Transformation Based Routing Overlay for Privacy and Reusability in Multi-Domain IoT

Renato Caminha Juacaba Neto, Pascal Mérindol, Fabrice Theoleyre

Transformation Based Routing Overlay for Privacy and Reusability in Multi-Domain IoT

Renato Caminha Juacaba Neto, Pascal Merindol and Fabrice Theoleyre
ICube, CNRS / University of Strasbourg, France
{caminha,merindol,theoleyre}@unistra.fr

***Abstract*—The Internet of Things (IoT) interconnects a large collection of low power devices to the Internet. Instead of relying on the usual approach where IoT devices push their data to a cloud, we envision a decentralized approach, where multiple IoT domains cooperate to exchange data safely. As privacy is critical regarding IoT data, each domain should define what it accepts to export. We propose to create a multi-domain overlay of border routers, defining their own privacy constraints. Border routers export their available data streams and policies to their peering domains. Such policies define how streams should be aggregated and transformed to be compliant with the privacy requirements. We take advantage of the Named Data Networking (NDN) architecture to enable data re-usability and in-network transformations. Our evaluation highlights the scalability of our approach: NDN content stores supporting data aggregation and transformation reduce both the network and the cache load while enforcing privacy natively.**

***Keywords*—Internet of Things; interests; privacy, data anonymization; Named Data Networking.**

I. Introduction

The Internet of Things (IoT) has received much attention in the last decade. A collection of controllers, actuators and sensors allows cyber physical systems (CPS, [1]) to take smart decisions. For instance, a smart building is able to limit its carbon footprint by adapting its heating system to both occupancy fluctuations and actual temperatures reported in each part of the building [2].

IoT applications rely on filtering and aggregation operations to extract relevant information from the large number of sensors and data generated. While usual approaches assume that the raw data is directly pushed to a cloud infrastructure, we are convinced that this centralization jeopardizes the network scalability, and creates privacy concerns as a single external actor concentrates all the data. In contrast, edge computing brings processing operations closer to the producers, to reduce the network load, and the latency [3]. Such approaches are also promising to enforce data privacy needs since in-network processing may help to ensure some level of anonymity.

In the same vein, data centric approaches such as Named Data Networking (NDN) [4] have raised considerable attention as a novel communication paradigm for IoT networks. In this context, IoT applications can indeed naturally benefit from features such as semantic requests and in-network caching. Additionally, named function networking [5] represents a pioneering piece of work to provide in-network processing. This novel paradigm allows consumers to insert lambda expressions directly in their requests such that NDN routers are then in charge of both retrieving data and computing the desired expression.

978-1-7281-8326-8/20/$31.00 ©2020 IEEE

Currently, most complex CPS rely on multiple domains with different owners, that may accept to share *some* of their data only to other trusted domains. In theory, data may be ciphered between domains. However, sharing data among multiple domains rely on access control rules that are very complex to manage. Without a full knowledge of the different stakeholders, it is particularly challenging to deploy in practice. Thus, we rather aim to rely on NDN border routers that filter and pre-process data-streams when they exit a domain. Streams can be transformed (e.g. aggregated and filtered) before leaving the network/Typically, when the interests of the customers concern simple operations like applying an average function on several datasets of the same kind, there exists an opportunity to apply these operations inside the network rather than letting customers perform them on their own.

In this paper, we focus on this multi-domains IoT scenario. We re-use here our previous framework [6]: we proposed to extend the NDN paradigm by including aggregation functionalities in border routers in order to provide privacy as well as improving the network performance. We extend it to handle any kind of transformation on the top of an overlay of border routers.

The contributions of this paper are as follows:

1) we propose an overlay of NDN aware border routers interconnecting multiple IoT domains. They exchange their anonymized datasets on the basis of a simple policy-based inter-domain routing protocol. Such policies define control plane exporting rules that describe which data stream can be exported and how;
2) we define how to implement in-network transformations in the data plane. Data-streams can be aggregated and filtered before exiting a domain. Privacy rules defined in the exporting policies are efficiently enforced using in-network caching;
3) 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 to improve scalability while enforcing privacy by design.

II. Related Work

Privacy has always been a key challenge for IoT. Packets may be ciphered, but it requires to pre-deploy and manage keys, which is a challenging task in complex large scale dynamic environments [7]. We focus in this paper on multi-domain IoT applications, where pre-shared keys and global access control schemes are not practically feasible.

Privacy can typically be preserved for IoT streams by applying masking operations (*e.g.*, attribute removal, accuracy reduction, and entry removal) to respect quantifiable privacy metrics [8]. A K-anonymous dataset guarantees that it is impossible to use attributes to distinguish any entry from $K-1$ other entries. Similarly, ϵ-differential [9] guarantees that multiple disclosures of the same data set are sufficiently similar to prevent data interpolation.

We detail here related works that can support IoT streams and consider privacy.

A. Named Data Networking (NDN)

NDN is a clean-slate approach to the current Internet stack, which fits very well the needs for IoT application to handle and directly forward pieces of data. Hierarchical names replace numerical addresses and enable semantic requests, prefix-based routing, and route aggregation [4]. Consumers create **interest** packets, that describe with *names* the information they search for. This interest is then forwarded according to the prefix name specified in the interest.

Every router can cache the data it forwards in its *Content Store*. In that way, a router can use this cache to reply directly to any novel interest that matches the same piece of data. A signature mechanism allows each piece of data to be authenticated, *i.e.*, a router cannot forge a chunk of data by its own.

Unfortunately, some IoT requirements are not supported natively by NDN. For instance, a consumer may be interested in *any* measurement in a given geographical area (*e.g.*, a temperature anywhere inside a building), which is not supported natively by NDN. Ascigil *et al.* [10] proposes a keyword-based approach but this work does not support the acquisition of large datasets aggregated from multiple domains.

Additionally, IoT applications often rely on streams of data [11]. Thus, subscription-based mechanisms [12] have been proposed to extend NDN; persistent pending interest table entries are employed to allow the association of multiple data packets of a given stream to the same interest. We employ this approach to ease IoT stream re-usability.

B. In-network Processing and Middle-boxes

In IoT, one has to consider heterogeneous infrastructures that support multiple stacks. To facilitate this situation, an MQTT proxy may help to interconnect producers and consumers [13]. However, existing solutions propose to rely on access control strategies to control the dissemination of sensitive information [14]. Alternatively, we may apply an approach based on middleboxes, to interconnect different domains. Such as the concept of semantic gateways, in charge of relaying annotated data [15]. However, middleboxes do not address the privacy concerns of a multi-domain 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 [16]. However, their focus is rather on wrapping the different resources, instead of the privacy concerns raised by such multi-domain architectures. Similarly, content proxies serve as a central, stable entity to connect publishers and subscribers, but still, a multi-domain is not considered [17].

In-network processing was originally designed to reduce bandwidth consumption in IoT networks. In the NDN paradigm, Named Function Networking [5] proposes to integrate lambda-calculus in interests, where each NDN router may process the data before sending the reply. An extension has also been proposed to cope with edge computing architectures to preferentially select resources which are closer to producers [18]. This kind of in-network processing approach represents a promising tool for implementing privacy enforcing functions inside the network (privacy protection functionality can be regarded as processing requirements).

III. A NDN multi-domain overlay for Privacy

We re-use here the terminology that we have already defined in [6]:

- a **domain** represents a collection of devices that are possessed and administrated by the same owner (*e.g.*, a smart home) or involved in the same system (*e.g.*, heating);
- a **data stream** is a continuous flow of data generated by a producer, *e.g.*, a temporal sequence of measurements;
- a **dataset** represents a data stream that a domain accepts to export, with its characteristics encoded as metadata (providing its nature and cardinality in particular). For instance, a domain may export the average energy consumption for 10,000 smart meters, in flats with 4 inhabitants.

To enable the exchange of datasets between multiple domains, we propose to construct an overlay of border routers. A border router is basically 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 peering 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 consist, respectively, in removing personal data that can lead to any kind of identification and sending only the average value to further obfuscate precise data. We also make a distinction between direct peers, that can trust each others, and the rest of the network, reached transitively, where additional transformations may be enforced.

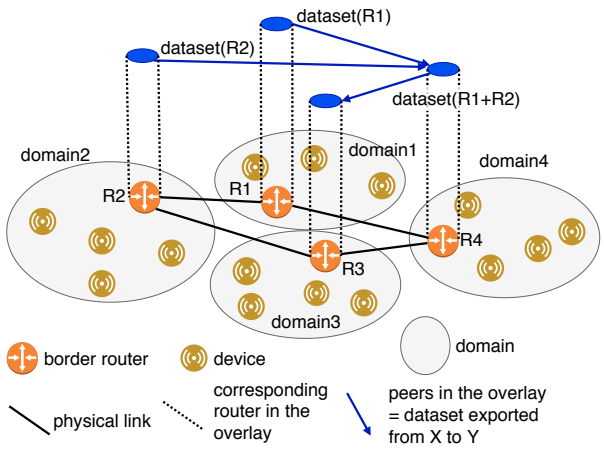

Fig. 1: Overlay on top of multiple domains. This overlay forms a logical tree for a given dataset (here customers in domain4 can directly access the entire set of producers).

Note that transformations based on data aggregation is at the core of our multi-domain architecture [6]. Indeed, such an aggregation not only hides individual data streams to respect privacy but also allows for scalability. Let us consider Figure 1: each domain has a border router in charge of exporting and importing data from/to outside. Typically, the overlay forms a tree due to the consumer-producer relationships between domains, *e.g.*, R4 is for instance a consumer for domain2, and a producer for domain3. Here R4 aggregates the data from two domains (domain1 and domain2 being here leafs of the tree). Such a tree is enough to capture well existing commercial relationship among domains. For example, a subscriber sends its electricity consumption to its electricity provider, that resells the anonymized set of data to a broker. Finally, the broker aggregates the data from different electricity providers to sell a large dataset to R&D / marketing / commercial entities. Besides, a tree structure prevents by design the occurrence of loops in the acquisition and transformation of data; indeed they may result in inconsistencies such as averaging the same data multiple times.

We choose to implement this overlay relying on the Named Data Networking (NDN) paradigm. This clean-slate approach is particularly relevant for IoT applications as a NDN router manipulates chunks of data and not anymore opaque packets. 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 [12]. 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. In that case, the border router detects the overlaps and uses its Content Store to save transmissions.

Our tree-based overlay defines the underlying control plane of our simple but efficient NDN routing scheme. Forks in such a structure actually emerge at the data-plane when a given NDN interest is decomposed into multiples because the necessary number of samples (the dataset cardinality) cannot be achieved considering only one child in the tree. We let the definition of more flexible and sophisticated routing schemes, using graph construct less strict than a single directed tree per dataset, for future work. The main challenge being about avoiding data duplication without 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.

A. 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 routers 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. To favor flexibility and enable the incremental deployment of relations between domains, border routers can form an overlay on top of the physical network, as illustrated in Fig. 1. This NDN overlay is a logical topology deployed on top of the physical one.

A domain first decides whether it aims to expose a portion of its data to its peering border routers. 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 form a novel one, exported to its peering domains. Globally, this overlay forms a directed tree that may not match the physical topology. In Figure 1, an arrow $(A \rightarrow B)$ in the overlay represents the dataset export 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).

In details, a border router executes the following tasks:

1) it collects data from its domain, 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 the data before it exits the domain. It uses directly its content store if the data is present, it adapts the interest otherwise (relaying the sub-interest(s) to its peers).

Each border router is identified by the domain name it belongs to. This way, we can blacklist some domains by just inserting a list of forbidden substrings in the prefix names. It is worth noting that a domain prefix does not have to be the base prefix of *all* the data it serves. For example, a border

router with the prefix */DOM2259* can still serve more generic prefixes, like */surrounding/temperature*.

Let us consider Fig. 2 which provides an overview of our solution. 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 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.

B. Policy engine

We exploit an overlay of border routers, where each domain selects what data it accepts to export, and with which (trusted) peering 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 an anonymization purpose. The novel aggregated stream can then be reused to answer several interests.

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 dataset (*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 basic 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. 2 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, that comprises:

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 [19]. Usages correspond typically to keywords (see [10]), 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 (cf. Fig. 2), 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 the 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.)

C. 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.

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. 2). 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.*, */Producer-Name/DataSet/$average?every=5minutes*. The exact naming for transformations and parameters are left as a 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, imported policies). Then, it selects a sufficient collection of datasets to answer the interest (that maps to a collection of peers).

In Figure 2, 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 modifies the initial interest packet by adapting the parameter describing the number of measurements in the two (sub-)interests it relays.

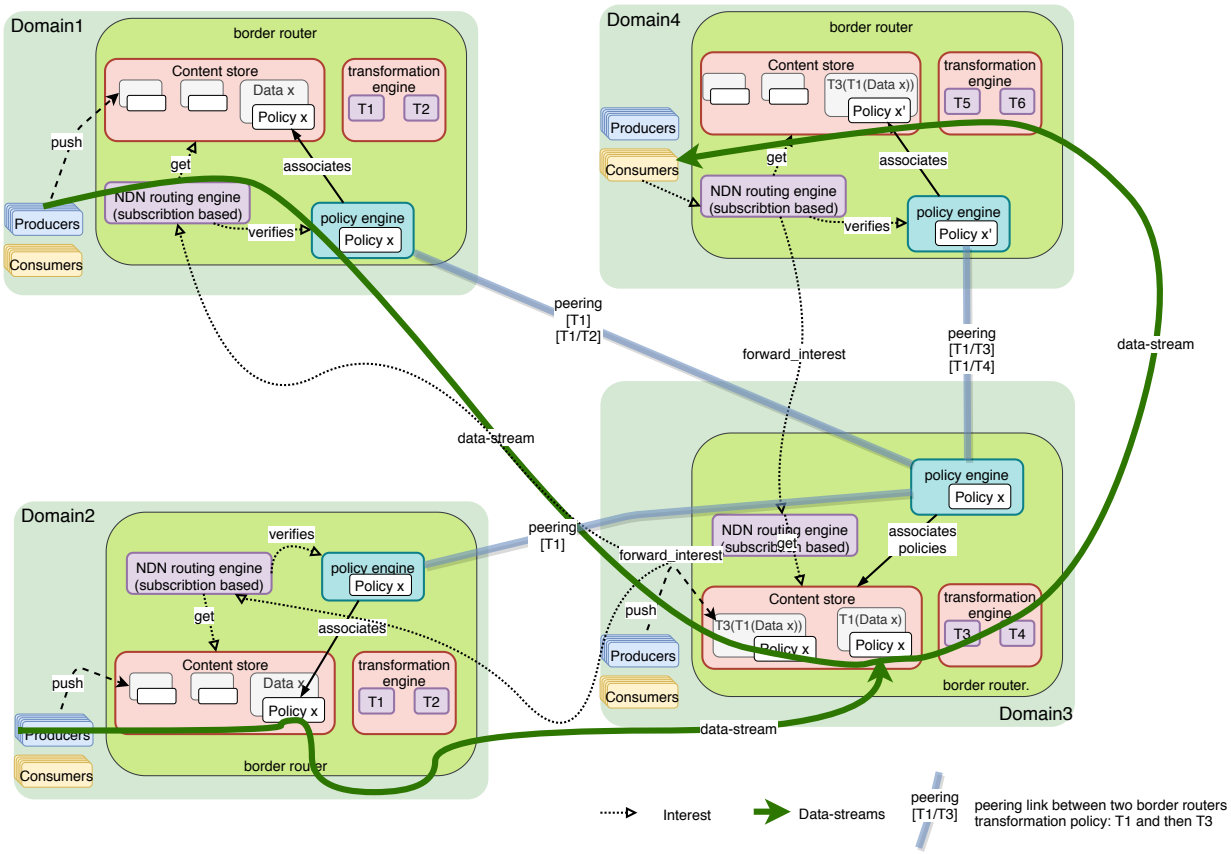

Fig. 2: An overall illustration of our multi-domain overlay solution with all its components.

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:

$$\begin{aligned} &\exists t \in \mathcal{F}_{src} \mid t \subseteq \mathcal{F}_{int} \wedge U_{int} \in U_{pol} \\ Pfn_{int} \notin BL_{pol} \wedge &\exists t \in \mathcal{F}_{stik} \mid t \subseteq \mathcal{F}_{int} \wedge U_{int} \in U_{pol} \end{aligned} \quad (1)$$

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.

D. 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, similarly to [20]. We let the exact definition of a global naming scheme of these transformations to future works.

E. 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:

$$\forall i \in [0, k_c], \mathcal{F}_{cached}(i) = \mathcal{F}_{int}(i) \quad (2)$$

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 as enforced in Eq. 2 but only if Eq. 1 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. 2. Domain3 is in charge of transforming the data between Domain4 and Domain1/Domain2. It collects the data, which has already been initially 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).

IV. Performance Evaluation

In order to evaluate the benefits of our solution described in Sec. III, we assess here its performance with simulations. We consider here 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 the number of measurements c_i and its value v_i. The transformation function tf takes as input a collection of samples $S_{i\in[1,k]}$ and returns:

$$S_{tf} = \left\langle \sum_{i\in[1,k]} c_i, \frac{\sum_{i\in[1,k]} v_i \cdot c_i}{\sum_{i\in[1,k]} c_i} \right\rangle \tag{3}$$

We may implement in a similar way transformations such as $Min()$, $Max()$, or more complex series transformations based on wavelets [21].

A. Simulation setup

We implement and compare the 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 III).

TABLE I: 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, $\in [2, 5]$

We extended the ndnSIM simulator (https://ndnsim.net/) to support all border router features and stream-based subscriptions. Our implementation is freely available[1].

We consider the following metrics:

Size of content stores measures the amount of data in the content store of each node. 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.

We proceed as following (Table I details the values of the parameters 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. Then, we add a fixed number of edges between random pairs of domains.
2) we construct the 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 I).
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

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

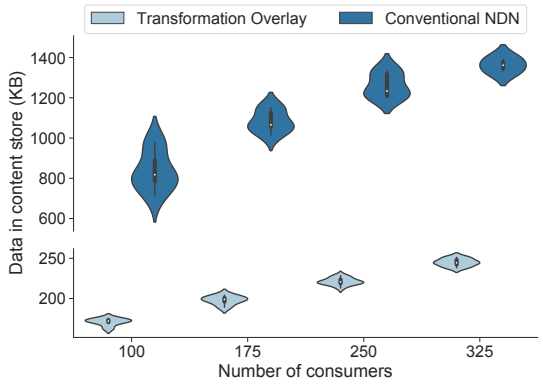

Fig. 3: Content store usage of Border routers: our transformation overlay does not require to collect and spread all the raw data.

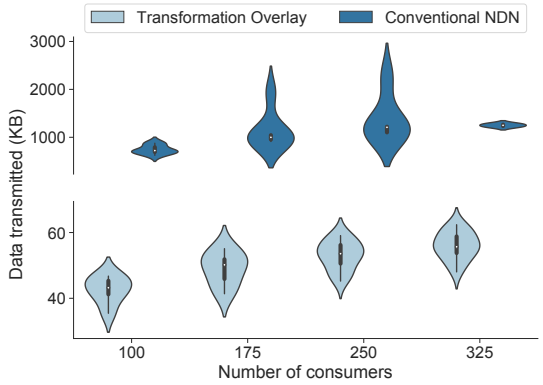

Fig. 4: Data transmissions on Border routers: our transformation overlay scales well regarding the network load for large number of consumers (and requests).

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 tends to exploit longer routers since physical routes must follow the links of the overlay. Paths in the conventional NDN approach are in general shorter.

B. Simulation results

We first measure the size of the content store for both solutions (Figure 3). 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.

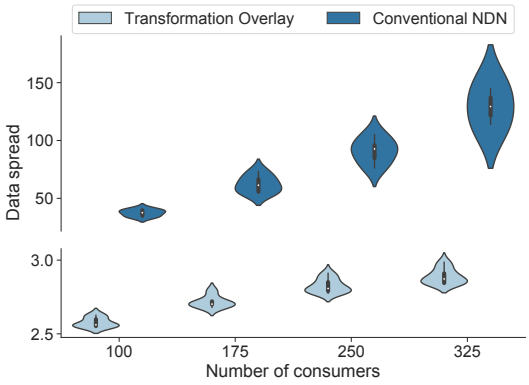

Fig. 5: Spread of content in the network

Then, we also measure the network load for all the routers (Fig. 4). 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 5 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.

Finally, figure 6 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 sub-optimal, 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,

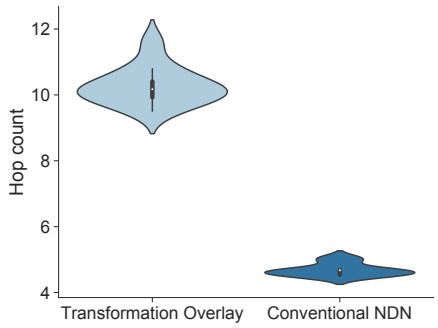

Fig. 6: Path size between consumers and producers

this increase may be much lower.

V. Conclusion and Future Works

In this paper, 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.

In the future, we plan to extend our routing scheme to take into account monetized data exchanges, with dynamic pricing strategies. Typically, a broker may advertise a lower price when several subscribers use the same dataset. Also, since the routing engine update existing subscriptions when novel interests arrive, many complex strategies become possible if overlaps with already handled interest are not sufficient. Such dynamic re-optimization and pricing models are challenging as subscriptions have to be constantly updated on-the-fly while still guaranteeing self-stabilization.

Acknowledgment

This work was supported by the French National Research Agency (ANR) project Nano-Net under contract ANR-18-CE25-0003.

References

[1] X. Yu and Y. Xue, "Smart grids: A cyber–physical systems perspective," *Proceedings of the IEEE*, vol. 104, no. 5, pp. 1058–1070, May 2016.

[2] J. Bakakeu, F. Schäfer, J. Bauer, M. Michl, and J. Franke, "Building cyber-physical systems – a smart building use case," in *Smart Cities: Foundations, Principles, and Applications*. John Wiley & Sons, Ltd, 2017, pp. 605–639.

[3] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, Oct 2016.

[4] L. Zhang *et al.*, "Named data networking," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 3, pp. 66–73, jul 2014.

[5] M. Sifalakis, B. Kohler, C. Scherb, and C. Tschudin, "An information centric network for computing the distribution of computations," in *Conference on Information-Centric Networking (ICN)*. ACM, 2014, pp. 137–146.

[6] R. C. J. Neto, P. Merindol, and F. Theoleyre, "A multi-domain framework to enable privacy for aggregated iot streams," in *Conference on Local Computer Networks (LCN)*. IEEE, 2020, pp. 1–4. [Online]. Available: https://icube-publis.unistra.fr/4-JMT20

[7] B. Li, D. Huang, Z. Wang, and Y. Zhu, "Attribute-based Access Control for ICN Naming Scheme," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 2, pp. 194–206, mar 2018.

[8] I. Wagner and D. Eckhoff, "Technical Privacy Metrics: A Systematic Survey," *ACM Computing Surveys*, vol. 51, no. 3, pp. 57:1–57:38, 2018.

[9] C. Dwork, "Differential privacy," *Encyclopedia of Cryptography and Security*, pp. 338–340, 2011.

[10] O. Ascigil, S. Reñé, G. Xylomenos, I. Psaras, and G. Pavlou, "A keyword-based ICN-IoT platform," in *Conference on Information-Centric Networking (ICN)*. ACM, 2017, pp. 22–28.

[11] J. Zhang, Q. Li, and E. M. Schooler, "iHEMS: An information-centric approach to secure home energy management," in *SmartGridComm*. IEEE, nov 2012, pp. 217–222.

[12] R. C. Sofia and P. M. Mendes, "An Overview on Push-Based Communication Models for Information-Centric Networking," *Future Internet*, vol. 11, no. 3, p. 74, mar 2019.

[13] H. W. Chen and F. J. Lin, "Converging mqtt resources in etsi standards based m2m platform," in *International Conference on Internet of Things (iThings)*, Sep. 2014, pp. 292–295.

[14] P. Colombo and E. 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.

[15] P. Desai, A. Sheth, and P. Anantharam, "Semantic Gateway as a Service Architecture for IoT Interoperability," in *International Conference on Mobile Services*, vol. 32, no. 2. IEEE, jun 2015, pp. 313–319.

[16] 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.

[17] 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.

[18] 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.

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

[20] M. Król and I. Psaras, "Nfaas: Named function as a service," in *Conference on Information-Centric Networking (ICN)*. ACM, 2017, pp. 134–144.

[21] S. Boubiche, D. E. Boubiche, A. Bilami, and H. Toral-Cruz, "Big data challenges and data aggregation strategies in wireless sensor networks," *IEEE Access*, vol. 6, pp. 20 558–20 571, 2018.