Towards end-to-end privacy for publish/subscribe architectures in the Internet of Things

Stevan Coroller, Sophie Chabridon, Maryline Laurent, Denis Conan, Jean Leneutre

Towards End-to-end Privacy for Publish/Subscribe Architectures in the Internet of Things

Stevan Coroller*, Sophie Chabridon*, Maryline Laurent*, Denis Conan*, Jean Leneutre†
*SAMOVAR, Télécom SudParis, CNRS, Université Paris-Saclay, France
†LTCI, Télécom ParisTech, Université Paris-Saclay, France

***Abstract*—The Internet of Things paradigm lacks end-to-end privacy solutions to consider its full adoption in real life scenarios in the near future. The recent enactment of the EU General Data Protection Regulation (GDPR) indeed emphasises the need for stronger security and privacy measures for personal data processing and free movement, including consent management and accountability by the data controller and processor. In this paper, we suggest an architecture to enforce end-to-end data usage control in Distributed Event-Based Systems (DEBS), from data producers to consumer services, taking into account some of the GDPR requirements concerning consent management and data processing transparency. Our architecture proposal is based on UCON$_{ABC}$ usage control models, which we overlap with a distributed hash table overlay for scalability and fault-tolerance concerns, and across and within systems data usage control. Our proposal highlights the benefits of combining both DEBS and end-user usage control architectures. To complete our approach, we quickly survey existing encryption models that ensure data confidentiality in topic-based Publish/Subscribe systems and highlight the remaining obstacles to transpose them to content-based DEBS with an overlay of brokers.**

***Index Terms*—Privacy, IoT, Usage Control, Content-based Distributed Event-Based Systems.**

I. Introduction

The Internet of Things (IoT) is an ever-growing topic in current research as it combines high stakes with unique constraints requiring specific solutions: how to distribute multiple context data flows between low performance heterogeneous devices and systems in constrained networks, with a generic, scalable and fault-tolerant infrastructure? As stated in [2], Distributed Event-Based Systems (DEBS), also called publish-subscribe systems, are a suitable solution to disseminate vast amounts of data in the IoT in a scalable way, especially when leveraging distributed infrastructures [6]. Furthermore, with recently increasing global expectations in terms of privacy, there is an urgent need for confidentiality-preserving solutions well tailored to the IoT. The survey [13] shows that confidentiality concerns in Pub/Sub architectures are addressed by two, commonly separated, lines of research: specific encryption models and security models based on access control. More precisely, the recent enactment of GDPR[1] raises new requirements in terms of confidentiality and data usage control, in particular: services must obtain the users' consent for a well-motivated purpose before collecting their data, allow users to revoke this consent at anytime, minimise data collection, and ensure transparency at all steps of data processing. This paper proposes a technical approach for addressing subparts of the GDPR requirements, in the very specific DEBS environment.

Topic-based and content-based filtering are two main filtering models of Pub/Sub architectures [6]. In topic-based filtering, event data are complemented with metadata tags called topics, and subscription filters correspond to regular expressions of topics that consumers are interested in. As a consequence, topics constitute a vocabulary or ontology that is shared by producers and consumers, and event data may remain opaque to routing. Content-based filters are conjunction of filters that parse different parts of the event data, and possibly some metadata that are added for instance to model the quality of event data. Consequently, the filtering is potentially more powerful, but more expensive, and the whole content of event data should be disclosed to routers. Since IoT systems are open systems, we chose content-based filtering for our study.

In this paper, we consider that access control alone is not sufficient to protect privacy and must be associated with usage control. We propose to integrate usage control, as defined by the UCON model [14], in Distributed Event-Based Systems with specific design choices for tackling the IOT: a broker-based infrastructure, content-based routing and Attribute-Based Access Control (ABAC) context-aware privacy policies.

The rest of this paper is organised as follows. In Section II, we specify the context of our work and we propose a motivating scenario that depicts what threat models are at stake. Then, in Section III, we detail our vision of the core part of a privacy-preserving content-based DEBS in order to provide usage control and obligations management in a broker-based dissemination architecture. In Section IV, we highlight recent works on topic-based encryption models that would complete our proposed architecture and explain the remaining issues to be addressed to transpose them to content-based DEBS. Finally, we discuss related works in Section V, and we conclude our paper and identify future works in Section VI.

II. Motivations and objectives

We present a motivating scenario in Section II-A with the description of the stakeholders in Section II-B and the corresponding threat models in Section II-C.

[1] https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex\%3A32016R0679

A. Motivating scenario

Alice is a bather on beaches watched by a lifeguard service. Bob is a lifeguard on duty on the same beach as Alice. The lifeguard service provides all bathers and lifeguards with wristbands equipped with geolocation sensors (*e.g.* RFID wristbands). The lifeguard service uses a data dissemination middleware to collect geolocations, along with personnal data like bathers age and name, for instance to increase children surveillance. The brokers of the dissemination middleware are deployed on all monitored beaches of the seafront. These brokers may be owned by third party organisations: e.g., local shops and snack bars, or public infrastructures such as bus stops. The rationale is that brokers are shared by multiple services as shown in Figure 1.

Bathers' location must be sent to local lifeguards—i.e. lifeguards on duty on the same beach. When registering to the service, bathers send their consent. Thereafter, local lifeguards can visualise on their smartphone bathers' location and will be alerted when some bathers are swimming too far from the shore. In this application, consent revocation is needed for several reasons: GDPR requires users to be able to revoke their consent at anytime; bathers may loose their geolocation device in the water, which could interfere with lifeguards' work; or bathers may leave the beach. Meanwhile, both bathers' and lifeguards' locations must be sent to the servers runnning the lifeguard service in order to adapt the number of lifeguards on each beach, for instance by shifting some lifeguards from one beach to another. This processing could take additional data into account: e.g. proportion of children, sea and swimming conditions, and the new assignments are broadcast to lifeguards using the same infrastructure. This scenario stresses out the need for content-based DEBS expressiveness: lifeguards need precise location figures to be able to quickly locate bathers in danger, but lifeguard servers only need to know on which beach bathers are located, and thus can accept low-precision location data. Such quality of context filtering is difficult to handle with topic-based filtering.

The first requirement that is demonstrated in this scenario is end-to-end data confidentiality—i.e. from data producers to consumer services, consent management (collection and withdrawing), along with data processing transparency on the consumer side, thus offering technical directions for addressing some of the GDPR principles. In addition, the approach must be generic so as to handle fine-grained consents, or to adapt to a wide range of application domains. The architecture must also scale and be fault-tolerant.

B. Scenario stakeholders

We identify the various stakeholders of our scenario:

1) Bathers: They produce sensitive data: e.g. their position on the beach.
2) Lifeguards: They subscribe to local bathers precise location and also publish their own low-precision location.
3) Lifeguard service: It subscribes to both bathers' and lifeguards' location data, and processes them on several servers for allocating lifeguards to beaches.

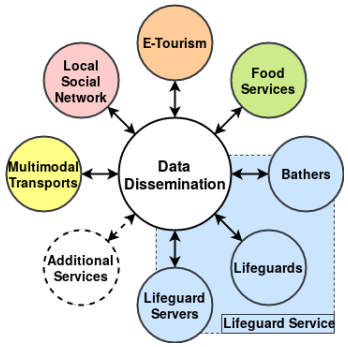

Fig. 1. Data dissemination and lifeguard service

4) Other services: There are various services like food services or local social networks that should be able to use the common infrastructure to disseminate context data. For instance, bathers may subscribe to a local social network to receive notifications about local events. Furthermore, bathers should be able to differentiate the precision of the location that they publish, and without any data leakage between the different services.

C. Threat models

In this section, we list the threats to data confidentiality, integrity and availability that are at stake in our scenario.

First of all, we make some assumptions. We consider that all communication channels between components (subscribers and publishers, commonly called clients, and brokers) are secured—i.e. authenticated and confidential. For this reason, we will further ignore several threat models like MITM attacks or rogue devices. We also consider that manufacturers are responsible for the security of their devices that results from implementation choices. Consequently, we will not consider the case of corrupted brokers or clients. The combination of these two assumptions ensures that all brokers and clients are authenticated.

In addition to the requirements identified in Section II-A, our work tackles the three following threat models that focus on privacy issues of relevance for our scenario:

1) Honest-but-curious brokers or semi-trusted brokers. It is a common assumption in security research. In our case, we consider that brokers strictly follow the DEBS routing algorithms, but may disclose any clear-text data they are provided with. Sensitive data do not only include publications content but also advertisement and subscription filters. The authors of [1] highlight more complex threat models where semi-trusted brokers can collude together or with subscribers to obtain unauthorised access to publications. These threat models are discussed in Section IV.
2) Semi-trusted subscribers. Authenticated subscribers may use loose subscription filters to receive numerous publications, even if they do not have the consent of data owners to receive such data. In our scenario, it

means that food service subscribers might abuse loose subscription filters to receive bathers geolocation.

3) Data misuse within consumers infrastructure. Once publications are sent to authorised subscribers, the infrastructure looses control over the data. Only legal obligations remain, like data owners consent, which do not prevent data consumers to process data as they will, and as long as their infrastructure is not completely inspected. For example, the lifeguard service could sell bathers geolocation to food services that may then focus their sellers to most busy beaches.

III. End-to-end usage control architecture for Pub/Sub networks

In this section, we detail our proposition: an end-to-end usage control architecture for broker-based DEBS. The overall architecture of the proposition is depicted in Figure 2. In Sections III-A and III-B, we introduce UCON$_{ABC}$ usage control model, and then we detail its adaptation to broker-based DEBS. In Section III-C, we describe the architecture of usage control on the consumer side. Concerning the last component, namely the usage log storage system, we do not further detail it in this paper because it constitutes an external and independent system.

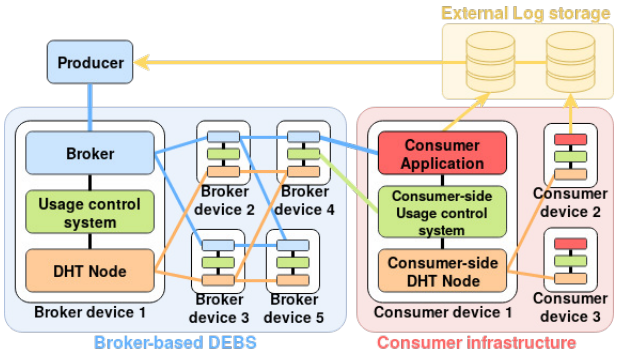

Fig. 2. DEBS with end-to-end usage control

A. UCON$_{ABC}$ usage control models

The UCON$_{ABC}$ family of reference models for usage control provides three decision factors: authorizations, obligations and conditions [14]. UCON$_{ABC}$ authorizations work in the same way as Attribute-Based Access Control to define users' rights over objects: users provide ABAC information that are matched against a policy to define their rights. Obligations allow to check if users have performed some determined actions before they can access objects. Conditions are external conditions bound to the execution environment, like current time or network load.

UCON$_{ABC}$ also provides two essential features: continuity of access decision and attribute mutability. Objects are protected by policies, and the previously-defined three decisions factors can be checked at three different instants: before users have access to the object (*pre*), during the access (*ongoing*), and after the access (*post*). When a user accesses an object, their ABAC information can be changed to modify their rights, as a consequence of the access.

These features and decision factors cover a wide range of use cases. In our scenario, pre-authorizations allow us to check that data consumers own producers' consent before receiving their data, and consequently avoid the "semi-trusted subscribers" threat model. Ongoing obligations and mutable attributes allow us to continuously check if consumers have sent back tamper-proof usage logs for previously received data, and thus to ensure data processing transparency.

B. UCON$_{ABC}$ adaptation to broker-based DEBS

In regular broker-based DEBS, as drawn in Figure 2, the clients, being producers or consumers, are connected to the system through a link to their access brokers. Internal or inner brokers, which do not have any local client connected to them, serve to route publications from the access broker of the producer to consumers via their access broker. The second role of brokers is to implement UCON$_{ABC}$ usage control. The usage control data are stored in a reliable and scalable manner into a distributed hash table (DHT) organised as an additional overlay on top of the brokers: the peers of the DHT are the brokers. Therefore, as drawn in Figure 3, brokers comprise three sub-components: the DEBS routing part for forwarding publications to interested consumers, the usage control system (UCS) that is embedded into the broker, and the DHT node that stores UCS data.

1) Brokers role in usage control: We choose XACML 3.0, which is a recognised OASIS standard commonly used to enforce ABAC in production environments, as the representation for usage control policies that are to be matched against ABAC information. The work presented in [4] and [5] provide XACML adaptations to handle all UCON$_{ABC}$ features. The XACML 3.0 documentation provides a reference architecture[2] to enforce access control. In Figure 3, we adapt this reference architecture.

We embed several components of the UCS directly into brokers. Hence, brokers play the role of Policy Enforcement Points (PEP). The usage control policies are specific to each producer because of being closely related to legal consents. This explains why we do not use any Policy Administration Point (PAP) to create generic policies, but producers send their own policies to their access broker, which stores them. Additionally, we define three different types of Policy Information Points (PIP), which are responsible for attribute queries: 1) the local PIP that fetches locally stored attributes, 2) the remote PIP that fetches attributes stored in other DHT nodes, and 3) the PIP responsible for fetching environment attributes in order to perform condition checks. Finally, we use a standard Policy Decision Point (PDP) that interacts with a Context Handler (CH) to fetch missing attributes from corresponding PIPs.

[2]http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html\#_Toc325047089

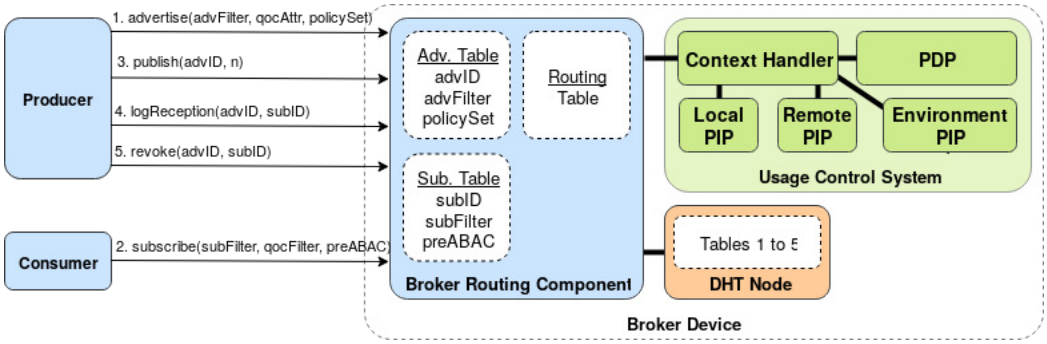

Fig. 3. Broker-based DEBS and usage control architecture

2) Data model for usage control: The data model of the usage control system comprises the following tables stored in the DHT:

T1: (advID, brokerID, policySet)
T2: (advID, subID, subAttr, valid)
T3: (advID, subID)
T4: (subID, advID)
T5: (advId, subID, interestedBrokers)

"adv", "sub", "ID" stand for "advertisement", "subscription", and "identifier", respectively. Underlined fields correspond to primary keys to build indexes. As described in [11], UCON_{ABC} can be specified in U-XACML, in the form of a tuple of five XACML policies including notably: pre-decision (U-PD), ongoing authorisations and conditions (U-OAC), and ongoing obligations and attribute updates (U-OBU). This tuple of XACML policies corresponds to the policySet field.

3) API for routing and usage control: We now detail the five types of messages that producers and consumers can send to their access broker, and the corresponding treatments. Producers declare the kind of data they will publish (call advertise). Then, they publish these data (call publish). Consumers declare the information they want to receive (call subscribe). As will be discussed in Section III-C, consumers' usages of the data are logged and they inform their access broker of new logs (call logReception). At anytime, after considering usage logs, producers can revoke their consents (call revoke).

a) Call advertise(advFilter, qocAttr, policySet): advId: Producers advertise before publishing so that they declare 1) which data they are willing to send (advFilter), 2) the quality of context[3] they guarantee on these data (in the form on attributes, qocAttr), and 3) their privacy requirements, in the form of policies (policySet). The caller receives a unique identifier (advId) and the broker adds a new T1 entry into the DHT.

b) Call subscribe(subFilter, qocFilter, preABAC): subId: Consumers subscribe to receive publications by providing 1) a filter to describe the information they are interested in (subFilter), 2) the quality of context they require on these data (qocFilter), and 3) the ABAC information that will be matched against U-PD policies (preABAC). A unique subscription identifier (subId) is returned and the system builds a covering tree in the overlay of brokers for routing publications to the consumer.

c) Matching advertisements and subscriptions: Anytime an advertisement is received, the access broker matches the new U-PD policy with the preABAC data locally stored. Anytime a subscription is received, every broker matches the new preABAC data with its locally stored U-PD policies. When a broker finds a matching, it stores T2, T3 and T4 entries into the DHT. Notably, subscriber attributes subAttr are initialised and the valid boolean is truthified into the T2 entry. This boolean value can be changed after any policy re-evaluation: if the result becomes DENY, then the boolean is falsified.

Since a matching can be cancelled (valid is falsified), a T5 entry is also updated so that interested brokers register to future notification of change to the fields subAttr or valid. In other words, we use the DHT to build a Pub/Sub system [8]. The difference between the overlay of the DEBS for routing and the overlay for the DHT is the following: the brokers are physically distributed (e.g. in smart cities) for data flow control of the routing function; the brokers form a DHT for storing usage control in sparse accesses.

d) Call publish(advID, n): When a producer publishes notification n, n is parsed to match advFilter (done at producer's access broker, namely B_{prod}), and next subFilter. Then, valid is checked to be true. The latter condition may be tested in three different ways according to message traffic and computation costs: 1) the check is performed by B_{prod} and all the brokers send n complemented by advId and the set {subID: valid}, 2) B_{prod} sends n with advId and all the brokers towards the consumer computes {subID: valid}, and 3) B_{prod} sends n with advId and the access broker of a subscriber computes valid.

In any case, publications are sent to consumers along with the related advID and subID so that usage logs created by consumers contain both advID and subID. Data owners can then identify the usage logs they receive from the log storage infrastructure. In addition, the U-OBU policy from each concerned policy set can be used to update subscribers' attributes everytime a publication is successfully routed. When attributes

[3]QoC and privacy are linked [3]

are updated, the U-OAC policies must be re-evaluated by the access broker of the publishers, and valid is potentially updated.

e) Call logReception(advID, subID): When a producer receives a new data usage log from the log storage infrastructure, this producer (or the storage infrastructure itself, by proxy), must send a logReception message to its access broker, which updates subAttr values in the corresponding T2 entry of the DHT. This update can be performed thanks to U-OBU's attribute update feature. The node responsible for updating policies for the (advId, subId) pair is notified of the updates via the DHT's internal Pub/Sub system. It then re-evaluates the U-OAC policy and potentially updates the valid attribute.

f) Call revoke(advID, subID): At anytime, a producer can revoke their consent. The workflow is similar to that of logReception. The access broker updates the revoke field of subAttr, the U-OBU policy is re-evaluated to imperatively decide DENY, and the valid attribute is falsified. This solution suffices to (eventually) stop consumers from receiving additional data from producers who revoked their consent. In order to go even further, an additional table could be created in the DHT to keep track of consumers' reputation that would be updated at revocation time by post-obligations and attribute update policies.

g) Calls unadvertise(advID) and unsubscribe(subID): Tables T3 and T4 of the DHT are present to manage these two calls.

h) Usage control in action: Let us return to the scenario of Section II. Alice publishes her geolocation and wants to receive a usage log of her sensitive data for every publication. The subAttr field contains a requiredLog field that is incremented by the U-OBU policy everytime a publication from Alice is published. The U-OBU policy checks whether requiredLog is strictly smaller than 1, and deny the access if not so. This field is decremented by the U-OBU policy at the reception of a logReception message. This constitutes a trivial use case. In real life scenarios, requiredLog would be replaced by timers, or a more advanced reputation system, for instance by using post-obligations and attribute update policies.

For now, our architecture handles usage control based on fine-grained users consent, which counteracts the second threat model presented in Section II-C, along with consent revocation. The use of obligations enables consumer-side processing transparency, but does not preventively counteract the third threat model. In addition, nothing ensures the authenticity of data usage logs. The next section links our DEBS usage control architecture to a consumer-side usage control solution, in order to tackle these two issues.

C. Consumer-side within and across systems usage control

In [7], the authors propose a within and across system usage control architecture which provides both preventive and detective usage control. "Within system usage control" is ensured by system calls monitoring: the usage control application includes a PEP based on the *strace* tool to intercept system calls that may affect monitored data, and a PDP that matches intercepted calls against data usage control policies. Unauthorised system calls can then be interrupted to perform preventive usage control, or they can be released but logged to enable detective usage control. This solution ensures that any copy of monitored data remains monitored, whatever its representation. "Across system usage control" is ensured by intercepting socket-related system calls and sending data usage control policies to remote machines before sending them monitored data. This requires that all machines that want to receive monitored data previously install the usage control application.

The only drawbackof this implementation is the overhead induced in computation, which makes it not suitable for monitoring real-time consumer applications, among others. The authors of [7] claim that "it depends on use case scenarios whether the imposed overheads are acceptable in practice". For this reason, in [16], the authors show that system calls should not be used for monitoring purposes, and provide an alternative low-intrusive solution, which is adapted to real-time systems requirements in terms of latencies. This alternative is to be used instead of [7]'s implementation in case system call monitoring is effectively proved not to be suited for applications which require low latencies.

The rationale for using the work of [7] in conjunction with our DEBS usage control architecture is that it tackles the third threat model "data misuse within consumers infrastructure", while ensuring that data usage logs are tamper-proof, as they can themselves be monitored by the usage control application.

By adding consumer-side within and across systems usage control, we obtain an end-to-end usage control architecture that fills the requirements stated in Section II. Its complete decentralisation using an overlay of brokers managing a DHT additionally provides both scalability and fault tolerance. In the first threat model (see Section II-C), third-party-owned brokers can only be considered as semi-trusted. Yet, brokers can realistically be considered as honest-be-curious, or able to collude with semi-trusted consumers, as explained by [1]. This last obstacle towards end-to-end privacy in the IoT is discussed in the next section.

IV. Semi-trusted third-party-owned brokers

As stated by [13], threat models related to the semi-trusted brokers assumption in Pub/Sub is tackled by research on encryption models and encrypted matching. Yet, some obstacles remain in this field, like key management and distribution. In addition, [1] argues that most of the recent works in this field still tackle simple threat models, ignoring the possibility of collusions between brokers and consumers.

The context of distributed content-based Pub/Sub implies additional constraints in the choice of a suitable encryption model to complete our end-to-end privacy-preserving architecture: most of the existing works rely on trusted authorities, which would introduce a single centralised element in our

fully decentralised architecture. Additionally, works based on *Attribute-Based Encryption* (ABE) rely on the fact that an exhaustive list of topics is globally known: some secrets are associated to each topic, and publications are encrypted with the secret of the topic they belong to. Thus, ABE is hardly transposable to content-based Pub/Sub where there is an infinite number of possible subscription filters.

The solution in [17] does not rely on ABE, but it tackles the trusted authority issue, and even addresses the key management issue by performing local key management on each broker. However, it still fails to efficiently render content-based high expressiveness.

V. Related Works

The authors of [10] provide a UCON implementation using the Cassandra DHT into P2P networks in the IoT. Just like our achitecture, they use a DHT to distribute UCON policies and ABAC information storage. However, this solution aims at providing smart devices collaboration and does not tackle privacy and confidentiality issues.

In [9], UCON is implemented into MQTT, a topic-based pub/sub protocol. This implementation being fully centralised, it lacks both scalability and fault tolerance. Yet, this implementation consists of a single application, completely separated from the broker one, and thus completely generic to the MQTT implementation of brokers and clients. This is not the case of our architecture, especially because we chose to embed several UCS components within the broker application. On the other hand, this implementation only handles access revocation in case consumers do not respect their obligations. It does not allow producers to revoke consumers' access at anytime, which was one of our first requirements.

The work presented in [15] aims at providing technical solutions to handle usage control based on users' consent, while also taking into account consent revocation issues. The approach is to piggyback usage control policies onto every piece of data, so that data is always transferred along with its usage control policy. Usage control is then enforced in a decentralised manner, on every machine that owns a copy of the monitored data. Yet, this solution heavily relies on encryption techniques, and thus requires a centralised trusted authority. This solution is closely related to [7], as it focuses rather on digital rights management issues.

Another line of research extends Information Flow Control (IFC) [12]. Data within messages are tagged with security labels. These labels are then checked by processes to ensure that data can only flow to processes with compatible labels. The approach proposed in [18] revisits IFC to build a middleware-based solution for policy-enforcement in order to enable a legally-compliant IoT. IFC and UCON share similarities and are promising solutions for the IoT. Although an in-depth comparison of these two approaches is still missing in the literature, we consider that the mutability of attributes in UCON is well suited to the dynamicity of context data in the IoT.

VI. Conclusion and future work

In this paper, we identified several requirements in terms of confidentiality and several threats to privacy that need to be addressed to ensure data privacy in the IoT. In the more restricted context of distributed content-based Pub/Sub, we presented a fully decentralised usage control architecture, to be used in conjunction with an already existing solution for within and across systems monitoring. Our proposal enables end-to-end data usage control, the management of the consent of the users with potential consent revocation at anytime, and ensures data processing transparency. It highlights the possibilities enabled by combining middleware security models with end-user data usage control, and the benefits of overlapping broker and DHT overlay networks within the same architecture, to ensure fault-tolerance and scalability.

Yet, our usage control architecture only provides complete end-to-end data confidentiality if we make the hypothesis that all brokers are fully trusted. In a real case scenario like the one we described in Section II-A, this assumption is not realistic, but no encryption model tackles the issue of semi-trusted brokers in content-based DEBS. This remains the only obstacle towards end-to-end privacy-preserving architectures in our context.

As part of our future work, we want to provide an implementation of our Pub/Sub usage control architecture, carry out performance tests to compare it with existing implementations of UCON$_{ABC}$ into Pub/Sub environments, and further discuss how obligations should be used in our architecture to adapt to the widest range of scenarios. Finally, we want to carry out further research to determine if security models are adapted to tackle the issue of data confidentiality under the semi-trusted brokers assumption.

References

[1] S. Belguith, S. Cui, M.R. Asghar, and G. Russello. Secure Publish and Subscribe Systems with Efficient Revocation. In *33rd ACM Symposium on Applied Computing*, pages 388–394. ACM, 2018.

[2] P. Bellavista, A. Corradi, M. Fanelli, and L. Foschini. A Survey of Context Data Distribution for Mobile Ubiquitous Systems. *ACM Computing Surveys*, 44(4):24:1–24:45, August 2012.

[3] S. Chabridon, R. Laborde, T. Desprats, A. Oglaza, P. Marie, and S. Machara Marquez. A Survey on Addressing Privacy together with Quality of Context for Context Management in the Internet of Things. *Annals of Telecommunications, Springer*, 69(1):47–62, February 2014.

[4] M. Colombo, A. Lazouski, F. Martinelli, and P. Mori. A Proposal on Enhancing XACML with Continuous Usage Control Features. In *Grids, P2P and Services Computing*, pages 133–146. Springer, 2010.

[5] E-Ghazia, U. and Masood, R. and Shibli, M. and Bilal, M. Usage Control Model Specification in XACML Policy Language. In *11th Conf. on Computer Information Systems and Industrial Management*, volume LNCS-7564. Springer, September 2012.

[6] P.T. Eugster, P.A. Felber, R. Guerraoui, and A.-M. Kermarrec. The Many Faces of Publish/Subscribe. *ACM Computing Surveys*, 35(2), June 2003.

[7] F. Kelbert and A. Pretschner. Data Usage Control for Distributed Systems. *ACM Trans. Priv. Secur.*, 21(3):12:1–12:32, April 2018.

[8] A.-M. Kermarrec and P. Triantafillou. XL Peer-to-peer Pub/Sub Systems. *ACM Computing Surveys*, 46(2):16:1–16:45, November 2013.

[9] A. La Marra, F. Martinelli, P. Mori, A. Rizos, and A. Saracino. Improving MQTT by Inclusion of Usage Control. In *Int. Conf. on Security, Privacy, and Anonymity in Computation, Communication, and Storage*, pages 545–560, 2017.

[10] A. La Marra, F. Martinelli, P. Mori, and A. Saracino. Implementing Usage Control in Internet of Things: A Smart Home Use Case. *IEEE Trustcom/BigDataSE/ICESS*, pages 1056 – 1063, 2017.

[11] A. Lazouski, F. Martinelli, and P. Mori. A Prototype for Enforcing Usage Control Policies Based on XACML. In *Trust, Privacy and Security in Digital Business*, pages 79–92. Springer, 2012.

[12] A. C. Myers and B. Liskov. A Decentralized Model for Information Flow Control. In *Proc. of the Sixteenth ACM Symp. on Operating Systems Principles*, pages 129–142, 1997.

[13] E. Onica, P. Felber, H. Mercier, and E. Rivière. Confidentiality-Preserving Publish/Subscribe: A Survey. *ACM Computing Surveys*, 49(2):27:1–27:43, June 2016.

[14] J. Park and R. Sandhu. The UCON ABC Usage Control Model. *ACM Trans. Inf. Syst. Secur.*, 7(1):128–174, February 2004.

[15] S. Pearson and M. Casassa-Mont. Sticky Policies: An Approach for Managing Privacy across Multiple Parties. *Computer*, 44(9):60–68, Sept 2011.

[16] M. Pohlack, B. Döbel, and A. Lackorzyński. Towards Runtime Monitoring in Real-Time Systems. In *Proceedings of the Eighth Real-Time Linux Workshop*, 2006.

[17] A. Shikfa, M. Önen, and R. Molva. Privacy-preserving content-based publish/subscribe networks. In *IFIP SEC 2009, 24th Int. Information Security Conference*, Pafos, CYPRUS, 05 2009.

[18] J. Singh, T. Pasquier, J. Bacon, J. Powles, R. Diaconu, and D. Eyers. Big Ideas Paper: Policy-driven Middleware for a Legally-compliant Internet of Things. In *Proceedings of the 17th International Middleware Conference*, pages 13:1–13:15, 2016.