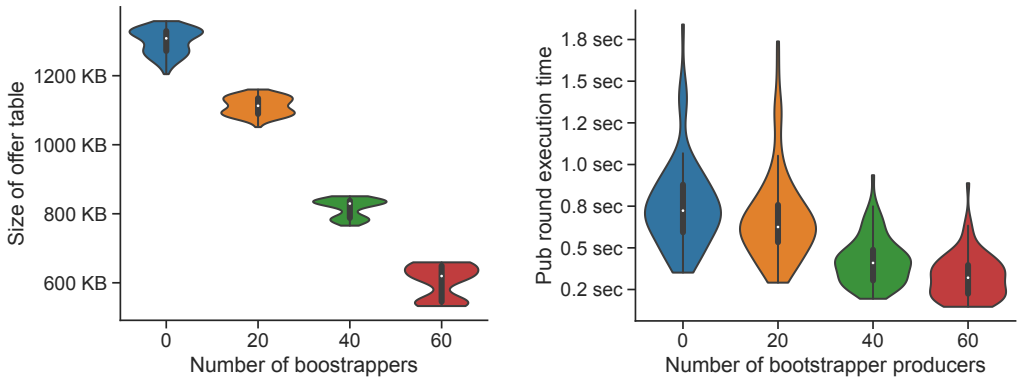
We now measure the impact of the privacy requirement defined by the producers (Fig. 5.6). We only report the metrics at the end of the last round, after the system has converged. We first focus on the dataset size (Fig. 5.6a). With a larger average k_{max}, the normalized answer dataset decreases. Indeed, we create probabilistically more privacy constraints, and we restrict the number of producers from which the subscriber can collect data. However, even with very strict constraints, the consumer can subscribe to a significant set of producers, that have probabilistically a smaller requirement. Thus, our system is robust to heterogeneous privacy constraints.

Let us focus now on the size of the offers' table for each broker (Fig. 5.6b). The number of offers is very small when privacy is straightforward ($k_{max} = 2$): any broker is able to combine one or two local producers. Thus, no incomplete offer exists, brokers do not need to explore and construct complex sets of offers. A few offers for each couple of attached producers is sufficient to cover the whole dataset. More privacy means a larger number of offers to mutually combine at first. Inversely, a very high privacy implies that incomplete offers at some point cannot be combined, and are not announced. The number of offers present in the table decreases.



(a) (b)

Figure 5.7: Effects of Δ_{min} on metrics



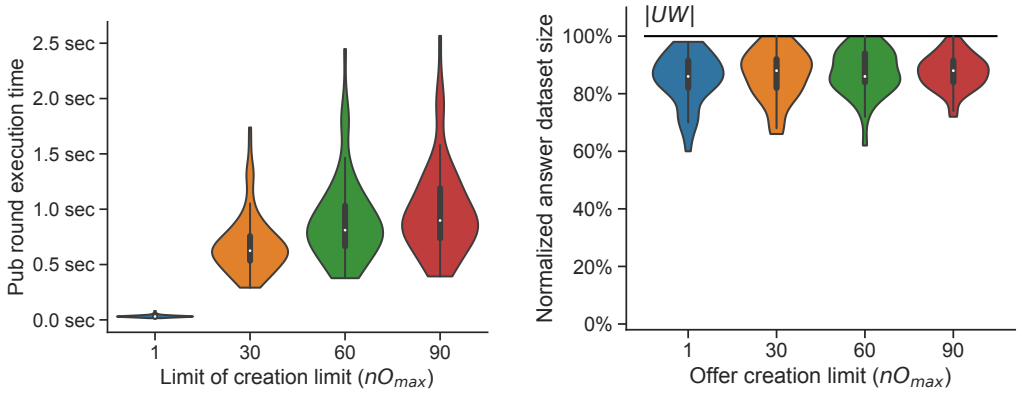
(a) (b)

Figure 5.8: Effects of bootstrapper producers on metrics

5.3.6 Impact of similarity

Let us now focus in Figure 5.7 on the impact of the similarity metric, and in particular the similarity threshold value in algo. 2.

When the broker does not consider similarity when merging different offers ($\Delta_{min} = 0$), the number of offers in the table is maximum (Fig. 5.7a). Indeed, a broker tries to maximize the number of combinations to not restrict the possibilities for subscription. Interestingly, the number of offers may significantly decrease (by 85%) when considering a very large similarity threshold value when merging offers. What is really interesting is that merging together similar offers (in our case from geographically close producers) has negligible impact on the subscription phase (Fig. 5.7b). The subscriber is able to construct a dataset of at least the same size with matching producers, whatever the Δ_{min} value is. Our method merging the data according to its similarities is efficient to reduce the number of offers without significant concession on the accuracy. It is worth noting that multi-dimensional similarities may be more challenging as it may limit this effect in more complex and large scale scenarios.



(a) (b)

Figure 5.9: Effects of nO_{max} on metrics

5.3.7 Impact of bootstrapping producers

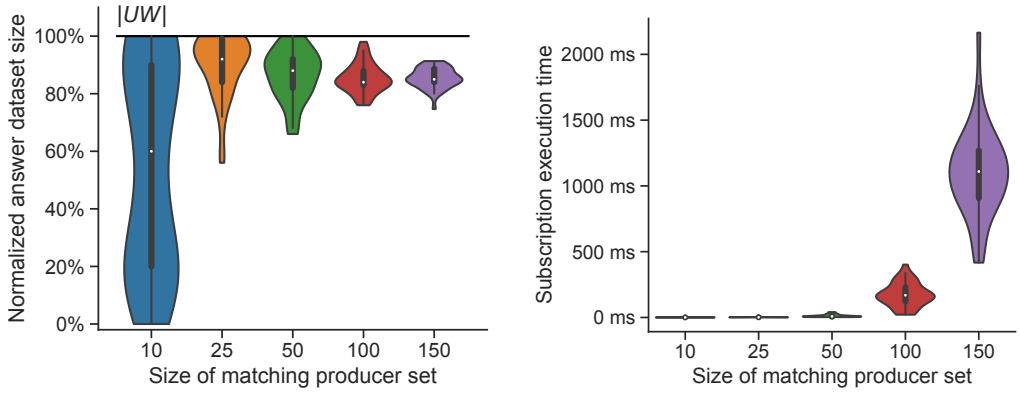
Let us now focus on the impact of presence of bootstrapping producers, *i.e.*, producers without any privacy requirement ($req(p) = 1$). They can be used by any broker to complete their incomplete offers, to allow later the subscriber to select a larger dataset. In particular, they allow a broker to reduce the number of offers to explore (Fig. 5.7a). Indeed, a broker does not need to generate a very large set of offers: combining one incomplete offer with a bootstrapping producer is sufficient. Mechanically, it reduces also the computation time (Fig. 5.8b), by almost 50% with 60 bootstrapping producers (one third of the producers). It is worth noting that this complexity reduction doesn't impact the accuracy. Indeed, the normalized answer dataset size remains unchanged (and thus not plotted): the subscriber can still use the same dataset for the answer, which is less restrictive than those without bootstrapping producers.

5.3.8 Offers limit creation

We now verify the impact of the parameter limiting the number of offers to construct in algo. 1. As soon as a broker has constructed nO_{max} offers, it stops the exploration. Our scheme is scalable since the nO_{max} has a limited impact on the computation time (Fig. 5.9a). In the extreme case ($nO_{max} = 0$), the exploration stops very soon, and the computation time is drastically reduced. The impact on the normalized answer data size is limited: the broker is still able to complete most of the offers. The minimum value is slightly slower since the most sensitive queries are more complicated to be fulfilled.

5.3.9 Selectivity of the query

Finally, we measure the impact of the query's selectivity, *i.e.*, the number of producers that match the query (Fig. 5.10). With a small number of producers (*e.g.*, $|MP| = 10$), the graph of producers is very sparse. Thus, it is very challenging to complete the incomplete offers (*i.e.*, the matching producers that have a high



(a) (b)

Figure 5.10: Impact of the number of producers that match the query ($|MP|$)

minimum aggregation requirement). Finally, the subscriber is able to construct a dataset of only 60% of the dataset which may be obtained with our centralized upper bound. Even if a solution exists to aggregate data, the brokers are so distant that they cannot combine their offers (they are distributively filtered before). However, as soon as the number of matching producers becomes reasonable, our scheme is efficient to construct an accurate set of offers.

We see a slight decrease in our aggregation metric as we increase the number of producers selected (fig.5.10a). This is bound to happen with two factors that impede finding a combination of offers to collect these producers. First, finding the set of offers that are non-overlapping becomes increasingly harder as the set increases. Second, The limit of combinations will not allow for finding a perfect set, which limits our subscription even if there exists a set of offers to answer the query perfectly.

Obviously, it impacts the computation time (Fig. 5.10b). This is due to the larger set of possible aggregations to answer the query (see line 6 of algorithm 2). Intuitively, a broker is directly attached to a larger set of matching producers. Thus, the number of incomplete offers decreases, as well as the number of offers at all. Even in the worst case, the subscription phase needs a few seconds to compute the largest set of matching producers while still respecting the privacy requirement of each of them.

5.4 Conclusion and Perspectives

We presented a publish-subscript method to share aggregated information among anonymous groups of IoT devices (domains). Our method protects the privacy by enforcing a minimum number of producers that must be aggregated together in order to have their information disclosed. The details of each producer are significantly reduced while maintaining information of the population.

We propose an algorithm to construct offers, that can be published, *i.e.*, disseminated in the network. Each offer specifies the level of privacy that must be respected. Inversely, we also propose a subscription algorithm able to combine all

these offers to answer to a query. It takes care of not using several times the same data in the aggregation tree which is implicitly constructed at subscription.

Our performance evaluations highlights the efficiency of our algorithms to identify correctly the offers to publish and to combine. We show that our method quickly reaches acceptable levels of aggregation within the possibilities allowed by the network. We also show that our method can be executed by resource constrained IoT gateways in order to disseminate and collect data in a multi-domain infrastructure.

While we solve the issue of correctly aggregating data among domains, our scheme still *has the major flaw* of stream deanonymization by comparing streams with similar aggregation sets. Consider that a single consumer or broker is able to enable streams $\{1, 2, 3\}$ and $\{2, 3\}$ to acquire the average values of the producers, and we denote $mult_s$ and val_s as the resulting multiplicy and average value of the answer dataset of stream s. From the equation of unweighted average, we can derive the value of stream $\{1\}$ by using *equation 5.4*.

$$\begin{aligned} mult_{\{1\}} &= mult_{\{1,2,3\}} - mult_{\{2,3\}} \\ val_{\{1\}} &= \left(mult_{\{1,2,3\}} \cdot val_{\{1,2,3\}} - mult_{\{2,3\}} \cdot val_{\{2,3\}}\right) \div mult_{\{1\}} \end{aligned} \tag{5.4}$$

With some adjustment we can make this equation generic to obtain subsets of streams that use the average function, other functions require different mathematical formulas. Moreover, the issue is that domains may enable streams that do not comply with minimum requirement of producers (Property **P3**) to acquire subsets of streams that do not comply with the minimum requirement of producers

Indeed, a broker should be able to detect enabled streams that allow for this and disable them. This filtering may be applied both during publication (not disseminating partially overlapping offers), and during subscription (active streams must concern non overlapping producers). If filtering is applied during the publication step, the opportunities of aggregation may be greatly reduced, so filtering only active streams would be more interesting. Additionally, domains can choose which streams to filter depending on criteria such as:

pricing more expensive streams replace existing cheaper streams if these novel are more rewarding;

privacy level streams of more domains replace streams with fewer domains, which result in more privacy since data is more aggregated

However, this is a local decision and streams may be collected through non-intersecting paths that would make detection impossible in this manner. A global means of enforcing minimum differences among domains is necessary.

Chapter 6

Conclusion and Future Research Directions

This thesis has explored the privacy issues of querying data from multi-owner IoT environments. We base our privacy model on domains: groups of IoT devices within the same scope of ownership, objective, and/or location. Privacy boundaries take shape after the frontiers of these domains as we assume that data is allowed to freely transit within a domain due to ownership or consent.

Our first contribution (Chapter 3) defines a multi-domain model where domains exchange data with each other through a scalable approach to provide privacy by design. Our model includes formally defined queries issued among domains. These queries are formed in such a way that private data is secured through anonymization. We then take advantage of our model and two freely available datasets to define a network framework to evaluate the feasibility of our approach. In particular, we show that LoRa networks can be used for dense multi-domain networks if deployed with enough density.

Our multi-domain framework serves as a concrete case study of multi-domain scenarios which are not yet widely deployed. It shows the characteristics of private data exchanged among these domains, *i.e.*, network flow, attributes and processing.

We initially developed this model in order to use it for our own simulations. However, this model proved to be limited in terms scalability in the number of domains. We could not stress our proposals to show the full benefits of reuse and complex aggregation paths. As you read, we rather used more generic and random evaluation scenarios. They allowed us to highlight the scalability of our proposals in the number of domains and the paths of aggregations used by domains to respect privacy constraints.

Nevertheless, our model can be re-used by others to study their proposals for multi-domain networks. It shows realistic traffic generation and chains of services with formally defined data streams. Proposals on base station deployment, service chaining and query handling can directly use this model. Either to exemplify the implementation of a real world application in their proposal or to simulate their proposal using the data found in the datasets.

Our second contribution (Chapter 4) is about an architecture for inter-domain communication which we have based on the Named Data Networking (NDN) network stack. By using NDN, we leverage producer and consumer authentication, semantic queries, and efficient dissemination of information via its cache mechanism and detection of redundant transmissions. Our architecture enforces transformations, *i.e.*, preprocessing operations, to data that transits among domains, which might be used for privacy or enrichment of data, *e.g.*, filtering, aggregation and masking. By imposing the NDN protocol, we also provide interoperability among domains that use different internal protocols.

The proposed architecture dissociates from other proposals for privacy among independent IoT networks that primarily employ complex encryption systems to lock data under different levels of security. We investigate the use of anonymization to both maximize reuse of data to answer multiple queries and ensure privacy. Our performance evaluation shows that we acquire great gains of performance with the reuse of anonymized data even if we do not provide full opaqueness.

Our third contribution (Chapter 5) improves our previous contribution by allowing domains to self organize in a more complex overlay topology. We indeed drop the strong assumption of our second contribution: domains are not any more organized as a tree to correctly aggregate data. We expect domains to provide data to each other simply based on commercial relationships which would likely result in the same data being sold to several domains. This might result in repeated data collection and aggregation if data is anonymous and indistinguishable. Thus, we provide a publication scheme to disseminate datasets and incrementally create combinations of producers that are compliant with the privacy restrictions of producers and without repeated data entries. We also provide a subscription scheme that uses information acquired during publication to answer metadata-based queries.

Our results show that we quickly reach the upper bound of aggregation possible while respecting minimum aggregation requirements. They also show that our scheme hardware requirements are well within the capabilities of resource restricted devices.

The contributions of this thesis provide us insights on the problem of privacy via anonymization for inter-domain communication. In the next section we discuss what contributions we can envision to expand the work of this thesis and also discuss what other directions may be taken to study the overall research question of privacy aware inter-domain communication.

6.1 Short Term Research Directions

We begin with short to medium term directions that can be directly taken from the work in this thesis.

6.1.1 Evaluation framework and LPWAN for multi-domain

We first discuss our multi-domain scenario and LPWAN evaluation (Chapter 3) where some clear paths can be taken to expand the model and improve our evaluation to give more insight on LPWAN evaluations.

Regarding the model, the part of the contribution that most relates to this thesis, it can be expanded with more domains and queries if we use available datasets. We picked NYC as the real-life location of our model due to the amount of data available though official datasets[11] or APIs such as Google Maps[12] and TomTom[13].

For example, the traffic volume for various years and time windows is publicly available and can be used to model a domain that monitors road incidents and alerts vehicles in the area. This domain would broadcast messages to the area around incidents to notify drivers and prevent further accidents. Messages would be dispatched at moments of unusual traffic found through anomaly detection [11] with the dataset. Extending the framework would allow it to be used for more complete simulations and possibly turn it into a benchmark for multi-domain proposals.

Our LPWAN evaluation can be easily expanded with different technologies. Narrowband IoT is a good candidate as it is another leader in the field of LPWAN in addition to LoRa [79]. While we use the mathematical model to simulate LoRa, Narrowband IoT simulation models are available for NS3[14] and Matlab[15]. It would illustrate the difference between the two technologies as Narrowband IoT is able to reach higher data rates but may not be able to handle as many devices as LoRa per base station.

Tackling the problem of base station deployment is also an interesting research path. With our framework we can simulate network traffic while observing spacial and temporal variation with great granularity. However, this is not true for all real life datasets, missing data limiting the utility of these datasets [140]. Thus, it would be interesting to gather different deployment methodologies [141] and study how they behave under different levels of details and missing data on expected network traffic. In other words, by simulating aspects of real datasets such as missing data and different levels of granularity (data collected daily/monthly/weekly) we investigate how resilient these are to imperfect data.

6.1.2 Multidomain architecture

Our architecture allows for domains to enable streams that on their own are valid (*i.e.*, respect privacy requirements and aggregated without overlap). However, the enabled streams may be deanonymizable if they partially overlap and the aggregation function allows for it. We show in Section 5.4 that we are able to acquire unaggregated data of a single domain with equation 5.4. The problem can be generalized as: given two streams S_1 and S_2, it may be possible to acquire $S_1 \setminus S_2$ or $S_2 \setminus S_1$. Consequentially, the size of these differences must be larger than the minimum requirements of the individual producers that compose them. More formally, given that r_p is the minimum requirement of a producer p, if producer $p \in S_1 \setminus S_2$ then $|S_1 \setminus S_2| \geq r_p$ or if $p \in S_2 \setminus S_1$ then $|S_2 \setminus S_1| \geq r_p$.

This is quite challenging because it implies the choice of active streams and the optimization of some metric, *e.g.*, privacy level, acquisition cost or monetary reward. Any novel streams enabled must verify this property with all existing

[11] http://opendata.cityofnewyork.us/
[12] https://developers.google.com/maps
[13] https://developer.tomtom.com/
[14] https://www.nsnam.org/wiki/NB-IOT
[15] https://fr.mathworks.com/help/lte/nb-iot-channels.html

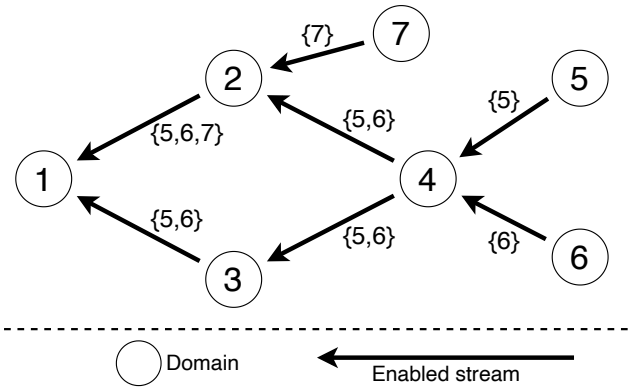


Figure 6.1: Example of deanonymizable streams

streams. This can be enforced by each domain checking novel streams only against those locally enabled or also include all streams enabled by other domains. While local verification ensures privacy with the scope of a domain, domains may not detect some situations where a malicious domain acquires deanonymizable streams through two or more different domains.

Let us consider the scenario in Figure 6.1. Here, domain 4 acts as a broker for domains 5 and 6 and supplies their aggregation to 2 and 3 but while 3 simply forwards this data, 2 will forward it after further aggregating the data of 7. Assuming all minimum requirements equal to 2, all enabled streams are valid according to properties described in Section 5.1.3. Observe that domains 2 and 3 cannot locally detect the two deanonymizable streams $\{5, 6, 7\}$ and $\{5, 6\}$ that are being supplied to 1.

We visualize two ways to implement global validation. The first would be to employ a trusted third party that keeps tracks of available streams: domains request to enable streams and this would allow it. This is not viable in our multi-owner scenario as it would centralize too much control and turn it into a single point of failure and give it all the information. The second mean of implementation would be to employ a distributed database such as a blockchain. While we remove the "all seeing" trusted third party, now all domains see all streams, which in of itself is too much public information, and we also include a lot of overhead to maintain the blockchain. Indeed, both options have their own drawbacks.

At publishing time, domains advertise their dataset and incrementally offer valid aggregations to others. We idealize a simple heuristic that could be used to minimize the problem by increasing the size of datasets ($|AS|$) in offers. It would enforce large aggregations by replacing smaller aggregation offers with larger ones. By enforcing larger aggregations we increase the probability of enabled streams resisting deanonymization. The exact changes to algorithm 1 is left to be formulated.

This however would make answering streams much more difficult since we would make available offers larger and more difficult to comply with line 16 of algorithm 2 where that verify that all data being aggregated matches the criteria of the query. Thus, analyzing the effects of this onto the aggregation capabilities is necessary.

Now, we change the discussion to more long term research directions that could be taken for inter-domain communication.

6.2 Long Term Research Direction

I have two directions for long term research. The first continues the work but by taking differential privacy as the base for the privacy model. The second diverges with the use of federated learning techniques.

6.2.1 Inter-domain privacy model based on differential privacy

A more endearing endeavor would be to replace k-anonymity as the base of inter-IoT privacy restrictions. A good candidate to replace k-anonymity would be differential privacy that dictates that query answers do not significantly change (based on a given ϵ parameter) depending on database entries (more formally defined in page 18). By creating a distributed privacy model based on the ϵ-differential metric, we could prevent the extrapolation of information from different query answers. Thus, solving the biggest problem in our last contribution.

It is already possible to reach approximated ϵ-differential datasets in a distributed manner [102, 103]. But it would be interesting to investigate this in multi-domain infrastructures.

By aggregating data from two or more datasets independently ϵ-differential datasets, the property (Eq. 2.2) is likely to be broken. The intuition is based on the diversity among the data of domains: different data subjects and locations. This means that entries from one set can be very different from the other and resulting queries are more affected by individual entries.

The different requirements of producers also translate to multiple ϵ parameters. Then, as datasets anonymized with different parameters are aggregated with each other, the resulting dataset must be anonymized to reach the more restricting (lower) value of ϵ.

Indeed, we need to adjust the privacy restrictions of merged datasets in order to make it compliant with producer's restrictions. This operation would be applied to anonymize the aggregated dataset after every aggregation. However, since the method to turn data ϵ-differential is through the addition of noise, the utility of data may be quickly lost due to the constant addition of noise. Making it necessary to investigate if such an approach is actually practical.

6.2.2 Federated learning approach

On the other hand, federated learning has taken increased attention from the literature to aggregate data. While this solution is not to aggregate information, it can be used to privately acquire information on the population. We go into further detail in the following section.

In federated learning, the system aims at training given machine learning model with the data of several users. However, each user data can only be accessed by that user and a global model cannot be trained by the merged dataset of all users. This makes each user locally train a model and transmit it to a concentrator entity that will train a global model from all the local models. Finally, the global model is distributed to all users[142].

For example, if each user is a building and the objective is to model the power consumption in each household based on the number of inhabitants. One could

apply linear regression in this case, *i.e.*, find a linear function $y = a \cdot x + b$ that best approximates to the data points of a dataset. Each building would find a pair a, b which models their household's data and submit this to an aggregator, that would merge all local models by finding a global linear model.

While this hides the individual entries of users, the parameters of a local model can be too detailed depending on the machine learning technique used. Thus, the addition of noise to or encryption of parameters using homomorphic encryption (see page 10) is employed. Another problem is the higher error in the resulting global model when compared to trainings performed using the global dataset[143].

This is similar to what we do in our last contribution but focused on machine learning. Our contribution can be extended to reuse techniques to protect local model parameters to in turn protect the intermediary aggregation results. Then, our k-anonymous based offer publication can be used to minimize the addition of noise and encryption when collecting and merging intermediary models.

More specifically, consider my example of smart buildings and modeling household consumption. The local parameters of each building would either require the addition of noise or the application of Homomorphic encryption. Let us consider only noise addition here. If we employ our offer publication scheme instead and merging models as they are forwarded, the resulting parameters would not need to be protected since they model data from multiple domains. This would result in less noise in the final global model since there would be less noise added to parameters.

At any rate, a federated aggregation scheme where data is modeled locally, and a global model is created from local models seems to be the direction that the community is moving to. It allows for the distributed and private modelling of data in a multi-domain architecture for artificial intelligence applications.

List of Figures

List of Tables

Agrégation de Flux de Données IoT Confidentiels Dans des Réseaux Multi-Propriétaires

L'Internet des Objets (IdO) est devenu une partie de la vie de nombreuses personnes au cours des dernières années et on s'attend toujours à ce que le marché se développe encore plus [5]. En général, les dispositifs IdO peuvent gérer et être imprégnés de systèmes tels que le chauffage, les appareils électriques, l'éclairage, et bien plus encore. Ces systèmes IdO imprègnent la vie moderne, des appareils ménagers aux environnements de travail et aux systèmes des espaces publics.

Microsoft a réalisé une enquête intéressante qui illustre encore mieux l'ampleur et la couverture de l'IdO [7]. Elle a ainsi conclus que la plupart des gens sont déjà ou seront bientôt impliqués dans un système IdO en tant que sujet surveillé ou en tant qu'utilisateur.

Au sein de ce vaste marché en pleine croissance, nous nous attendons à ce que les systèmes IdO déployés se chevauchent à la fois géographiquement et en termes de fonctionnalités, car ils coexistent dans la même infrastructure [8]. En effet, nous nous attendons à l'émergence d'une infrastructure complexe avec plusieurs systèmes IdO coexistants offrant une large gamme de services (multi-IdO).

La collecte et le traitement des données provenant d'appareils intelligents parmi les réseaux IdO seront une caractéristique essentielle de ces scénarios. La collecte d'ensembles de données brutes peut être à la fois plus verbeuse que nécessaire et trop coûteuse en raison du volume de données à transmettre et à stocker. Ce qui impose de prétraiter l'information pour l'adapter aux besoins du consommateur [14]. Il est également courant de ne divulguer que les ensembles de données qui ont été anonymisés avec des opérations permettant de préserver la vie privée.

Ensuite, dans le contexte du multi-IdO, les données que les réseaux divulguent doivent être contrôlées pour maintenir les restrictions des différents producteurs. Par exemple, les requêtes ne doivent pas demander et accéder aux identifiants (par exemple, le numéro de sécurité sociale) ou elles doivent demander des données moyennées au lieu d'échantillons individuels. Les données divulguées doivent faire l'objet d'une attention particulière, car nous nous attendons à ce que les réseaux s'échangent des données entre eux.

En résumé, trois problèmes principaux découlent de l'utilisation de données

provenant de différents réseaux IdO : l'interopérabilité, l'évolutivité et la confidentialité. Cette thèse se concentre sur les problèmes d'efficacité et de confidentialité tels qu'énoncés par la principale question de recherche abordée dans ce rapport:

Défi Scientifique

Comment assurer l'agrégation dans le réseau de flux IdO provenant de multiples services et propriétaires indépendants tout en respectant chacune de leurs contraintes de confidentialité ?

L'évolutivité est la clé ici puisque nous avons affaire à des scénarios multi-IdO vastes et denses. L'interopérabilité prend un rôle secondaire dans cette thèse car nous ne nous y intéressons pas, mais nous maintenons la condition que chaque système IdO fonctionne avec des dispositifs, des protocoles et une sémantique différents [9]. Ensuite, la sécurité est grandement affectée par le traitement d'un environnement multi-propriétaire en raison des objectifs, de la crédibilité et des restrictions différentes de chaque service.

Nous fournissons trois contributions principales dans cette thèse qui divergent un peu de l'approche habituelle de la sécurité entre les domaines. Notre approche permet la communication inter-système en adoptant une approche centrée sur les données où celles-ci sont anonymisées au fur et à mesure de leur diffusion.

État de l'art

Nous discutons des différents domaines de recherche sur lesquels nous nous sommes concentrés pour répondre à la question de recherche de cette thèse.

Flux de données IdO

Dans les écosystèmes de l'IdO, nous trouvons des dispositifs ayant des objectifs différents et les capteurs sont les principaux producteurs de données. Nous trouvons ces derniers en grand nombre afin de surveiller de nombreuses caractéristiques comme la température et la pression atmosphérique. Ces producteurs envoient leurs données de mesure dans le réseau afin que les contrôleurs, les actionneurs et d'autres entités décisionnelles les utilisent (consomment) pour leurs divers objectifs [17]. Ainsi, les producteurs diffuseront en continu des mesures horodatées pour permettre aux consommateurs d'être en possession de données constamment mises à jour et de stocker des chronologies complètes [13]. Les producteurs pousseront leurs données, à travers le réseau, vers les consommateurs abonnés à leurs flux : soit de manière ponctuelle, soit en fonction de l'occurrence des événements [19].

Lorsque les producteurs ne sont pas directement associés aux consommateurs, les réseaux nécessitent une entité intermédiaire pour arbitrer la communication, c'est-à-dire des courtiers [23]. Plusieurs solutions prêtes pour le marché sont également disponibles pour ce type d'archétype, par exemple, MQTT [29] et CoAP Observe [24].

Les producteurs décrivent leurs informations afin d'être mis en correspondance avec les requêtes. Ces descriptions sont fournies sous la forme d'attributs de métadonnées et décrivent des informations telles que le type de données et le lieu de la mesure [13]. Des modèles complexes de description des métadonnées sont nécessaires en raison des disparités dans la syntaxe et la sémantique des données entre les producteurs [38, 12].

Nous nous concentrons plutôt sur les exigences de traitement de ces requêtes. Les consommateurs demanderont les données sous une certaine fonction de calcul, par exemple, le filtrage, la catégorisation, l'agrégation. Ces fonctions sont importantes pour fournir aux consommateurs des données personnalisées (ayant plus de valeur). nous nous concentrons sur les algorithmes d'agrégation.

Flux de données agrégées

La définition la plus élémentaire des fonctions d'agrégation les définit comme des fonctions qui, lorsqu'elles sont appliquées à un ensemble d'entrées, produisent une sortie contenant des informations décrivant l'ensemble des entrées. Par exemple, des descriptions statistiques qui fournissent un résumé des données d'entrée tout en rendant difficile l'extraction des données d'origine. Ces fonctions agissent comme des opérations de protection de la vie privée qui permettent à la fois de préserver l'utilité des données et de masquer les informations sensibles.

Les stratégies d'application de ces fonctions d'agrégation dans les réseaux IdO sont classées en trois grands groupes : centralisée, en grappe et arborescente [40]. L'agrégation centralisée est la méthode classique où toutes les données sont collectées à un puits, une passerelle ou un consommateur (destination) pour être agrégées [41]. Pour les stratégies en grappe, les dispositifs élisent une tête de grappe parmi les dispositifs environnants pour transmettre ses informations [42]. Les stratégies basées sur les arbres sont la principale méthode d'agrégation locale de l'IdO : en raison de la connaissance minimale de la topologie et de l'agrégation correcte [39].

La confidentialité devient alors un problème clé pour l'agrégation en réseau, car l'entité agrégatrice doit avoir accès aux données afin d'appliquer l'opération d'agrégation. Le chiffrement homomorphique a alors été proposé pour résoudre ce problème car les fonctions mathématiques peuvent toujours être appliquées lorsque ce type de chiffrement est utilisé [52]. Cependant, un nombre limité d'ensembles d'opérations s'est avéré être valable pour ce type de chiffrement [54]. Ainsi, nous avons choisi de ne pas considérer ce type de solution dans cette thèse.

Réseau de Données Nommé

Le Réseau de Données Nommé (RDN) [55] est une approche propre à la pile Internet actuelle, qui répond très bien aux besoins des applications IoT. Conformément à la mentalité centrée sur le contenu, les consommateurs interrogeront le réseau en décrivant les données demandées. Les routeurs transmettront ensuite ces requêtes aux producteurs qui y répondent. In inclut des fonctionnalités telles que le transfert sémantique, la signature des paquets et la mise en cache distribuée.

Malheureusement, le RDN n'est pas directement applicable à l'IdO en raison des fonctionnalités manquantes et des limitations de l'IdO [56, 57]. L'abonnement

aux flux est l'un de ces mécanismes qui ne sont pas pris en charge car, par défaut, il existe une association 1 à 1 entre les intérêts et les messages de données. La communication basée sur les courtiers n'est pas non plus prise en charge de manière native par les routeurs. La mise en cache avec des dispositifs restreints est un défi en raison de l'espace de stockage limité disponible et de l'alternance des dispositifs entre les modes actif et veille pour économiser la batterie [57]. Le calcul en réseau est introduit dans le paradigme RDN par Réseau de Fonctions Nommées [68] qui propose d'intégrer le lambda-calcul dans les intérêts, où chaque routeur RDN peut traiter les données avant d'envoyer la réponse.

Anonymisation des flux

L'anonymisation assure la confidentialité en diminuant la précision des données. Au cours de l'anonymisation, les attributs de l'identifiant sont supprimés ou remplacés. Les quasi-identifiants sont ensuite modifiés afin de diminuer la probabilité d'associer les sujets à leurs informations sensibles. D'autres attributs peuvent être laissés tels quels, catégorisés (par exemple, en changeant l'âge exact en groupe d'âge) ou en ajoutant un bruit contrôlé pour masquer les valeurs exactes.

Ce processus est effectué jusqu'à ce que des mesures de confidentialité telles que k-anonymous [94] et e-differential [95] soient atteintes. Elles quantifient les niveaux de confidentialité en termes de probabilité de désanonymisation des informations. Ils mesurent la similarité des entrées de données entre elles et la similarité entre des ensembles de données entiers.

Un ensemble de données k-anonymes exige que chaque entrée soit indiscernable de k-1 autres entrées. En d'autres termes, chaque combinaison de valeurs de quasi-identifiants a au moins k entrées dans l'ensemble de données. Même si l'on sait que les informations d'un sujet se trouvent dans un ensemble de données et que les attributs identifiables sont connus, la probabilité de l'identifier est de $1/k$. Plusieurs algorithmes ont été proposés pour atteindre le l-anonyme [96, 97, 98] qui se concentrent sur le regroupement des entrées de données afin de modifier au minimum les quasi-identifiants pour maximiser l'utilité des données.

Étude de cas des services Multi-IoT sous LPWAN

La pléthore de services de la ville intelligente met en évidence les divers avantages que IdO peut apporter à la population, aux entreprises et aux autorités locales [105]. Une collection de capteurs est généralement disséminée dans une ville intelligente, et surveille un large ensemble de paramètres environnementaux tels que la vitesse du trafic, la qualité de l'air, la météo, le bruit, etc.

Les villes intelligentes attendent également que les différents systèmes cohabitent entre eux et échangent des données tout en protégeant les données sensibles. L'isolement a un coût [106] : les mêmes ressources radio doivent être partagées entre un plus grand nombre de flux, ce qui est particulièrement préjudiciable dans la bande ISM.

Nous étudions un scénario de Système de transport intelligent (STI) où des services distincts partagent des données pour offrir un système de transport globalement efficace et respectueux de la vie privée pour les villes intelligentes. Les

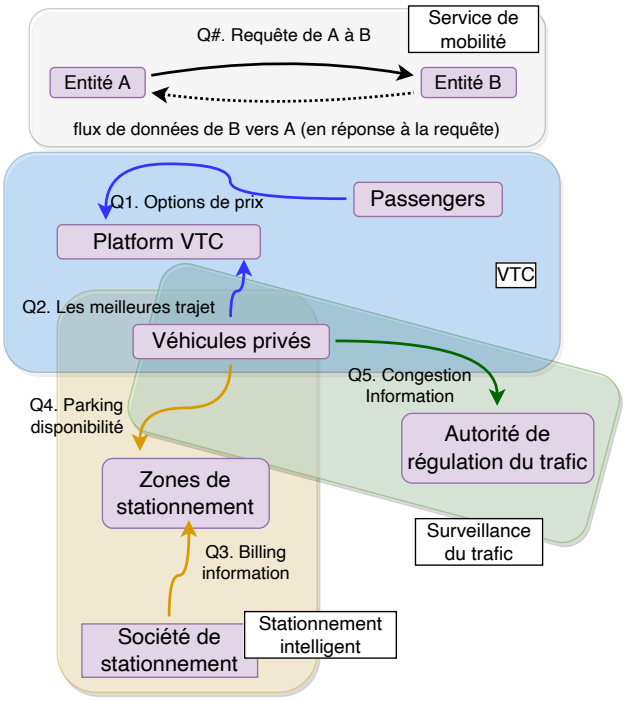


Figure 2: Aperçu des services

modèles de systèmes existants pour les IdO se concentrent sur une seule application ou un seul service et sur l'infrastructure qui lui est nécessaire [107]. Alors que notre système est un écosystème formellement défini de services qui peuvent à la fois fournir et consommer des données vers et depuis d'autres services.

Nous exploitons également notre construction pour répondre à une question secondaire : le Réseaux Longue Distance à Faible Consommation peut-il être utilisé comme moyen de communication pour les services multi-services ? Comment ces services doivent-ils être déployés afin de gérer ce trafic ?

Proposition de modèle multiservice pour le transport intelligent

Les services qui composent notre modèle multiservice sont les suivants :

Voiture de Transport avec Chauffeur (VTC) où des plateformes mettent en relation des passagers avec des conducteurs de véhicules (dit: au taxi à la demande).

Stationnement intelligent où des capteurs surveillent les places de stationnement occupées, enregistrant le moment où les véhicules arrivent et quittent les places.

Surveillance du trafic où le ITS mesure la vitesse moyenne pour chaque segment de route en capturant des instantanés de la vitesse des véhicules.

La figure 3.1 illustre notre scénario, où les différentes applications et entités partagent des informations entre elles.

Nous nous appuyons sur une version légèrement étendue de SQL pour définir explicitement les requêtes (*c.-à-d.*, les flèches sur la Fig. 3.1) qui déclenchent les flux de données entre les entités (dans le sens inverse des requêtes). En outre, nous faisons correspondre les données IdO au paradigme de la base de données relationnelle en considérant les sources de données, les échantillons et les attributs comme des tables, des lignes et des colonnes, respectivement. Nous ajoutons ensuite simplement les mots-clés suivants dans notre langage de requête étendant SQL pour décrire les flux d'IdO : **COMPUTE** indique une liste de calculs à appliquer aux données en séquence ; **EVENT** décrit une ou plusieurs conditions qui peuvent activer/désactiver un flux (mesures de capteurs locaux et messages distants) ; **EVERY** indique quand générer de nouvelles données pour le flux ; **UNTIL** : indique la condition pour arrêter le flux.

Services VTC

Requête 1 Les passagers demandent à plusieurs plateformes (*c.-à-d.*, intermédiaire) les meilleures cotations pour effectuer une course en spécifiant l'heure de départ, et le lieu géographique de départ et d'arrivée.

```
SELECT ride_platform, cost_to($destination) as ride_cost
FROM ride_platforms
ORDER BY ride_cost DESC
LIMIT 5
EVERY minute
UNTIL EVENT client_decision
```

Query 1: Demande du passager envoyée à un ensemble de courtiers.

Requête 2 Les chauffeurs s'inscrivent sur les plateformes pour recevoir les courses non satisfaites lorsqu'ils sont sur le point de terminer leur course en cours.

```
SELECT ride_begin, ride_end, estimated_price
FROM ride_platforms
ORDER BY ride_score(*) DESC
LIMIT 10
EVERY 5 minutes
UNTIL EVENT driver_decision
```

Query 2: Un conducteur demande les passagers disponibles.

Service de stationnement intelligent

Requête 3 Surveillance de l'occupation d'une ou plusieurs places de stationnement et envoi de notifications lorsqu'un véhicule entre ou sort d'une place.

```
SELECT event_type, time_stamp, car_plate, parking_spot_id
FROM parking_parking
EVERY EVENT park_begin, park_end
```

Query 3: Suivi des entrées de véhicules.

Requête 4 Les conducteurs demandent la disponibilité des aires de stationnement à proximité pour trouver un parking disponible.

```
SELECT parking_area_id,  available_count, distance(arrival_location
    , parking_location) as geo_distance
FROM parking_areas
WHERE available_count > THRESHOLD_VALUE
ORDER BY geo_distance DESC
LIMIT 10
EVERY minute
UNTIL EVENT end_of_trip
```

Query 4: Disponibilité du parking.

Service intelligent d'acheminement du trafic routier

Requête 5 service collectant les mesures de localisation et de vitesse d'un grand nombre de véhicules (*p.ex.*, bus, voitures privées, taxis) pour dresser un profil de la congestion en temps réel.

```
SELECT hash(car_plate), latitude, longitude, current_speed
FROM consenting_vehicles
EVERY 3 minutes
```

Query 5: Informations sur la congestion routière.

Modélisation des services avec des ensembles de données ouvertes sur la mobilité

Nous avons exploité les deux ensembles de données suivants :

VTC: nous utilisons un ensemble de données sur les trajets effectués par les sociétés Uber, Juno, Lyft et Via à New York City (NYC).

Parking usage: la ville de Melbourne fournit des mesures de son large déploiement de capteurs de stationnement.

Quelques ajustements sont nécessaires pour que ces deux ensembles de données soient compatibles entre eux et avec notre modèle. Tout d'abord, adapter les localisations géographiques puisque le service décrit par ces deux jeux de données est déployé dans des villes différentes. Nous rassemblons une liste de parkings à NYC pour cartographier les parkings à Melbourne. Nous les cartographions en fonction du volume de trafic et de leur proximité avec Manhattan (le centre commercial de NYC). Ensuite, pour imiter le déplacement des véhicules pour la requête 5, nous étendons l'ensemble de données NYC en calculant les trajectoires directes entre les lieux de ramassage et de dépôt pour chaque trajet.

Nous modélisons les requêtes 1 et 2 en supposant que les passagers demandent des options de prix au cours d'un trajet et que les conducteurs demandent de nouveaux trajets dès qu'ils en ont terminé un. Nous modélisons les événements de la requête 3 avec les données d'arrivée et de départ observées dans l'ensemble de données de Melbourne. Nous modélisons le temps inter-requêtes de la requête 4 en supposant qu'il y a eu une recherche de parking disponible à chaque fois qu'une place de parking a été prise. Dans la requête 5, nous considérons qu'un sous-ensemble de véhicules envoie ses informations de vitesse (*p.ex.*, un sous-ensemble de véhicules est équipé pour cela ou qui consent à ce que ses données soient utilisées).

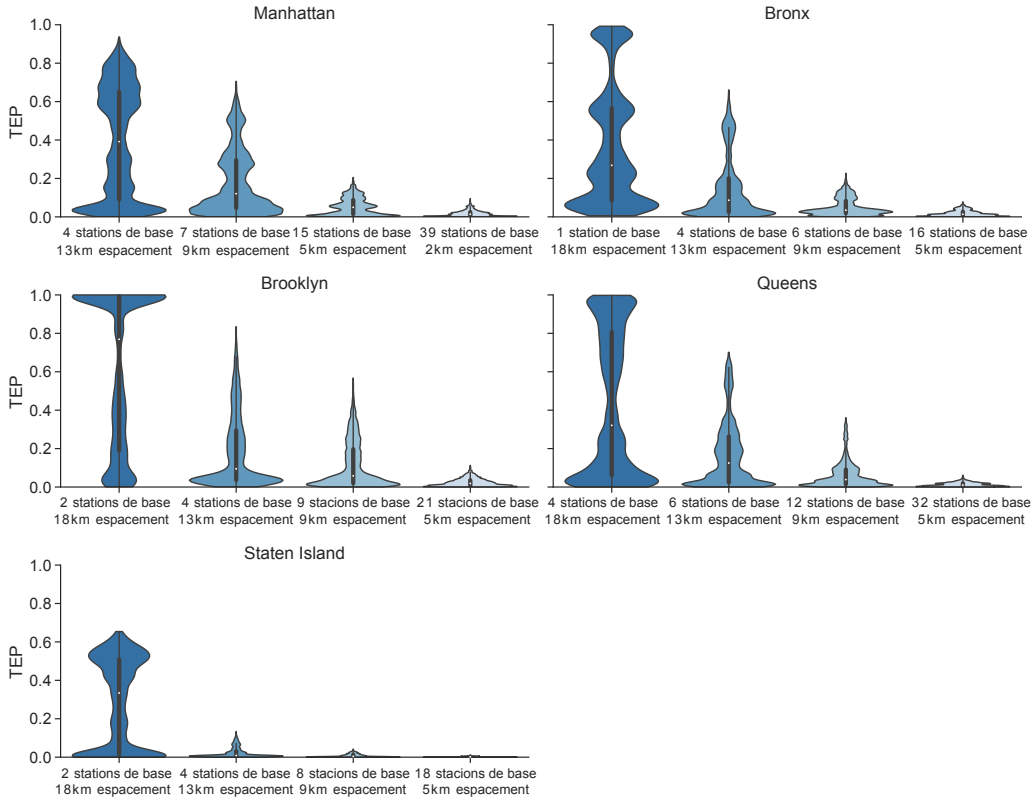


Figure 3: PER des flux sous différentes densités de stations de base.

Analyse de l'évolutivité avec un déploiement LoRa

LoRa est un réseau cellulaire où les appareils se connectent à des stations de base (dit LoRa gateways). Les dispositifs sont capables de transmettre aux stations de base jusqu'à une douzaine de kilomètres de distance et de profiter de débits binaires de l'ordre du kilobit par seconde. Pour mener notre analyse d'évolutivité, nous considérons le modèle analytique LoRa décrit dans [116] : pour une quantité donnée de trafic réseau, il prédit le RAP de chaque appareil associé à une passerelle LoRa.

Nous pouvons maintenant nous appuyer sur notre modèle avec les ensembles de données réelles présentés précédemment. Pour tenir compte des fluctuations temporelles du trafic réseau, nous choisissons 200 intervalles de temps aléatoires d'une heure, et utilisons le nombre moyen de messages par seconde pendant chacun de ces intervalles. En considérant des échantillons de temps suffisamment indépendants, nous sommes en mesure d'étudier l'évolutivité de LoRa pour ces services de mobilité intelligents. Ensuite, nous positionnons nos stations de base dans un format de grille.

Nous étudions la densité de stations de base requise par nos services. Bien que chaque passerelle soit théoriquement capable de desservir des nuds situés jusqu'à 9 km de distance, le trafic est trop élevé et le réseau qui en résulte devient encombré. Ainsi, nous considérons un espacement de 5 km pour Manhattan, de 13 km pour

Staten Island et de 9 km pour tous les autres arrondissements afin d'atteindre le Taux d'Erreur de Paquets (TEP) affiché dans la figure 3 pour chaque zone. S'il est nécessaire de réduire encore plus le TEP, un schéma d'optimisation localisé doit être utilisé pour déployer des stations de base supplémentaires dans les zones encombrées (zones rouges).

Conclusions

Nous avons défini ici un scénario de mobilité intelligente globale, où plusieurs services cohabitent tout en étant supportés par des passerelles LoRa. Nous avons défini un scénario de mobilité intelligente globale dans lequel plusieurs services cohabitent en étant supportés par des passerelles LoRa. Nous avons ensuite émulé le trafic généré par un tel scénario en utilisant des jeux de données disponibles publiquement, collectés respectivement à NYC et Melbourne. Bien que notre analyse d'évolutivité tende à indiquer que le fait de s'appuyer uniquement sur LoRa est suffisant et donc envisageable pour un déploiement ITS unifié, nous notons que le nombre de passerelles LoRa doit être soigneusement ajusté en fonction du trafic. Dans l'ensemble, la capacité du réseau est atteinte rapidement, ce qui peut créer des problèmes d'évolutivité pour les scénarios à grande échelle.

Les résultats de ce travail mettent en évidence les besoins des scénarios multiservices et multi-propriétaires. Malheureusement, nous n'utilisons pas ce cadre dans les évaluations de nos propositions suivantes. Nous nous concentrons plutôt sur les infrastructures à plus grande échelle, ce qui est trop limité pour mettre nos propositions à l'épreuve. Notre proposition suivante est une architecture qui met en uvre des calculs permettant la confidentialité lorsque les données sont diffusées entre les services.

Superposition RDN pour la confidentialité et la réutilisation dans l'IdO multi-domaine

Dans ce chapitre, nous abordons le problème de la recherche d'une méthode permettant d'apporter une communication évolutive et respectueuse de la vie privée à une infrastructure d'IdO à grande échelle composée de dispositifs de plusieurs propriétaires déployés pour différents objectifs. Nous proposons une interconnexion virtuelle (overlay) de plusieurs réseaux d'IdO basée sur des relations fournisseur/-client où les routeurs d'RDN prennent en charge la communication entre les réseaux. Ces politiques définissent les règles du plan de contrôle qui décrivent quel flux de données peut être exporté et comment il doit être traité en premier. Notre méthode assure la confidentialité sans les mécanismes de chiffrement complexes habituellement employés dans de tels scénarios.

Domaines, Requêtes & Agrégation

IdO réseaux informatiques peuvent naturellement être délimités par des frontières virtuelles définies par des critères tels que la propriété, l'application et l'objectif de déploiement. Nous désignons ces groupes de dispositifs par le terme ***domaines***. Selon cette définition, tout dispositif peut échanger librement des données

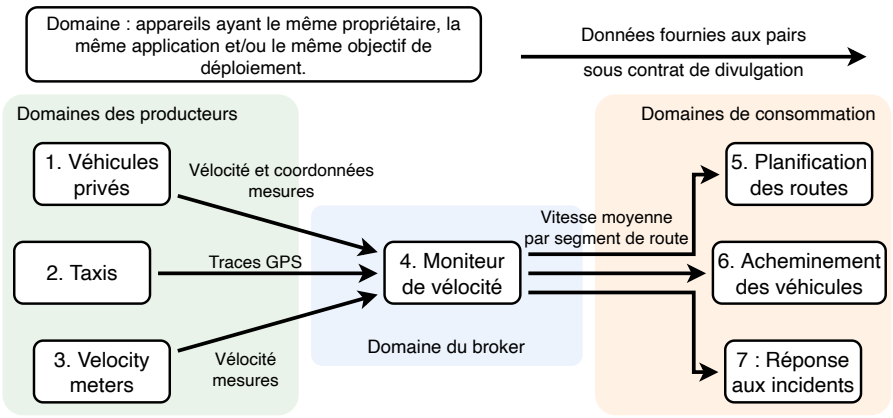


Figure 4: Exemple de transport dans une ville intelligente

puisqu'elles appartiennent au même propriétaire ou qu'elles ont été consenties pour être utilisées par cette application.

Prenons par exemple la topologie logique illustrée dans la Figure 4 dérivée de notre modèle du chapitre 3. Le domaine 4 (moniteur de vélocité) est un ***subscriber*** pour les données diffusées par les trois domaines de gauche, et ***publishes*** un flux agrégé pour les trois domaines de droite.

Dans notre modèle, un consommateur n'est pas intéressé par les données de producteurs spécifiques, il est plutôt intéressé par un type de données. Ainsi, une requête décrit les données souhaitées sur la base d'informations telles que le type (*p.ex.*, température, vitesse, puissance), la cardinalité (le nombre d'échantillons et de capteurs), la fréquence et la localisation. En outre, les consommateurs demandent normalement des données prétraitées au lieu des échantillons bruts des capteurs. Les calculs souhaités incluent normalement une certaine forme de fonction d'agrégation, *c.-à-d.*, une fonction qui résume les informations sur un groupe de valeurs, afin de réduire la verbosité des ensembles de données.

Énoncé du problème : Flux inter-domaines respectueux de la vie privée

Nous supposons que les domaines sont curieux mais honnêtes, *c.-à-d.*, qu'ils peuvent lire et profiter de toutes les informations auxquelles ils ont accès mais ne s'écarteront pas du protocole. Le cryptage serait le moyen direct de retenir ces domaines curieux. Ensuite, l'agrégation en réseau obligerait les domaines à déchiffrer, à appliquer des transformations et enfin à déchiffrer le résultat. Le cryptage homomorphique résoudrait ce problème, mais nous l'ignorons car il est limité à certaines transformations [54]. Ainsi, nous cherchons à assurer la confidentialité sans nous appuyer nécessairement sur le chiffrement.

Des transformations telles que l'agrégation et le filtrage peuvent être exploitées afin de fournir des niveaux de confidentialité similaires à ceux des ensembles de données anonymes. Nous supposons que les domaines de peering se font confiance, *c.-à-d.*, qu'un domaine peut s'attendre à ce que ses pairs (voisins) respectent les politiques de confidentialité sur lesquelles ils se sont mis d'accord.

Proposition : Une architecture multi-domaine RDN

Pour permettre l'échange de jeux de données entre plusieurs domaines, nous proposons de construire une superposition de ***border routers***. Ces routeurs sont chargés de toutes les communications inter-domaines en gérant les flux et en appliquant les exigences de confidentialité. La superposition qui les relie est une topologie logique, construite sur la base des relations de confiance entre les domaines. ***Dans ce chapitre, nous supposons que les domaines forment des arbres*** dans l'overlay.

Notez que les transformations sont au cur de notre architecture multi-domaine. Nous modélisons les exigences de confidentialité (politique d'exportation des données) comme une collection de transformations (d'anonymisation) qui doivent être appliquées à l'ensemble de données exporté. Nous faisons également une distinction entre les pairs directs, qui se font mutuellement confiance, et le reste du réseau, atteint de manière transitive, où des transformations supplémentaires peuvent être appliquées.

Dans notre proposition, chaque requête est composée de la description du jeu de données avec la chaîne de transformations souhaitée. En outre, l'usage et le domaine de consommation sont ajoutés aux paramètres des intérêts pour vérifier la conformité aux politiques de confidentialité. Lorsqu'un routeur frontalier reçoit une requête, il doit vérifier que le flux de données résultant respecte la politique d'exportation. Une requête est acceptée si la chaîne de transformation demandée est au moins aussi restrictive que la chaîne définie dans la politique. Si elle est acceptée, les données sont traitées selon la chaîne demandée et transmises au consommateur.

Nous choisissons d'implémenter cette superposition tout en nous appuyant sur le paradigme RDN en raison de ses avantages pour l'IdO. Nous nous appuyons sur les fonctions unikernel [69], qui sont des fonctions récupérables par nom (comme les paquets de données RDN) et s'exécutent au-dessus d'une virtualisation légère. Nous étendons également les mécanismes de cache de l'RDN pour détecter les ensembles de données qui se chevauchent afin d'utiliser les données partiellement traitées. Différents intérêts émis par différents consommateurs peuvent réutiliser les mêmes flux, au moins partiellement. Si un intérêt nécessite une chaîne de transformations qui est un sur-ensemble de la chaîne d'un intérêt auquel on a déjà répondu, le contenu du cache est directement réutilisé.

Évaluation des performances

Afin d'évaluer les avantages de notre solution, nous évaluons ses performances à l'aide de simulations. Nous considérons des requêtes où les consommateurs sont intéressés par la valeur moyenne d'une collection de mesures. Nous implémentons et comparons notre approche avec le RDN conventionnel. Au moment de la rédaction de cet article, il s'agissait de la meilleure comparaison que nous pouvions fournir puisque la protection de la vie privée par l'application de l'anonymisation et de l'agrégation n'est pas largement étudiée par la communauté des chercheurs.

Nos figures affichent les métriques suivantes pour évaluer notre solution en plusieurs dimensions : **Taille des magasins de contenu** mesure la quantité de données dans le magasin de contenu de chaque nud (dit la taille du cache). **Charge du réseau** mesure la somme des données transmises par tous les dispositifs, afin

de quantifier à la fois les besoins en bande passante et la consommation de batterie utilisés par les flux ; **Délai d'installation normalisé** mesure le temps entre la transmission de l'intérêt du consommateur et l'arrivée du premier paquet de données pour le flux correspondant. **Data spread** compte le nombre de routeurs RDN qui stockent chaque morceau de données.

Ensuite, nous mesurons la charge réseau pour tous les routeurs (*Fig. 5*). Bien que chaque fragment de données puisse être diffusé efficacement dans le réseau conventionnel RDN, cela ne suffit pas à limiter la surcharge du réseau. Notre superposition de transformation est également capable de réduire de manière significative la charge du réseau.

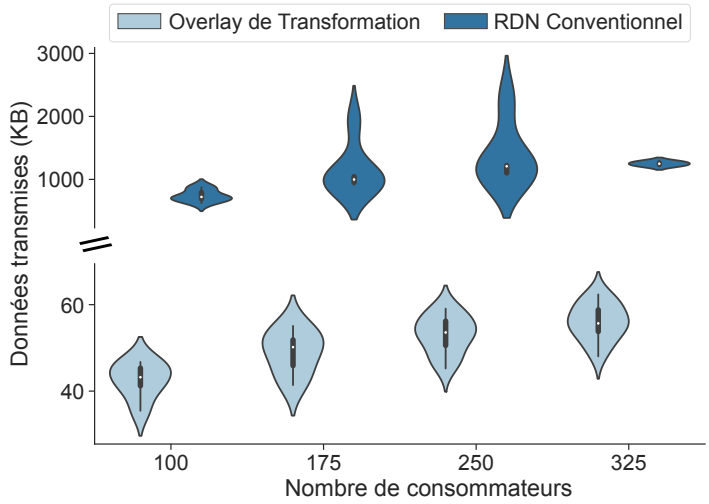


Figure 5: Transmissions de données sur les routeurs frontaliers : notre superposition de transformation s'adapte bien à la charge du réseau pour un grand nombre de consommateurs (et de demandes).

Figure 6 se concentre sur les caractéristiques de confidentialité. L'approche RDN conventionnelle diffuse les données brutes dans tous les réseaux, la diffusion étant significative même si les routes les plus courtes sont utilisées. La propagation est limitée par notre superposition basée sur la transformation qui transforme les données avant de les envoyer au consommateur ou au niveau suivant de l'arbre.

Conclusion

Nous avons présenté une nouvelle solution respectueuse de la vie privée pour échanger en toute sécurité des flux de données privées dans des réseaux IdO multi-domaines. Les données sont transformées et mises en cache avant de quitter le domaine afin que les préoccupations en matière de confidentialité soient respectées tout en permettant la réutilisation à l'aide de magasins de contenu intelligents qui examinent les chevauchements entre les intérêts parmi les ensembles de données transformés. Nos simulations mettent en évidence l'évolutivité de notre solution.

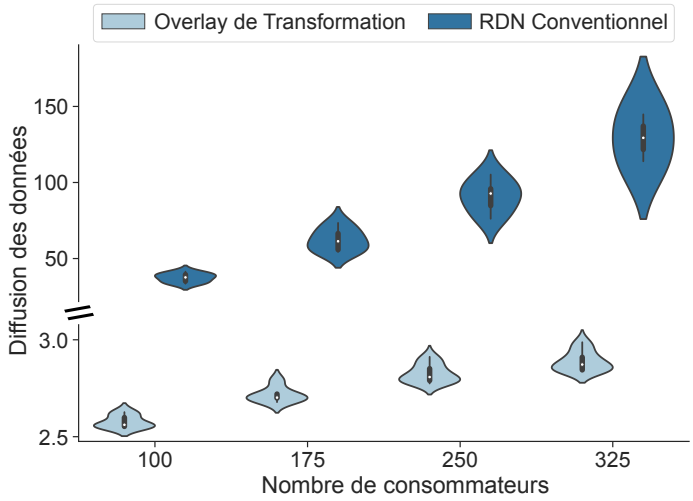


Figure 6: Diffusion du contenu dans le réseau.

Routage conforme à la confidentialité pour l'agrégation interdomaine

Dans ce chapitre, nous allons plus loin dans la direction du chapitre précédent en abandonnant la topologie arborescente préexistante. Les domaines peuvent alors échanger avec n'importe quel autre domaine comme ils le souhaitent. La superposition se transforme alors en un maillage, les domaines se fournissant mutuellement des données. En effet, les données peuvent être transmises par cycles ou arriver à la même destination par des routes alternatives. L'agrégation des données nécessite d'explorer des ensembles de données qui ne se croisent pas tout en vérifiant les exigences minimales de confidentialité, ici basées sur le k-anonymat. Nous verrons ici que nous obtenons de plus grandes agrégations en faisant tourner les données en boucle dans la superposition (sans chevauchement des ensembles de données) afin d'acquérir suffisamment de producteurs pour atteindre les exigences minimales d'agrégation et être ensuite diffusées dans le réseau.

Structure et exigences de la superposition maillée

Nous modélisons les relations entre les domaines sous la forme d'un graphe orienté $G(V, E)$ où chaque sommet est un réseau IdO et une arête $u \to v$ existe si le réseau u est un fournisseur de données à son pair v. Ces entités sont capables d'agir comme des courtiers entre d'autres entités en diffusant des informations acquises auprès d'autres entités.

Nous considérons que les données sont suffisamment privées pour être diffusées dans le réseau lorsque, selon les exigences de leur producteur, elles sont suffisamment agrégées. Nos exigences de confidentialité sont basées sur la k-anonyme qui détermine que les données ne peuvent pas être distinguées de k-1 autres données. Plus précisément, les pairs directs peuvent acquérir des données sans agrégation, en raison d'un accord mutuel. Cependant, ces pairs ne sont pas autorisés à transmet-

tre les données tant qu'elles ne répondent pas aux exigences de confidentialité des producteurs.

Notre objectif est de créer des agrégations valides dans un environnement multi-domaines. Nous proposons que les trois propriétés suivantes soient préservées lors du calcul des données agrégées : Seules les données d'intérêt doivent être impliquées ; Appliquer des ensembles de données anonymes non intersectés ; Les exigences d'agrégation doivent être préservées.

Notre proposition s'appuie sur notre architecture basée sur le RDN en ajoutant les mécanismes permettant aux domaines d'agréger correctement les données. Notre système se comporte comme un système de publication/abonnement. Pendant la publication, les routeurs diffusent les combinaisons valides de producteurs. Ensuite, lors de l'abonnement, le consommateur est capable d'identifier les producteurs qui correspondent à une requête, et de construire un ensemble valide de flux avec les combinaisons découvertes lors de l'étape de publication.

Notre méthode maintient une agrégation correcte grâce à l'utilisation d'ID de producteurs qui identifient les données utilisées dans les agrégations tout en étant dissociés des producteurs spécifiques. Un producteur associe son ID de producteur et des attributs descriptifs (*c.-à-d.*, métadonnées) aux données qu'il génère. Nous proposons de fusionner les ensembles de données qui contiennent des producteurs similaires.

Phase de publication

Chaque routeur frontalier doit identifier les ensembles de données qu'il peut publier. La publication consiste à filtrer et/ou transformer les offres d'agrégation provenant des voisins entrants (offres d'entrée) et à exporter les offres d'agrégation vers les voisins sortants (offres de sortie).

Une offre d'agrégation est une paire $o = (AS, r)$ où AS désigne l'ensemble des ID des producteurs qui sont proposés dans l'offre o et r désigne l'exigence minimale de ces producteurs dans l'offre o. Une telle offre indique que toute combinaison de producteurs $C \subseteq AS$ peut être activée tant qu'elle est conforme à l'exigence minimale, *c.-à-d.*, $|C| \geq r$.

Les producteurs divulguent initialement leur jeu de données sous la forme d'une offre dont AS ne contient que leur identifiant local et r est une exigence propre. Ces offres ne peuvent pas être diffusées car elles ne respectent pas les contraintes de confidentialité des producteurs. Nous étendons ces offres en utilisant les identifiants des producteurs d'autres offres. Le routeur doit s'assurer que ces ID ne se superposent pas afin d'éviter les répétitions. De plus, les données sont plus susceptibles d'être sélectionnées par la même requête si elles sont similaires. Lors de la publication, nous limitons la création d'offres dont la similarité entre les producteurs de AS est supérieure à une certaine valeur seuil minimale.

Phase d'abonnement

Un consommateur doit exploiter les offres d'agrégation disponibles reçues pendant la publication pour répondre à une requête. Un routeur frontalier extrait l'ensemble des producteurs qui correspondent aux critères de la requête, puis le consommateur doit sélectionner un ensemble d'offres à utiliser pour acquérir l'agrégation souhaitée.

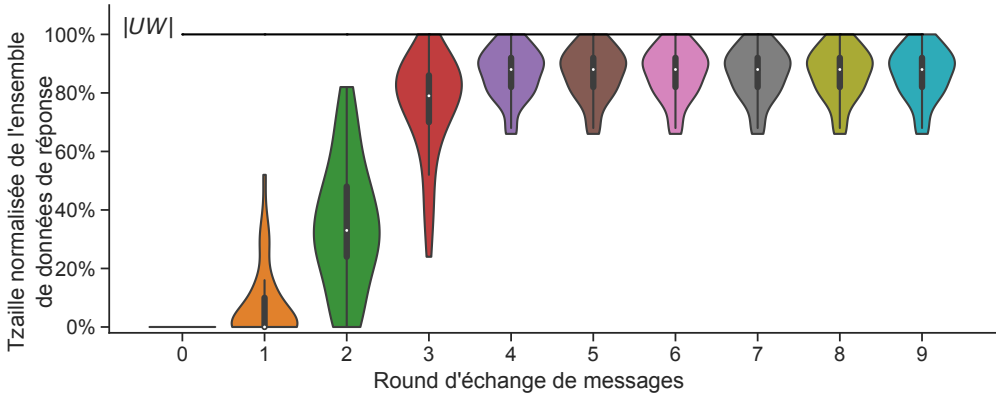


Figure 7: Capacité d'agrégation à différents tours.

La maximisation du nombre de producteurs dans cette combinaison permet au consommateur de collecter un ensemble de données plus riche. Le défi est alors d'identifier des offres qui ne se croisent pas, et telles que leur union maximise le nombre de producteurs. De la même manière que pour le problème de publication, nous devons trouver des cliques maximales dans le graphe de conflit afin de maximiser le nombre de producteurs proposés au consommateur. Il s'agit donc également d'un problème NP-Hard car plusieurs combinaisons de producteurs peuvent être possibles.

Nous proposons ici deux règles d'arrêt pour notre heuristique : s'arrêter lorsque nous atteignons le nombre minimum de producteurs requis (nP_{min}) ; et imposer également une limite nC_{max} de combinaisons de producteurs explorées.

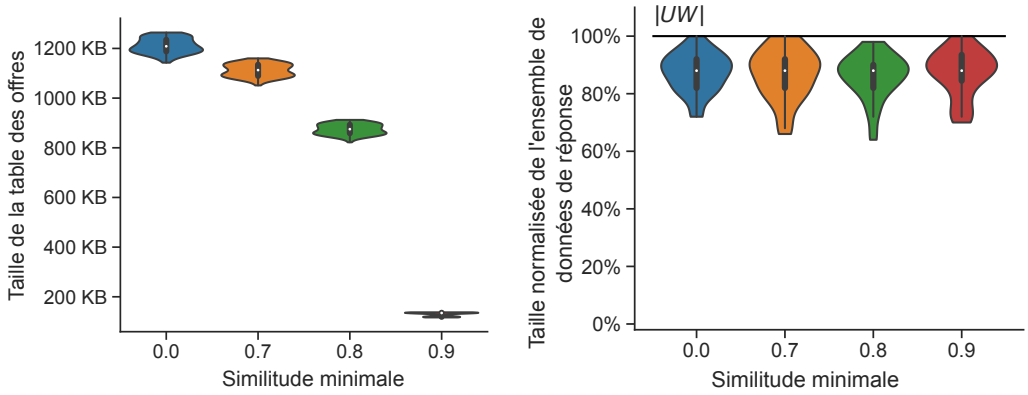
Évaluation des performances

Pour évaluer la performance de notre proposition, nous l'évaluons sur des topologies denses composées de nombreux producteurs attachés à plusieurs domaines de courtiers. Les producteurs fournissent leurs informations à plusieurs courtiers qui acquièrent les données et, à leur tour, diffusent les informations agrégées résultantes à d'autres courtiers (et à leurs clients). Il en résulte des informations biaisées générées artificiellement (ensembles de producteurs qui se croisent).

Nous supposons ici que l'espace descriptif des producteurs est simplement les coordonnées physiques normalisées (un espace carré dont le côté est égal à 1). La similarité entre les métadonnées est supposée être le complément de la distance entre les producteurs (en ce qui concerne leurs emplacements physiques en 2D) par rapport à l'excentricité de l'espace euclidien bidimensionnel (*c.-à-d.*, la distance maximale dans notre carré en pratique).

6.2.3 Convergence de l'étape de publication

Nous étudions d'abord la convergence de la phase de publication, en mesurant la taille moyenne du jeu de données à la fin de chaque tour (Fig. 7). Notre système semble converger rapidement : chaque courtier est capable de compléter les offres non conformes. Le nombre de tours dépend à la fois de l'excentricité de l'espace



(a) (b)

Figure 8: Effets de Δ_{min} sur les métriques

Figure 9: Effects of Δ_{min} on metrics

(sa distance maximale) et de l'exigence d'agrégation minimale, qui peut générer un ensemble plus important d'offres incomplètes.

Concentrons-nous maintenant dans la Figure 9 sur l'impact de la métrique de similarité, et en particulier du seuil de similarité. Lorsque le courtier ne tient pas compte de la similarité lors de la fusion de différentes offres ($\Delta_{min} = 0$), le nombre d'offres dans la table est maximal. Il est intéressant de noter que le nombre d'offres peut diminuer de manière significative (de 85Ce qui est vraiment intéressant, c'est que la fusion d'offres similaires (dans notre cas, provenant de producteurs géographiquement proches) a un impact négligeable sur la phase d'abonnement (Fig. 8b).

Conclusions

Cette thèse a exploré les problèmes de confidentialité liés à l'interrogation de données provenant d'environnements IoT multi-propriétaires. Nous basons notre modèle de confidentialité sur des domaines : des groupes de dispositifs IoT dans le même périmètre de propriété, d'objectif, et/ou de localisation. Les limites de la vie privée prennent forme après les frontières de ces domaines, car nous supposons que les données sont autorisées à transiter librement au sein d'un domaine en raison de la propriété ou du consentement.

Notre première contribution (Chapitre 3) définit un modèle multi-domaine dans lequel les domaines échangent des données entre eux par le biais d'une approche évolutive afin de garantir la confidentialité dès la conception. Notre deuxième contribution (Chapitre 4) concerne une architecture pour la communication inter-domaines que nous avons basée sur la pile réseau Named Data Networking.

L'architecture proposée se distingue des autres propositions de protection de la vie privée au sein de réseaux IoT indépendants qui utilisent principalement des systèmes de cryptage complexes pour verrouiller les données sous différents niveaux de sécurité. Nous étudions l'utilisation de l'anonymisation pour maximiser la réu-

tilisation des données afin de répondre à de multiples requêtes et garantir la confidentialité. Notre évaluation des performances montre que nous obtenons de grands gains de performance avec la réutilisation de données anonymes, même si nous ne fournissons pas une opacité totale.

Notre troisième contribution (Chapitre 5) améliore notre contribution précédente en permettant aux domaines de s'auto-organiser dans une topologie de superposition plus complexe. Nous nous attendons à ce que les domaines se fournissent mutuellement des données simplement sur la base de relations commerciales, ce qui entraînerait probablement la vente des mêmes données à plusieurs domaines. Cela pourrait entraîner une collecte et une agrégation répétées des données si celles-ci sont anonymes et indiscernables. Ainsi, nous fournissons un schéma de publication pour diffuser des ensembles de données et créer de manière incrémentielle des combinaisons de producteurs qui sont conformes aux restrictions de confidentialité des producteurs et sans entrées de données répétées. Nous proposons également un système d'abonnement qui utilise les informations acquises lors de la publication pour répondre aux requêtes basées sur les métadonnées.

Nos résultats montrent que nous atteignons rapidement la limite supérieure d'agrégation possible tout en respectant les exigences minimales d'agrégation. Ils montrent également que les exigences matérielles de notre schéma sont bien en deçà des capacités des dispositifs à ressources limitées.

Les contributions de cette thèse nous donnent un aperçu du problème de la confidentialité via l'anonymisation pour les communications inter-domaines.

Bibliography

[1] Renato Juaçaba Neto, Pascal Mérindol, and Fabrice Théoleyre. "Data Aggregation for Privacy Protection of Data Streams Between Autonomous IoT Networks". In: ***Symposium on Computers and Communications (ISCC)***. IEEE, 2021, pp. 1–6. DOI: 10.1109/ISCC53001.2021.9631401.

[2] Renato Juaçaba Neto, Pascal Mérindol, Antoine Gallais, and Fabrice Théoleyre. "Scalability of LPWAN for Smart City Applications". In: ***International Wireless Communications and Mobile Computing Conference (IWCMC)***. 2021, pp. 74–79. DOI: 10.1109/IWCMC51323.2021.9498698.

[3] Renato Juaçaba Neto, Pascal Mérindol, and Fabrice Théoleyre. "Transformation Based Routing Overlay for Privacy and Reusability in Multi-Domain IoT". In: ***International Symposium on Network Computing and Applications (NCA)***. IEEE, 2020, pp. 1–8. DOI: 10.1109/NCA51143.2020.9306709.

[4] Renato Juaçaba Neto, Pascal Mérindol, and Fabrice Théoleyre. "A Multi-Domain Framework to Enable Privacy for Aggregated IoT Streams". In: ***Conference on Local Computer Networks (LCN)***. IEEE. 2020, pp. 401–404. DOI: 10.1109/LCN48667.2020.9314825.

[5] ***Ericsson mobility report***. Tech. rep. Ericsson, 2021.

[6] Shanzhi Chen, Hui Xu, Dake Liu, Bo Hu, and Hucheng Wang. "A Vision of IoT: Applications, Challenges, and Opportunities With China Perspective". In: ***IEEE Internet of Things Journal*** 1.4 (2014), pp. 349–359. ISSN: 2327-4662.

[7] Microsoft. ***IoT Signals***. Tech. rep. October. 2021, p. 57.

[8] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi. "Internet of Things for Smart Cities". In: ***IEEE Internet of Things Journal*** 1.1 (2014), pp. 22–32. ISSN: 2372-2541.

[9] Giancarlo Fortino, Claudio Savaglio, Carlos E. Palau, Jara Suarez de Puga, Maria Ganzha, Marcin Paprzycki, Miguel Montesinos, Antonio Liotta, and Miguel Llop. "Towards Multi-layer Interoperability of Heterogeneous IoT Platforms: The INTER-IoT Approach". In: 2018, pp. 199–232. ISBN: 9783319613000.

[10] Keiichi Yasumoto, Hirozumi Yamaguchi, and Hiroshi Shigeno. "Survey of Real-time Processing Technologies of IoT Data Streams". In: ***Journal of Information Processing*** 24.2 (2016), pp. 195–202. ISSN: 1882-6652.

[11] Andrew A. Cook, Goksel Misirli, and Zhong Fan. "Anomaly Detection for IoT Time-Series Data: A Survey". In: ***IEEE Internet of Things Journal*** 7.7 (2020), pp. 6481–6494. ISSN: 2327-4662.

[12] Ajinkya Prabhune, Rainer Stotzka, Vaibhav Sakharkar, Jürgen Hesser, and Michael Gertz. "MetaStore: an adaptive metadata management framework for heterogeneous metadata models". In: ***Distributed and Parallel Databases*** 36.1 (2018), pp. 153–194. ISSN: 15737578.

[13] M Shona and B N Arathi. "A Survey on the Data Management in IoT". In: ***International Journal of Scientific and Technical Advancements*** 2.1 (2016), pp. 261–264. ISSN: 2454-1532.

[14] Shusen Yang. "IoT Stream Processing and Analytics in the Fog". In: ***IEEE Communications Magazine*** 55.8 (2017), pp. 21–27. ISSN: 0163-6804.

[15] Daniel Le Métayer. "A Formal Privacy Management Framework". In: ***Formal Aspects in Security and Trust***. Springer, 2009, pp. 162–176.

[16] Latanya Sweeney. "Weaving Technology and Policy Together to Maintain Confidentiality". In: ***Journal of Law, Medicine & Ethics*** 25.2-3 (1997), pp. 98–110. ISSN: 1073-1105. DOI: 10.1111/j.1748-720X.1997.tb01885.x.

[17] Sharu Bansal and Dilip Kumar. "IoT Ecosystem: A Survey on Devices, Gateways, Operating Systems, Middleware and Communication". In: ***International Journal of Wireless Information Networks*** 27.3 (2020), pp. 340–364. ISSN: 1068-9605.

[18] Priyank Sunhare, Rameez R. Chowdhary, and Manju K. Chattopadhyay. "Internet of things and data mining: An application oriented survey". In: ***Journal of King Saud University - Computer and Information Sciences*** (2020). ISSN: 13191578.

[19] Navid Nikaein, Markus Laner, Kaijie Zhou, Philipp Svoboda, Dejan Drajic, Milica Popovic, and Srdjan Krco. "Simple traffic modeling framework for machine type communication". In: ***Proceedings of the International Symposium on Wireless Communication Systems*** (2013), pp. 783–787. ISSN: 21540225.

[20] Padma Balaji Leelavinodhan, Massimo Vecchio, Fabio Antonelli, Andrea Maestrini, and Davide Brunelli. "Design and Implementation of an Energy-Efficient Weather Station for Wind Data Collection". In: ***Sensors*** 21.11 (2021), p. 3831. ISSN: 1424-8220.

[21] Nicolas Museux and Jef Vanbockryck. "Event based heterogeneous sensors fusion for public place surveillance". In: ***2007 10th International Conference on Information Fusion***. IEEE, 2007, pp. 1–8. ISBN: 978-0-662-45804-3.

[22] Gilles Callebaut, Guus Leenders, Jarne Van Mulders, Geoffrey Ottoy, Lieven De Strycker, and Liesbet Van der Perre. "The Art of Designing Remote IoT DevicesTechnologies and Strategies for a Long Battery Life". In: ***Sensors*** 21.3 (2021), p. 913. ISSN: 1424-8220.

[23] Agnius Liutkevicius, Arunas Vrubliauskas, and Egidijus Kazanavicius. "A survey of wireless sensor network interconnection to external networks". In: ***Novel Algorithms and Techniques in Telecommunications and Networking***. Springer, 2010, pp. 41–46. ISBN: 9789048136629.

[24] Klaus Hartke. ***Observing resources in the constrained application protocol (CoAP)***. Tech. rep. 2015.

[25] Yali Wang, Yang Zhang, and Junliang Chen. "An SDN-based publish/subscribe-enabled communication platform for IoT services". In: ***China Communications*** 15.1 (2018), pp. 95–106. ISSN: 16735447.

[26] Pin Lv, Licheng Wang, Huijun Zhu, Wenbo Deng, and Lize Gu. "An IOT-oriented privacy-preserving publish/subscribe model over blockchains". In: ***IEEE Access*** 7 (2019), pp. 41309–41314. ISSN: 21693536.

[27] Michele Amoretti, Riccardo Pecori, Yanina Protskaya, Luca Veltri, and Francesco Zanichelli. "A Scalable and Secure Publish/Subscribe-Based Framework for Industrial IoT". In: ***IEEE Transactions on Industrial Informatics*** 17.6 (2021), pp. 3815–3825. ISSN: 19410050.

[28] Olivier Blazy, Emmanuel Conchon, Mathieu Klingler, and Damien Sauveron. "An IoT Attribute-Based Security Framework for Topic-Based Publish/-Subscribe Systems". In: ***IEEE Access*** 9 (2021), pp. 19066–19077. ISSN: 21693536.

[29] Biswajeeban Mishra and Attila Kertesz. "The use of MQTT in M2M and IoT systems: A survey". In: ***IEEE Access*** 8 (2020), pp. 201071–201086.

[30] Zach Shelby, Klaus Hartke, and Carsten Bormann. ***The constrained application protocol (CoAP)***. rfc 7252. IETF, 2014.

[31] Dinesh Thangavel, Xiaoping Ma, Alvin Valera, Hwee-Xian Tan, and Colin Keng-Yan Tan. "Performance evaluation of MQTT and CoAP via a common middleware". In: ***2014 IEEE Ninth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)***. April. IEEE, 2014, pp. 1–6. ISBN: 978-1-4799-2843-9. DOI: 10.1109/ISSNIP.2014.6827678.

[32] World Wide Web Consortium. ***SPARQL 1.1 Overview***. Tech. rep. 2013.

[33] Fabiano Ferreira Luz and Marcelo Finger. "Semantic Parsing Natural Language into SPARQL: Improving Target Language Representation with Neural Attention". In: ***CoRR*** abs/1803.04329 (2018). arXiv: 1803.04329.

[34] Haemin Jung and Wooju Kim. "Automated conversion from natural language query to SPARQL query". In: ***Journal of Intelligent Information Systems*** 55.3 (2020), pp. 501–520. ISSN: 1573-7675.

[35] Xiaoyu Yin, Dagmar Gromann, and Sebastian Rudolph. "Neural machine translating from natural language to SPARQL". In: ***Future Generation Computer Systems*** 117 (2021), pp. 510–519. ISSN: 0167-739X.

[36] Frank Manola, Eric Miller, Brian McBride, et al. ***RDF primer***. Tech. rep. 2004.

[37] Manu Sporny, Dave Longley, Gregg Kellogg, Markus Lanthaler, and Niklas Lindström. ***JSON-LD 1.0***. Tech. rep. 2014.

[38] Pratikkumar Desai, Amit Sheth, and Pramod Anantharam. "Semantic Gateway as a Service Architecture for IoT Interoperability". In: ***IEEE ICMS***. Vol. 32. 2. IEEE. IEEE, 2015, pp. 313–319. ISBN: 978-1-4673-7284-8.

[39] T. Kiruthiga and N. Shanmugasundaram. "In-network Data Aggregation Techniques for Wireless Sensor Networks: A Survey". In: ***Lecture Notes on Data Engineering and Communications Technologies***. Vol. 66. 2021, pp. 887–905.

[40] Rajesh Kumar Yadav and Monica Gupta. "Data Aggregation Algorithms in IoT: An Organized Evaluation of The Literature". In: ***2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT)***. Icssit. IEEE, 2020, pp. 300–304. ISBN: 978-1-7281-5821-1.

[41] Atif Alamri, Wasai Shadab Ansari, Mohammad Mehedi Hassan, M. Shamim Hossain, Abdulhameed Alelaiwi, and M. Anwar Hossain. "A Survey on Sensor-Cloud: Architecture, Applications, and Approaches". In: ***International Journal of Distributed Sensor Networks*** 9.2 (2013), p. 917923. ISSN: 1550-1477. DOI: 10.1155/2013/917923.

[42] Sunil Kumar Singh, Prabhat Kumar, and Jyoti Prakash Singh. "A Survey on Successors of LEACH Protocol". In: ***IEEE Access*** 5 (2017), pp. 4298–4328. ISSN: 2169-3536.

[43] Sun Limin, Li Jianzhong, Chen Yu, and Zhu Hongsong. "Wireless sensor networks". In: ***Beijing: Tsinghua University Press*** 5 (2005), pp. 7–8.

[44] Zahra Beiranvand, Ahmad Patooghy, and Mahdi Fazeli. "I-LEACH: An efficient routing algorithm to improve performance & to reduce energy consumption in Wireless Sensor Networks". In: ***The 5th Conference on Information and Knowledge Technology***. IEEE, 2013, pp. 13–18. ISBN: 978-1-4673-6490-4.

[45] Jia Xu, Ning Jin, Xizhong Lou, Ting Peng, Qian Zhou, and Yanmin Chen. "Improvement of LEACH protocol for WSN". In: ***2012 9th International Conference on Fuzzy Systems and Knowledge Discovery***. 2012, pp. 2174–2177.

[46] Yong Yao and Johannes Gehrke. "The cougar approach to in-network query processing in sensor networks". In: ***ACM Sigmod record*** 31.3 (2002), pp. 9–18.

[47] Amin Shahraki, Amir Taherkordi, Øystein Haugen, and Frank Eliassen. "Clustering objectives in wireless sensor networks: A survey and research direction analysis". In: ***Computer Networks*** 180.June (2020). ISSN: 13891286.

[48] Tim Winter, Pascal Thubert, Anders Brandt, Jonathan W Hui, Richard Kelsey, Philip Levis, Kris Pister, Rene Struik, Jean-Philippe Vasseur, Roger K Alexander, et al. "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks." In: ***rfc*** 6550 (2012), pp. 1–157.

[49] Samuel Madden, Michael J Franklin, Joseph M Hellerstein, and Wei Hong. "TAG: A Tiny Aggregation Service for Ad-Hoc Sensor Networks". In: ***5th Symposium on Operating Systems Design and Implementation (OSDI 02)***. 2002.

[50] Stephanie Lindsey, Cauligi Raghavendra, and Krishna M Sivalingam. "Data gathering algorithms in sensor networks using energy metrics". In: ***IEEE Transactions on parallel and distributed systems*** 13.9 (2002), pp. 924–935.

[51] Suman Nath, Phillip B Gibbons, Srinivasan Seshan, and Zachary Anderson. "Synopsis diffusion for robust aggregation in sensor networks". In: ***ACM Transactions on Sensor Networks (TOSN)*** 4.2 (2008), pp. 1–40.

[52] Zvika Brakerski and Vinod Vaikuntanathan. "Efficient Fully Homomorphic Encryption from (Standard) LWE". In: ***SIAM Journal on Computing*** 43.2 (2014), pp. 831–871. ISSN: 0097-5397.

[53] Rongxing Lu, Kevin Heung, Arash Habibi Lashkari, and Ali A Ghorbani. "A Lightweight Privacy-Preserving Data Aggregation Scheme for Fog Computing-Enhanced IoT". In: ***IEEE Access*** 5 (2017), pp. 3302–3312.

[54] Michael Naehrig, Kristin Lauter, and Vinod Vaikuntanathan. "Can Homomorphic Encryption Be Practical?" In: ***Proceedings of the 3rd ACM Workshop on Cloud Computing Security Workshop***. CCSW '11. Chicago, Illinois, USA: Association for Computing Machinery, 2011, pp. 113124. ISBN: 9781450310048.

[55] Lixia Zhang, Alexander Afanasyev, Jeffrey Burke, Van Jacobson, Kc Claffy, Patrick Crowley, Christos Papadopoulos, Lan Wang, and Beichuan Zhang. "Named data networking". In: ***ACM SIGCOMM Computer Communication Review*** 44.3 (2014), pp. 66–73. ISSN: 01464833.

[56] Marica Amadeo, Claudia Campolo, Jose Quevedo, Daniel Corujo, Antonella Molinaro, Antonio Iera, Rui L. Aguiar, and Athanasios V. Vasilakos. "Information-centric networking for the internet of things: challenges and opportunities". In: ***IEEE Network*** 30.2 (2016), pp. 92–100.

[57] Ravi Ravindran, Yanyong Zhang, Luigi Grieco, Anders Lindgren, Jeff Burke, Bengt Ahlgren, and Aytac Azgin. ***Design Considerations for Applying ICN to IoT***. Tech. rep. ICN Research Group, 2019, pp. 1–51.

[58] Sobia Arshad, Muhammad Awais Azam, Mubashir Husain Rehmani, and Jonathan Loo. "Recent Advances in Information-Centric Networking-Based Internet of Things (ICN-IoT)". In: ***IEEE Internet of Things Journal*** 6.2 (2019), pp. 2128–2158. ISSN: 2327-4662. eprint: 1710.03473.

[59] Wassim Drira and Fethi Filali. "A Pub/Sub extension to NDN for efficient data collection and dissemination in V2X networks". In: ***Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014***. IEEE, 2014, pp. 1–7. ISBN: 978-1-4799-4786-7.

[60] Rute C. Sofia and Paulo M. Mendes. "An Overview on Push-Based Communication Models for Information-Centric Networking". In: ***Future Internet*** 11.3 (2019), p. 74. ISSN: 1999-5903.

[61] Marica Amadeo, Claudia Campolo, and Antonella Molinaro. "Internet of Things via Named Data Networking: The support of push traffic". In: ***International Conference and Workshop on the Network of the Future (NOF)***. IEEE. IEEE, 2014, pp. 1–5. ISBN: 978-1-4799-5280-9.

[62] Jiachen Chen, Mayutan Arumaithurai, Lei Jiao, Xiaoming Fu, and K. K. Ramakrishnan. "COPSS: An efficient content oriented publish/subscribe system". In: ***ANCS***. ACM/IEEE. 2011.

[63] S. Ananthakrishnan, Mohit P. Tahiliani, Deepaknath Tandur, and Hariram Satheesh. "Group based Publisher-Subscriber Communication Primitives for ndnSIM". In: ***2020 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)***. Vol. 2020-Decem. IEEE, 2020, pp. 1–6. ISBN: 978-1-7281-9290-1.

[64] Minsheng Zhang, Vince Lehman, and Lan Wang. "Scalable name-based data synchronization for named data networking". In: ***IEEE INFOCOM 2017 - IEEE Conference on Computer Communications***. IEEE, 2017, pp. 1–9. ISBN: 978-1-5090-5336-0.

[65] Divya Gupta, Shalli Rani, Syed Hassan Ahmed, and Rasheed Hussain. "Caching Policies in NDN-IoT Architecture". In: ***Integration of WSN and IoT for Smart Cities***. Springer, 2020, pp. 43–64. DOI: 10.1007/978-3-030-38516-3_3.

[66] John P. Baugh and Jinhua Guo. "A Per-Face Popularity Scheme to Increase Cache Robustness in Information-Centric Networks". In: ***Procedia Computer Science*** 134 (2018), pp. 267–274. ISSN: 18770509. DOI: 10.1016/j.procs.2018.07.170.

[67] Serdar Vural, Pirabakaran Navaratnam, Ning Wang, Chonggang Wang, Lijun Dong, and Rahim Tafazolli. "In-network caching of Internet-of-Things data". In: ***2014 IEEE International Conference on Communications (ICC)***. 2014, pp. 3185–3190. DOI: 10.1109/ICC.2014.6883811.

[68] Manolis Sifalakis, Basil Kohler, Christopher Scherb, and Christian Tschudin. "An Information Centric Network for Computing the Distribution of Computations". In: ***Conference on Information-Centric Networking (ICN)***. ACM. Paris, France, 2014, pp. 137–146.

[69] Michał Król and Ioannis Psaras. "NFaaS: Named Function as a Service". In: ***ACM ICN***. ACM. Berlin, Germany: Association for Computing Machinery, 2017, pp. 134–144. ISBN: 9781450351225.

[70] C. Scherb, D. Grewe, M. Wagner, and C. Tschudin. "Resolution strategies for networking the IoT at the edge via named functions". In: ***Annual Consumer Communications Networking Conference (CCNC)***. IEEE. 2018.

[71] Marica Amadeo, Giuseppe Ruggeri, Claudia Campolo, and Antonella Molinaro. "IoT Services Allocation at the Edge via Named Data Networking: From Optimal Bounds to Practical Design". In: ***IEEE Transactions on Network and Service Management*** 16.2 (2019), pp. 661–674. ISSN: 1932-4537.

[72] Aqeel Kazmi, Martin Serrano, and John Soldatos. "VITAL-OS: An Open Source IoT Operating System for Smart Cities". In: ***IEEE Communications Standards Magazine*** 2.2 (2018), pp. 71–77. ISSN: 2471-2825.

[73] Ken Figueredo, Dale Seed, and Vanja Subotic. "Preparing for Highly Scalable and Replicable IoT Systems". In: ***IEEE Internet of Things Magazine*** 3.3 (2020), pp. 94–98. ISSN: 2576-3180.

[74] Andy Stanford-Clark and Hong Linh Truong. "Mqtt for sensor networks (mqtt-sn) protocol specification". In: ***International business machines (IBM) Corporation version*** 1.2 (2013), pp. 1–28.

[75] H. W. Chen and F. J. Lin. "Converging MQTT Resources in ETSI Standards Based M2M Platform". In: ***International Conference on Internet of Things (iThings)***. 2014, pp. 292–295.

[76] Pietro Colombo and Elena Ferrari. "Access Control Enforcement Within MQTT-based Internet of Things Ecosystems". In: ***Symposium on Access Control Models and Technologies***. ACM. Indianapolis, Indiana, USA, 2018, pp. 223–234.

[77] L. Rodrigues, J. Guerreiro, and N. Correia. "RELOAD/CoAP architecture with resource aggregation/disaggregation service". In: ***International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)***. IEEE. 2016.

[78] C. Gündoğan, P. Kietzmann, T. C. Schmidt, and M. Wählisch. "HoPP: Robust and Resilient Publish-Subscribe for an Information-Centric Internet of Things". In: ***Conference on Local Computer Networks (LCN)***. IEEE. 2018, pp. 331–334.

[79] Deutsche Telekom IoT. ***NB-IoT, LoRaWAN, Sigfox: An up-to-date comparison***. Tech. rep. 2021, p. 22.

[80] Alexandru Lavric, Adrian I. Petrariu, and Valentin Popa. "SigFox Communication Protocol: The New Era of IoT?" In: ***2019 International Conference on Sensing and Instrumentation in IoT Era (ISSI)***. 2019, pp. 1–4.

[81] Renato Juaçaba Neto, Emanuel Bezerra Rodrigues, and Carina Teixeira de Oliveira. "Performance Analysis of Resource Unit Configurations for M2M Traffic in the Narrowband-IoT System". In: ***Simpósio Brasileiro de Telecomunicações e Processamento de Sinais (SBrT)***. 2017.

[82] Nicolas Sornin et al. "LoRaWAN Specification". In: ***LoRa alliance*** (2015).

[83] Javier Lopez, Ruben Rios, Feng Bao, and Guilin Wang. "Evolving privacy: From sensors to the Internet of Things". In: ***Future Generation Computer Systems*** 75 (2017), pp. 46–57. ISSN: 0167739X.

[84] Lionel Metongnon, Ramin Sadre, and Eugene C. Ezin. "Distributed Middlebox Architecture for IoT Protection". In: ***International Conference on Network and Service Management (CNSM)***. IEEE. IEEE, 2019, pp. 1–7. ISBN: 978-3-903176-24-9.

[85] Carla Mouradian, Diala Naboulsi, Sami Yangui, Roch H. Glitho, Monique J. Morrow, and Paul A. Polakos. "A Comprehensive Survey on Fog Computing: State-of-the-Art and Research Challenges". In: ***IEEE Communications Surveys and Tutorials*** 20.1 (2018), pp. 416–464. ISSN: 1553877X.

[86] Nadine Kashmar, Mehdi Adda, Mirna Atieh, and Hussein Ibrahim. "A review of access control metamodels". In: ***Procedia Computer Science*** 184.2019 (2021), pp. 445–452. ISSN: 18770509.

[87] Seham Alnefaie, Asma Cherif, and Suhair Alshehri. "Towards a Distributed Access Control Model for IoT in Healthcare". In: ***2019 2nd International Conference on Computer Applications Information Security (ICCAIS)***. 2019, pp. 1–6.

[88] Mohammad Wazid, Ashok Kumar Das, Vanga Odelu, Neeraj Kumar, Mauro Conti, and Minho Jo. "Design of Secure User Authenticated Key Management Protocol for Generic IoT Networks". In: ***IEEE Internet of Things Journal*** 5.1 (2018), pp. 269–282. ISSN: 2327-4662.

[89] Pradnya Patil, M. Sangeetha, and Vidhyacharan Bhaskar. "Blockchain for IoT Access Control, Security and Privacy: A Review". In: ***Wireless Personal Communications*** 117.3 (2021), pp. 1815–1834. ISSN: 0929-6212.

[90] Shantanu Pal, Ali Dorri, and Raja Jurdak. "Blockchain for IoT access control: Recent trends and future research directions". In: ***Journal of Network and Computer Applications*** (2022), p. 103371. ISSN: 10848045. eprint: 2106.04808.

[91] Shantanu Pal, Tahiry Rabehaja, Ambrose Hill, Michael Hitchens, and Vijay Varadharajan. "On the Integration of Blockchain to the Internet of Things for Enabling Access Right Delegation". In: ***IEEE Internet of Things Journal*** 7.4 (2020), pp. 2630–2639. ISSN: 2327-4662.

[92] Josep Domingo-Ferrer and Jordi Soria-Comas. "Data Anonymization". In: ***Risks and Security of Internet and Systems***. Ed. by Javier Lopez, Indrajit Ray, and Bruno Crispo. Cham: Springer, 2015, pp. 267–271. ISBN: 978-3-319-17127-2.

[93] Isabel Wagner and David Eckhoff. "Technical Privacy Metrics: A Systematic Survey". In: ***ACM Computing Surveys*** 51.3 (2018), 57:1–57:38. ISSN: 0360-0300.

[94] LATANYA SWEENEY. "k-ANONYMITY: A MODEL FOR PROTECTING PRIVACY". In: ***International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*** 10.05 (2002), pp. 557–570.

[95] Cynthia Dwork. "Differential privacy". In: ***International Colloquium on Automata, Languages, and Programming***. Springer. 2006, pp. 1–12.

[96] Julián Salas and Vicenç Torra. "A General Algorithm for k-anonymity on Dynamic Databases". In: ***Data Privacy Management, Cryptocurrencies and Blockchain Technology***. Ed. by Joaquin Garcia-Alfaro, Jordi Herrera-Joancomartí, Giovanni Livraga, and Ruben Rios. Cham: Springer International Publishing, 2018, pp. 407–414. ISBN: 978-3-030-00305-0.

[97] Karuna Arava and Sumalatha Lingamgunta. "Adaptive k-Anonymity Approach for Privacy Preserving in Cloud". In: ***Arabian Journal for Science and Engineering*** 45.4 (2020), pp. 2425–2432. ISSN: 2193-567X.

[98] Wantong Zheng, Zhongyue Wang, Tongtong Lv, Yong Ma, and Chunfu Jia. "K-Anonymity Algorithm Based on Improved Clustering". In: ***Algorithms and Architectures for Parallel Processing***. Ed. by Jaideep Vaidya and Jin Li. Cham: Springer International Publishing, 2018, pp. 462–476. ISBN: 978-3-030-05054-2.

[99] Woo-Seok Choi, Matthew Tomei, Jose Rodrigo Sanchez Vicarte, Pavan Kumar Hanumolu, and Rakesh Kumar. "Guaranteeing Local Differential Privacy on Ultra-Low-Power Systems". In: ***2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture (ISCA)***. 2018, pp. 561–574.

[100] Alexander Krall, Daniel Finke, and Hui Yang. "Gradient Mechanism to Preserve Differential Privacy and Deter Against Model Inversion Attacks in Healthcare Analytics". In: ***2020 42nd Annual International Conference of the IEEE Engineering in Medicine Biology Society (EMBC)***. 2020, pp. 5714–5717.

[101] Chuan Ma, Long Yuan, Li Han, Ming Ding, Raghav Bhaskar, and Jun Li. "Data Level Privacy Preserving: A Stochastic Perturbation Approach based on Differential Privacy". In: ***IEEE Transactions on Knowledge and Data Engineering*** (2021), pp. 1–1.

[102] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. "Our Data, Ourselves: Privacy Via Distributed Noise Generation". In: ***Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)***. Vol. 4004 LNCS. 2006, pp. 486–503. ISBN: 3540345469.

[103] Elaine Shi, Hubert Chan, Eleanor Rieffel, Richard Chol, and Dawn Song. "Privacy-Preserving Aggregation of Time-Series Data". In: ***Network & Distributed System Security Symposium***. San Diego, California, 2011, p. 17.

[104] Ajesh Koyatan Chathoth, Abhyuday Jagannatha, and Stephen Lee. "Federated Intrusion Detection for IoT with Heterogeneous Cohort Privacy". In: ***CoRR*** abs/2101.09878 (2021). arXiv: `2101.09878`.

[105] Jie Lin, Wei Yu, Nan Zhang, Xinyu Yang, Hanlin Zhang, and Wei Zhao. "A Survey on Internet of Things: Architecture, Enabling Technologies, Security and Privacy, and Applications". In: ***IEEE Internet of Things Journal*** 4.5 (2017), pp. 1125–1142. ISSN: 2327-4662.

[106] Cristina Marquez et al. “How Should I Slice My Network? A Multi-Service Empirical Evaluation of Resource Sharing Efficiency”. In: ***MobiCom***. ACM. New Delhi, India, 2018, pp. 191–206. ISBN: 9781450359030.

[107] Ritika Lohiya and Ankit Thakkar. “Application Domains, Evaluation Data Sets, and Research Challenges of IoT: A Systematic Review”. In: ***IEEE Internet of Things Journal*** 8.11 (2021), pp. 8774–8798. ISSN: 2327-4662.

[108] Jorge Pérez, Marcelo Arenas, and Claudio Gutierrez. “Semantics and complexity of SPARQL”. In: ***ACM Transactions on Database Systems*** 34.3 (2009), pp. 1–45. ISSN: 0362-5915.

[109] Danh Le-Phuoc, Minh Dao-Tran, Josiane Xavier Parreira, and Manfred Hauswirth. “A Native and Adaptive Approach for Unified Processing of Linked Streams and Linked Data”. In: ***The Semantic Web (ISWC)***. Ed. by Lora Aroyo, Chris Welty, Harith Alani, Jamie Taylor, Abraham Bernstein, Lalana Kagal, Natasha Noy, and Eva Blomqvist. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 370–388. ISBN: 978-3-642-25073-6.

[110] N. Bizanis and F. A. Kuipers. “SDN and Virtualization Solutions for the Internet of Things: A Survey”. In: ***IEEE Access*** 4 (2016), pp. 5591–5606.

[111] Zhe Xu, Zhixin Li, Qingwen Guan, Dingshui Zhang, Qiang Li, Junxiao Nan, Chunyang Liu, Wei Bian, and Jieping Ye. “Large-Scale Order Dispatch in On-Demand Ride-Hailing Platforms”. In: ***International Conference on Knowledge Discovery & Data Mining***. New York, NY, USA: ACM, 2018, pp. 905–913. ISBN: 9781450355520.

[112] R. Mangiaracina et al. “Smart parking management in a smart city: Costs and benefits”. In: ***IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI)***. 2017.

[113] Mohammed Amine Falek et al. “To Re-Route, or not to Re-Route:Impact of Real-Time Re-Routing in Urban Road Networks”. In: ***Journal of Intelligent Transportation Systems: Technology, Planning, and Operations*** (2020).

[114] Hyunjae Kang, Dong Hyun Ahn, Gyung Min Lee, Jeong Do Yoo, Kyung Ho Park, and Huy Kang Kim. ***IoT network intrusion dataset***. 2019.

[115] Stephen Makonin, Bradley Ellert, Ivan V. Bajić, and Fred Popowich. “Electricity, water, and natural gas consumption of a residential house in Canada from 2012 to 2014”. In: ***Scientific Data*** 3.1 (2016), p. 160037. ISSN: 2052-4463.

[116] D. Bankov, E. Khorov, and A. Lyakhov. “Mathematical model of LoRaWAN channel access with capture effect”. In: ***IEEE PIMRC***. 2017.

[117] Kiril Zelenkovski, Filip Karafiloski, and Igor Mishkovski. “Riemann Garbage Bin: A Self-sustained Waste Management System”. In: May 2019.

[118] Branden Ghena et al. “Challenge: Unlicensed LPWANs Are Not Yet the Path to Ubiquitous Connectivity”. In: ***MobiCom***. ACM. Los Cabos, Mexico, 2019. ISBN: 9781450361699.

[119] Juan David Arias Correa, Alex Sandro Roschildt Pinto, and Carlos Montez. "Lossy Data Compression for IoT Sensors: A Review". In: ***Internet of Things*** 19.March (2022), p. 100516. ISSN: 25426605. DOI: 10.1016/j.iot.2022.100516.

[120] Caner Bektas, Stefan Bocker, Benjamin Sliwa, and Christian Wietfeld. "Rapid Network Planning of Temporary Private 5G Networks with Unsupervised Machine Learning". In: ***IEEE Vehicular Technology Conference*** 2021-September (2021). ISSN: 15502252.

[121] Zuozhou Li and Shudong Li. "LTE network planning based on game theory". In: ***2011 International Conference on Computer Science and Service System, CSSS 2011 - Proceedings*** (2011), pp. 3963–3966.

[122] Damien B. Jourdan and Olivier L. De Weck. "Layout optimization for a wireless sensor network using a Multi-Objective Genetic Algorithm". In: ***IEEE Vehicular Technology Conference*** 59.5 (2004), pp. 2466–2470. ISSN: 15502252.

[123] J. K. Han, B. S. Park, Y. S. Choi, and H. K. Park. "Genetic approach with a new representation for based station placement in mobile communications". In: ***IEEE Vehicular Technology Conference*** 4.54ND (2001), pp. 2703–2707. ISSN: 07400551.

[124] Roger Fletcher. ***Practical methods of optimization***. John Wiley & Sons, 2013.

[125] Tai Liu, Zain Tariq, Jay Chen, and Barath Raghavan. "The Barriers to Overthrowing Internet Feudalism". In: ***Proceedings of the 16th ACM Workshop on Hot Topics in Networks - HotNets-XVI***. 2017, pp. 72–79. ISBN: 9781450355698.

[126] Bing Li, Dijiang Huang, Zhijie Wang, and Yan Zhu. "Attribute-based Access Control for ICN Naming Scheme". In: ***IEEE Transactions on Dependable and Secure Computing*** 15.2 (2018), pp. 194–206. ISSN: 1545-5971.

[127] Marco Leo, Federica Battisti, Marco Carli, and Alessandro Neri. "A federated architecture approach for Internet of Things security". In: ***2014 Euro Med Telco Conference (EMTC)***. 2014, pp. 1–5.

[128] O. Wolfson, B. Xu, S. Chamberlain, and L. Jiang. "Moving objects databases: issues and solutions". In: ***Proceedings. Tenth International Conference on Scientific and Statistical Database Management (Cat. No.98TB100243)***. 1998, pp. 111–122.

[129] F. Cayre, C. Fontaine, and T. Furon. "Watermarking security: theory and practice". In: ***IEEE Transactions on Signal Processing*** 53.10 (2005), pp. 3976–3987.

[130] Yunpeng Zhang and Xuqing Wu. "Access Control in Internet of Things: A Survey". In: ***Journal of Network and Computer Applications*** 144.May 2018 (2016), pp. 79–101. ISSN: 10958592. eprint: 1610.01065.

[131] Jing Qiu, Zhihong Tian, Chunlai Du, Qi Zuo, Shen Su, and Binxing Fang. "A Survey on Access Control in the Age of Internet of Things". In: ***IEEE Internet of Things Journal*** 7.6 (2020), pp. 4682–4696. ISSN: 2327-4662.

[132] Josep Domingo-Ferrer, Sara Ricci, and Jordi Soria-Comas. "A Methodology to Compare Anonymization Methods Regarding Their Risk-Utility Tradeoff". In: ***International Conference on Modeling Decisions for Artificial Intelligence (MDAI)***. 2017, pp. 132–143. ISBN: 9783319674216.

[133] Onur Ascigil, Sergi Reñé, George Xylomenos, Ioannis Psaras, and George Pavlou. "A keyword-based ICN-IoT platform". In: ***Conference on Information-Centric Networking (ICN)***. ACM. 2017, pp. 22–28.

[134] S. Boubiche, D. E. Boubiche, A. Bilami, and H. Toral-Cruz. "Big Data Challenges and Data Aggregation Strategies in Wireless Sensor Networks". In: ***IEEE Access*** 6 (2018), pp. 20558–20571.

[135] In Lee. "The Internet of Things for enterprises: An ecosystem, architecture, and IoT service business model". In: ***Internet of Things*** 7 (2019), p. 100078. ISSN: 2542-6605.

[136] Latanya Sweeney. "k-Anonymity: a Model for Protecting Privacy". In: ***International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*** 10.05 (2002), pp. 557–570. ISSN: 0218-4885.

[137] Shyam Boriah, Varun Chandola, and Vipin Kumar. "Similarity Measures for Categorical Data: A Comparative Evaluation". In: ***International Conference on Data Mining***. Vol. 1. SIAM. 2008, pp. 243–254. ISBN: 978-0-89871-654-2.

[138] Richard M Karp. "Reducibility among combinatorial problems". In: ***Complexity of computer computations***. Springer, 1972, pp. 85–103.

[139] Paul Erdös and Alfréd Rényi. "On Random Graphs. I". In: ***Publicationes Mathematicae*** 6 (1959), pp. 290–297.

[140] P.D. Acharjya and Kauser Ahmed. "A Survey on Big Data Analytics: Challenges, Open Research Issues and Tools". In: ***International Journal of Advanced Computer Science and Applications*** 7.2 (2016). ISSN: 21565570. DOI: 10.14569/IJACSA.2016.070267.

[141] N. C. Eli-Chukwu, J. M. Aloh, and C. O. Ezeagwu. "A Systematic Review of Artificial Intelligence Applications in Cellular Networks". In: ***Engineering, Technology & Applied Science Research*** 9.4 (2019), pp. 4504–4510. ISSN: 1792-8036.

[142] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. "Practical Secure Aggregation for Privacy-Preserving Machine Learning". In: ***Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security***. ACM, 2017, pp. 1175–1191. ISBN: 9781450349468. DOI: 10.1145/3133956.3133982.

[143] Chen Zhang, Yu Xie, Hang Bai, Bin Yu, Weihong Li, and Yuan Gao. "A survey on federated learning". In: ***Knowledge-Based Systems*** 216 (2021), p. 106775. ISSN: 09507051. DOI: 10.1016/j.knosys.2021.106775.

Renato CAMINHA JUAÇABA NETO

Privacy-aware Aggregation of IoT Streams in Multi-Owner Networks

Résumé

Les déploiements actuels de systèmes IoT pour les villes intelligentes et les réseaux intelligents ont une forte densité d'appareils. Ce qui conduit les réseaux à rencontrer des problèmes car l'espace de déploiement et l'infrastructure réseau sont limités. Pour résoudre ce problème, les réseaux ouvrent leurs frontières et partagent leurs capteurs et services avec d'autres, permettant aux autres réseaux d'accéder à leurs capteurs et services internes. Cette thèse se concentre sur les préoccupations des producteurs en matière de confidentialité lorsque les données sont divulguées à d'autres réseaux. Ce qui est une conséquence des différents niveaux de consentement avec lesquels les données sont produites. Dans cette thèse, nous étudions comment fournir une collecte de données respectueuse de la vie privée entre différents systèmes tout en préservant les contraintes de chaque propriétaire de données. Nous nous concentrons sur l'utilisation de l'agrégation, c'est-à-dire les fonctions qui décrivent des groupes de données, par exemple la moyenne et le minimum. Pour cet objectif, nous fournissons trois contributions qui offrent un aperçu des moyens de diffuser des données agrégées dans une infrastructure IoT multi-propriétaires tout en respectant les contraintes des producteurs impliqués.

Résumé en anglaise

Current deployments of IoT systems for smart cities and smart grids have a high density of devices. Which lead networks to encounter issues as deployment space and network infrastructure are limited. To tackle this, networks open their boundaries and share their sensors and services to others, allowing the other networks to access their internal sensors and services. The focus of this thesis are the privacy concerns of producers as data is disclosed to other networks. Which is a consequence of the different consent levels that data is produced with. In this thesis we investigate how to provide privacy-aware data collection among different systems while preserving the constraints of each data owner. We focus on the usage of aggregation, that is, functions that describe groups of data, \eg average and minimum. For this objective we provide three contributions that offer insight on the means to stream aggregated data in a multi-owner IoT infrastructure while respecting the restraints of involved producers.