IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 4, 2019, accepted January 31, 2019, date of publication February 12, 2019, date of current version March 8, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2899076

A Comprehensive Security Framework for Publish/Subscribe-Based IoT Services Communication

LI DUAN[1], CHANG-AI SUN[1], YANG ZHANG[2], WEI NI[3], AND JUNLIANG CHEN[2]

[1]School of Computer and Communication Engineering, University of Science and Technology Beijing, Beijing 100083, China
[2]State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China
[3]Data61, CSIRO, Sydney, NSW 2122, Australia

Corresponding author: Li Duan (duanli268@126.com)

This work is supported by the Project funded by China Postdoctoral Science Foundation under Grant 2017M620617, and in part by the National Natural Science Foundation of China under Grant 61872039.

ABSTRACT The publish/subscribe paradigm provides a loosely-coupled and scalable communication model for the large-scale IoT service systems, such as supervisory control and data acquisition (SCADA). Data confidentiality and service privacy are two crucial security issues for the publish/subscribe model deployed in different domains. The existing access control schemes for such model cannot address both the issues at the same time. In this paper, we propose a comprehensive access control framework (CACF) to bridge this gap. The design principle of the proposed framework is twofold: (a) a bi-directional policy matching scheme for protecting the privacy of IoT services; and (b) a fully homomorphic encryption scheme for encrypting published events to protect data confidentiality. We analyze the correctness and security of the CACF, moreover, we prototype CACF based on Apache ActiveMQ, an open source message broker, and evaluate its performance. The experimental results indicate that our security system meets the latency requirements for very high-quality SCADA services.

INDEX TERMS Access control, publish-subscribe, Internet of Things, data privacy.

I. INTRODUCTION

The communication systems in the existing Internet of Things (IoT) applications [1], such as smart girds, typically follow a request/reply interaction model, in which access to data is directly controlled by data providers [2]. This tightly-controlled model may not be scalable for large-scale IoT systems. A publish/subscribe model [3] features loose decoupling among event publishers and event subscribers, and can be more appropriate for IoT services communication infrastructure and enable flexible and dynamic collaborations among IoT services [4], [5].

In the context of open and wide-area IoT systems, various IoT services from different domains have to interplay for their mutual interests [6]. For example, supervisory control and data acquisition (SCADA) systems [7], which act as the core of the power grid, can interact with other services, as shown in Fig.1. In SCADA, collaborated services mainly include: acquisition & estimation (AE) service (S1), event analysis (EA) Service (S2), human machine interface (HMI) service (S3), control agent (CA) service (S4), warning service (S5), and the resource model (RM) service (S6). Multiple services are involved in maintaining the distributed control of SCADA. Each service S_i ($i = 1, \ldots, 6$) is connected through a notification broker NB_i ($i = 1, \ldots, 6$) into the broker network for publishing and subscribing events. A subscribing service describes its event request by ''event type'' or ''event topic'', the NB assumes the final delivery of events produced by a publishing service. The interactions among services depend on the access control policies and the attributes of participating services. In this case, the publisher and the subscriber can be anonymous, multicasting, and indirect.

In this scenario, there are two outstanding security flaws to be resolved [8]–[11]. One is that the published event data may be eavesdropped on by attackers through the pervasive IoT service interactions. The other is that the privacy information of collaborated services may be disclosed during service collaborations. Thus, it is desirable to design an access control scheme to tackle the security and privacy issues related

The associate editor coordinating the review of this manuscript and approving it for publication was Kim-Kwang Raymond Choo.

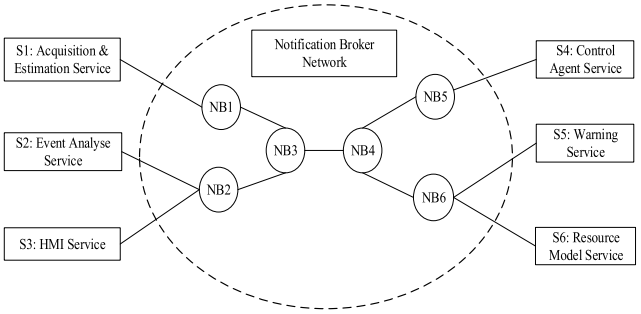


FIGURE 1. Pub/Sub-based IoT services communication infrastructure.

to IoT service interactions from two aspects [12]: (1) from a data perspective, it is necessary to protect the confidentiality of published data to ensure that the broker only delivers the data to qualified subscribers who are interested; (2) from an application perspective, it is necessary to protect the sensitive information of IoT services during collaborations.

Attribute-based access control is one of the most popular access control models [13], which enables publishers to define access policies to assess whether subscribers are qualified, and subscribers to specify their subscription policies to decide which events are interesting to them. It is difficult to tackle the above security challenges, especially in anonymous, multicast, and indirect service interactions. Following are two reasons: (1) the subscribers do not directly receive events from the publishers, and the publishers cannot directly reject the subscriber's requests on their events; and (2) one event is subscribed by multiple subscribers and different subscribers possibly receive the events from different NBs. Thus, in terms of access to services and their interacting events, it is impossible to execute surveillance for requests by using an omniscient reference monitor as done in the traditional method. Furthermore, most existing access control schemes may not be suitable for such large-scale IoT services collaborations. For example, in a conversation-based access control model [14], a service checks whether a client has proper credentials before the service starts a new conversation with the client. This indicates that the access control scheme requires direct interaction of the service and the client. In the access control scheme of DDS security standard [15], the publisher must share a symmetric encryption key with all subscribers. If there are too many subscribers, the use of symmetric keys for enforcing access control can prevent scalability to large-scale IoT service collaborations. In addition, the privacy of subscribers is not protected, since a publisher knows who subscribe to its service.

To bridge the gaps of existing access control schemes for publish/subscribe-based IoT services, we propose a comprehensive access control framework (CACF). In CACF, policies and attributes are first embedded in data and services [16] to preserve the multicast property. Bi-directional policy matching and policy enforcement are crucial to achieving our security goals. Bi-directional Policy matching means that the NB needs to check whether the associated attributes of the published data satisfy the subscription policies specified by subscribers. Meanwhile, the NB needs to check whether the attributes of the subscriber satisfy the access policy associated with the published data. The direct matching can result in privacy information leakage. Thus an attribute encoding approach is used in CACF. Policy enforcement is based on fully homomorphic encryption (FHE) scheme [17]. In previous works, security approaches [8], [9], [18] have focused on the confidentiality of data. Privacy-protection approaches [19]–[21] have focused on preserving service privacy. To the best of our knowledge, few existing approaches can holistically protect data confidentiality and services privacy at the same time.

In this work, our proposed CACF not only supports data confidentiality protection and homomorphic operations on encrypted data, but also access control for IoT services. Moreover, the publisher and the subscriber in our framework do not need to share any key, reducing key management burdens of publishers. We choose Apache ActiveMQ as JMS broker [22] and extend it to perform policy evaluation. The main contributions of this paper are as follows:

- We propose a comprehensive access control framework for collaborative IoT services. The framework allows the publishers and subscribers to control the event data by the bi-directional policy matching between protection requirements and services' capabilities.
- Bi-directional policy matching and policy enforcement are designed to embrace the proposed framework. As the first component, attribute encoding and anonymous-set-based principle are adopted to provide service privacy. As the second component, a fully homomorphic encryption scheme is adopted to protect data confidentiality.
- We analyze the correctness and security of our framework by proving that qualified subscribers can access encrypted events, while unqualified subscribers cannot access the events even if they are allowed to collude. We implement a prototype of the framework and evaluate its performance from the perspective of data delivery latency for different sizes of event data, different numbers of policies, and different numbers of attributes on the subscribers.

The remainder of the paper is organized as follows. Section II describes the FHE scheme used in our framework and overviews the algebraic structure of authorization policies. In Section III, we present the detailed construction process of our CACF. We present correctness and security analysis of our solution in Section IV and performance evaluation in Section V. Section VI review the related work. Section VII concludes this paper and points out future research.

II. PRELIMINARIES

In this section, we describe the fully homomorphic encryption (FHE) scheme, defined in [17], and attribute-based access control model to be used in the proposed framework.

A. FULLY HOMOMORPHIC ENCRYPTION

In our CACF, events are encrypted with an FHE scheme which supports homomorphic operations on encrypted events. The key switching operation provides an efficient way of re-encrypting events for particular subscribers.

Let q be a prime, Z_q an integer domain modulo q, and n an integer. The FHE scheme Π_{FHE} we use is a symmetric encryption scheme, Π_{FHE} contains the following operations:

FHE.SecKey(q, n): It takes the parameter q and n as the input to generate a secret key K.

FHE.PubKey(K): It takes a secret key K as the input to generate a public key PK.

FHE.Enc(K, v): To encrypt a, it outputs an n-dimensional vector ciphertext $C = (c_1, \ldots, c_n)$ with $c_i \in Z_q$, i.e., $Enc(K, v) = (c_1, \ldots, c_n)$.

FHE.Dec(K, C): It outputs v from the ciphertext $C = (c_1, \ldots, c_n)$ under the key K, i.e., $Dec(K, C) = v$.

FHE.Add(K, C_1, C_2): It takes two ciphertexts $C_1 = (c_{11}, \ldots, c_{1n})$, and $C_2 = (c_{21}, \ldots, c_{2n})$ encrypted under the same key K. If $Dec(K, C_1) = v_1$ and $Dec(K, C_2) = v_2$, then $Dec(K, C_1 \oplus C_2) = (v_1 + v_2) mod\ q$ and $Dec(K, d \odot C_1) = d * v_1\ mod\ q$, where $\oplus$ denotes the vector addition, $\odot$ denotes the scalar multiplication of a vector, and $d \in Z_q$.

FHE.Multi(K, v_1, v_2): It takes two plaintexts $v_1 \in Z_q$, and $v_2 \in Z_q$, the ciphertext $Enc(K, v_1 * v_2)$ is obtained by the expression $((c_{11} * c'_{11}) \odot pek_{11}) \oplus \cdots \oplus ((c_{ki} * c_{kj}) \odot pek_{ij}) \oplus \cdots \oplus ((c_{nn} * c_{nn}) \odot pek_{nn})$, where the n-dimensional vector pek_{ij} $(1 \le i \le n, 1 \le j \le n)$ is a public evaluation key derived from K.

Given a secret key K_p of a publisher and a public key PK_s of a subscriber, Π_{FHE} also contains the generating algorithm of a switching key, which is used to convert a ciphertext encrypted with K_p into a ciphertext under K_s, which is the secret key of the subscriber. The work in [23] gives the steps of generating the switching key. In the following, we describe the property of the switching key and re-encryption.

Let $\texttt{KeySwitch}(PK_s, K_p) = SW$ be the operation of generating the switching key $SW = \{SW_1, \ldots, SW_n\}$, and each SW_i is a n-dimensional vector. Suppose $Dec(K_p, (c_1, \ldots, c_n)) = v$. Then, the re-encryption of $C = (c_1, \ldots, c_n)$ with SW, denoted by $ReEnc(SW, C)$, is defined by the expression $(c_1 \odot SW_1) \oplus \cdots \oplus (c_n \odot SW_n)$.

Let $C' = ReEnc(SW, C)$. Then, we have $Dec(K_s, C') = v$. The correctness of the key switching is analyzed in [23]. Note that our framework can also use other FHE schemes, such as the scheme proposed in [24] and [25].

B. ATTRIBUTE-BASED AUTHORIZATION POLICY

In this paper, we adopt the attribute-based access control model. An attribute w can be written as an attribute-value pair (nm, val), i.e., the attribute nm has the value val. The attribute conjunction can be expressed as $w_1 \wedge \cdots \wedge w_m$, i.e., a subscriber has the attributes w_1, …, w_m, where "$\wedge$" is a logical "AND" operator, and "$\vee$" is a logical "OR" operator.

Given an event topic tp, a policy for this topic uses attributes to specify the subscribers which are qualified to access the data of topic tp.

Definition 1: An authorization policy is represented as $\Gamma_{tp} = (w_{11} \wedge \cdots \wedge w_{1m_1}) \vee \cdots \vee (w_{n1} \wedge \cdots \wedge w_{nm})$, meaning that a subscriber can access the data of topic tp if it has at least one set of the conjunctive attributes from $w_{11} \wedge \cdots \wedge w_{1m}$ to $w_{n1} \wedge \cdots \wedge w_{nm}$.

The attributes of a subscriber are represented as $\gamma = (w'_{11} \wedge \cdots \wedge w'_{1m}) \vee \cdots \vee (w'_{k1} \wedge \cdots \wedge w'_{km})$, meaning that the subscriber has k different sets of conjunctive attributes. If at least one set of conjunctive attributes in γ also appears in Γ_{tp}, then we say γ satisfies Γ_{tp}. We assume the attributes of a subscriber are linked to its public key and can be known to the publishers and brokers.

A Bloom Filter is a simple, space-efficient, and randomized data structure for representing a set of strings compactly for efficient membership query [9]. In our framework, an access control policy is encoded into bloom filters and sent as part of access credentials when publishing an encrypted event. For Γ_{tp}, we will generate n bloom filters for each set of the conjunctive attributes. The operation $\texttt{genBF}(w_{i1} \wedge \cdots \wedge w_{im})$ generates the bloom filter that can check the membership for m attributes $w_{i1}, \ldots, w_{im}$.

III. THE CONSTRUCTION OF CACF

In this section, we first present an overview of the CACF architecture, and then present the detailed matching procedure and access control procedure in CACF.

A. CACF ARCHITECTURE

Our CACF architecture has three layers, as illustrated in Fig. 2. The upper layer and the bottom layer assume the privacy protection function of application services; the two layers correspond to bi-directional policy matching operations between the protection requirements and capabilities of services. The middle layer assumes the confidentiality protection function of published events, corresponding to policy enforcement operation. These operations are performed by different collaborating services, as shown in Fig. 3. We describe these services as follows.

1) PUBLISHING SERVICE

A publishing service consists of three important modules: *authorization centre, policy translator*, and *encryption*. It firstly publishes an event announcement on a notification broker for an event type and attached policies, which provides event access to subscribing services through the broker. The *authorization centre* is in charge of managing attributes and access control policies in CACF. Before publishing the event on the broker, two steps are performed. The *policy translator* module firstly generates and assigns access credentials for each valid subscriber, the generation of access credentials are based on attributes of authorized subscribers and access control policies of protecting published event. Then the

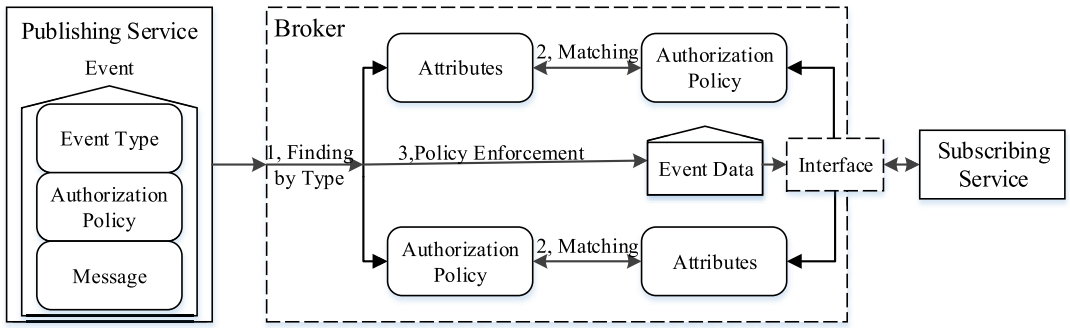


FIGURE 2. Comprehensive access control framework(CACF) for IoT service collaboration.

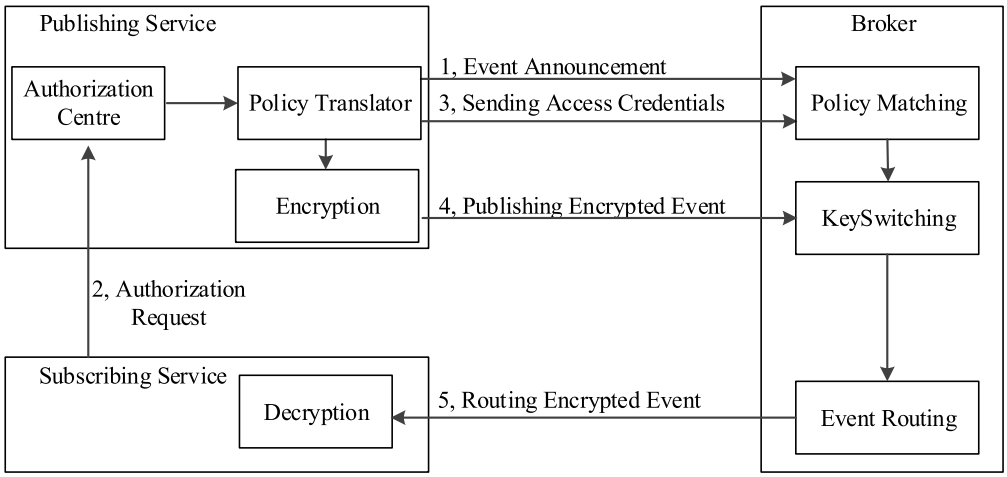


FIGURE 3. Data dissemination and authorization procedure in CACF.

encryption module encrypts the published event under its specified access control policies.

2) BROKER

The broker has three modules: *policy matching, keyswitching*, and *event routing*. It is responsible for storing the published event and providing event access services to subscribers based on their requested event types. In particular, according to a service message and the authorization request, the broker first finds the access control policy through an event type. If the type of the subscribed event is part of the received event, the policy matching module takes charge of checking whether the subscriber's attributes satisfy policies of the newly published event. If the matching results are not empty, then the subscriber is a valid consumer. For the valid subscriber, the *policy matching* module also evaluates whether attributes of the published event can satisfy the subscription policies specified by subscribers. If the results are true, the *keyswitching* module re-encrypts the encrypted event before routing the published event through the *event routing* module.

3) SUBSCRIBING SERVICE

Before sending its authorization request by an event type, each subscribing service specifies the subscription policies for protecting its own sensitive information, such that the subscribing service only receives the event whose attributes meet the subscription policies of the service. The subscribing service has a secret key which is used to re-decrypt and decrypt the routed event by the decryption module.

In CACF, an authorization procedure and a data dissemination procedure are illustrated in Fig. 3. The detailed steps are as follows: (1) a publisher publishes the event prefix announcement; (2) a subscriber expresses its authorization request by an event type; (3) according to the authorization request, the publisher generates and sends access credentials to the broker; (4) the publisher publishes encrypted events of the subscribed type. After receiving encrypted events, the broker first decides whether the subscriber is an authorized subscriber. For the valid subscriber, its broker switches the encrypted event under the publisher's secret key into the event ciphertext under the subscriber's public key. (5) According to the *event routing* module of the notification broker, the event is forwarded to the subscribing service. Later, the subscribing service can recover the event through a *decryption* module.

In our work, there are two security hypotheses. First, the public/secret keys and public parameters that are used in the CACF are assigned by the certificate authority. Second,

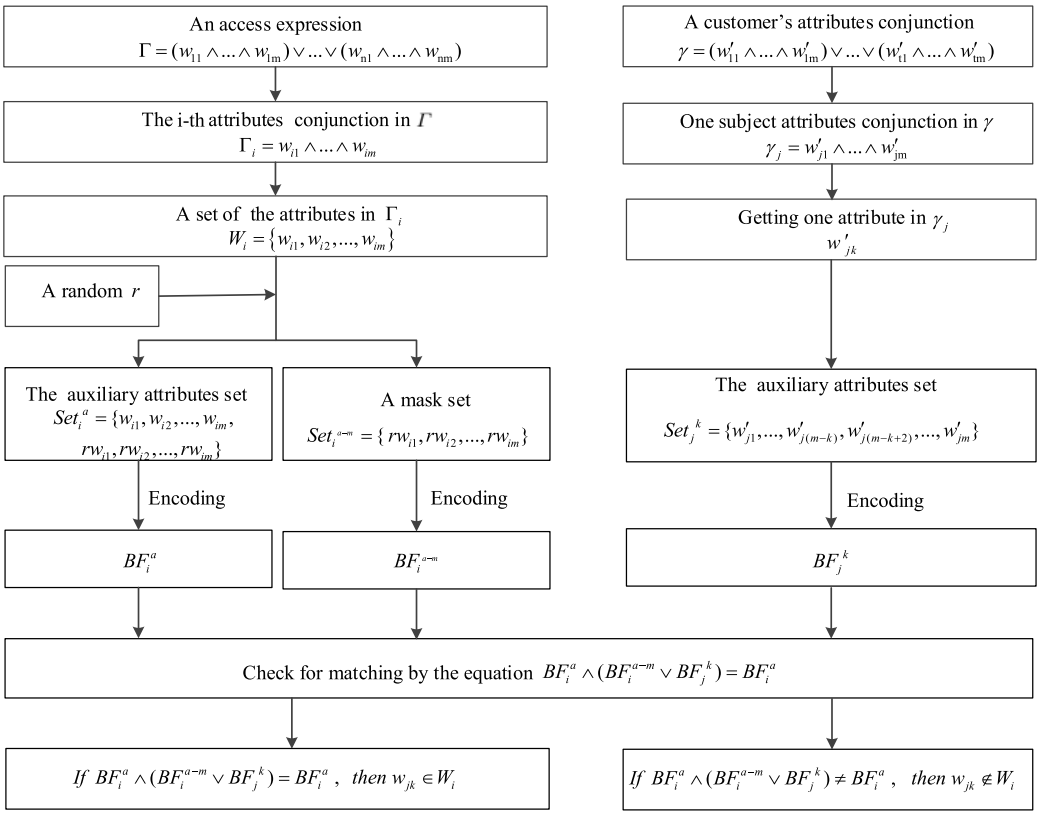


FIGURE 4. Policy matching procedure.

the notification brokers are honest-but-curious. They follow the protocol to check bi-directional policy matching and transform encrypted event, but they can also be curious about the published event and subscribers' interest. Third, publishing services are honest. However, in practice, subscribing services can be dishonest. They may attempt to access unauthorized events by colluding with each other or with the broker.

Among collaborative IoT services, there is a many-to-many relationship among multiple publishers and subscribers. In this section, we take one publisher and one subscriber to describe the access control procedure in our framework. Here we assume that a publisher P publishes an event with a type tp. Its access expression is denoted by a propositional formula Γ, where $\Gamma = (w_{11} \wedge \cdots \wedge w_{1m}) \vee \cdots \vee (w_{n1} \wedge \cdots \wedge w_{nm})$. Let a subscriber S have an attribute conjunction $\gamma = (w'_{11} \wedge \cdots \wedge w'_{1m}) \vee \cdots \vee (w'_{t1} \wedge \cdots \wedge w'_{tm})$. The process of finding the event type is obvious. In order to realize functions of each part, authorization policies and attributes are encoded to support rapid matching. Then a method of masking the attributes and policies is used to preserve service privacy. The attribute-based encryption scheme is last applied to support the scalable access control framework. In the following sections, we will present the matching technology and methods, as well as policy enforcement procedure in detail.

B. MATCHING PROCEDURE

For one event type, its authorization policy includes multiple attribute conjunctions, and different attribute conjunctions may have a different length. For different subscribers, they may have different attribute conjunctions, and the lengths of attribute conjunctions may be different. In our work, we assume the length of each attribute conjunction in an authorization policy is the same as the length of each subscriber's attribute conjunction. The core idea of checking the bi-directional policy matching is to check whether the attributes of subscribers are included in that of the authorization policy, as illustrated in Fig. 4. Here, we present the basic bi-directional policy matching ideal and matching procedure, correctness proof and security analysis of policy matching scheme have been discussed in [19].

Directly matching between attributes and policies would lead to the leakage of some critical information. Before realizing the first-layer matching, we encode the attributes of subscribers, and encode the attributes in the access policy conjunctions. In order to keep clear attributes unknown to adversaries, these attributes are encoded into anonymous attribute sets. Policy encoding is to encode a set of attributes which are included in attribute conjunction, into a hash value using a one-way set hash. A Bloom Filter (BF) is used to implement this function.

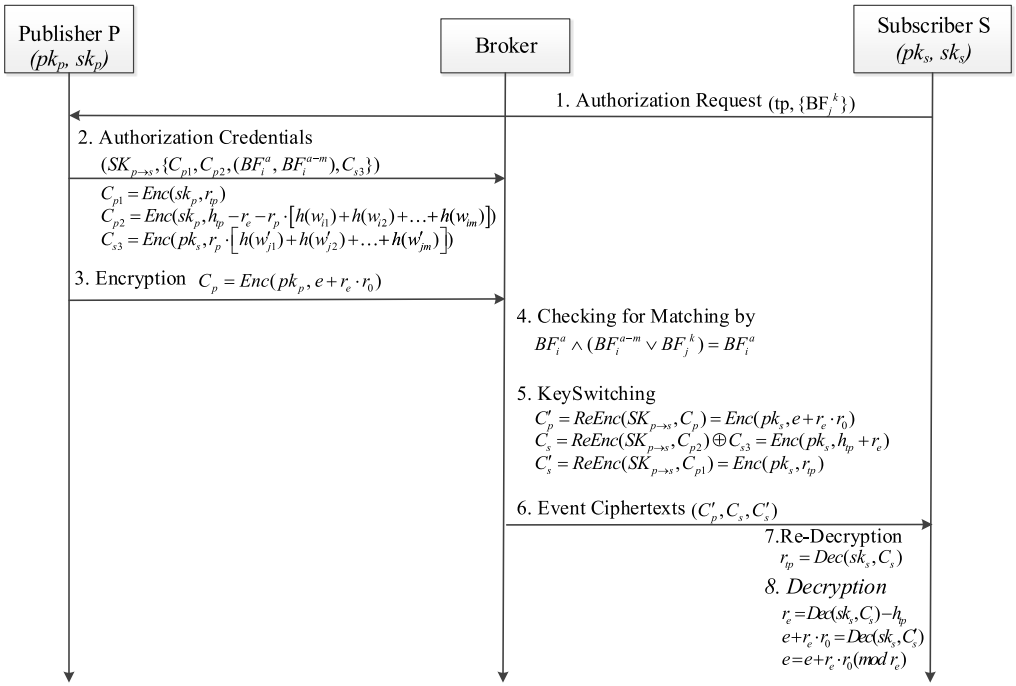


FIGURE 5. Interactive sequences in CACF.

For example, for the $i-th$ $(1 \leq i \leq n)$ access conjunction Γ_i in Γ, where $\Gamma_i = w_{i1} \wedge \ldots \wedge w_{im}$, the attributes in Γ_i form a set $W_i = \{w_{i1}, w_{i2}, \ldots, w_{im}\}$. For each $w_{il} \in W_i$ $(1 \leq l \leq m)$, we choose a random string r as an alias of w_{il} and add each blinded attribute rw_{il} into an anonymous attribute set $Set_i^a = \{w_{i1}, w_{i2}, \ldots, w_{im}, rw_{i1}, rw_{i2}, \ldots, rw_{im}\}$. Replacing w_{il} with rw_{il} to generate a masked attribute set $Set_i^{a-m} = \{rw_{i1}, rw_{i2}, \cdots, rw_{im}\}$. Here, r is kept secret such that all elements in Set_i^{a-m} are unknown to the notification brokers, clients and adversaries. For the attribute conjunction γ of a subscriber, and its $j-th$ $(1 \leq j \leq m)$ conjunction $\gamma_j = w'_{j1} \wedge \ldots \wedge w'_{jm}$, all attributes in γ_j form a set $W'_j = \{w'_{j1}, w'_{j2}, \ldots, w'_{jm}\}$. According to the definition and the generating method of auxiliary sets in [19], we obtains its auxiliary attributes set $Set_j^1, \ldots, Set_j^k, \ldots, Set_j^{n-m}$, where $Set_j^k = \{w'_{j1}, \ldots, w'_{j(m-k)}, w'_{j(m-k+2)}, \ldots, w'_{jm}\}$.

Basing on a bloom filter with the size m, the attribute set Set_i^a and the blinded attribute set Set_i^{a-m} are encoded into BF_i^a and BF_i^{a-m} respectively. Each auxiliary attribute set Set_j^k is encoded into BF_j^k. When BF_i^a is masked, checking whether the attribute set Set_j^k $(1 \leq k \leq n-m)$ is included in Set_i^a is difficult. Thus we encode the blinded mask set Set_i^{a-m} into an independent Bloom Filter BF_i^{a-m}. Because one-way hash functions are used in BF, it is difficult to remove the mask strings. Through "OR" and "AND" operations with the mask strings, checking the operations of the inclusion relationship are carried out by the equation $BF_i^a \wedge (BF_i^{a-m} \vee BF_j^k) = BF_i^a$. If the equation holds, then we can decide that the attribute set W'_j is included in the attribute set W_i.

C. ACCESS CONTROL PROCEDURE

The access control procedure contains three phrases: *initial phrase, policy authorization phrase* and *policy enforcement phrase.* Assume a publisher P has a key pair (pk_p, sk_p), and a subscriber S has a key pair (pk_s, sk_s). The data flow in CACF is illustrated in Fig. 5.

At the *initial phase* of the access control procedure, for each type of event e, P chooses a random number r_e as a temporary identifier, where r_e is bigger than the event e. P chooses another random number r_p as a policy identifier, h is a hash function used in CACF. The *authorization phase* consists of sending a subscription request by the subscriber (*Phase 1*) and generating the access credentials (*Phase 2*). The *policy enforcement phase* involves publishing events by publisher (*Phase 3*), policy matching (*Phase 4*), key switching (*Phase 5*) and sending converted ciphertext for enabling event access by the broker (*Phase 6*), as well as event decryption by the subscriber (*Phase 7*).

Phase 1: Before sending an authorization request to a type of event *tp*, the subscriber first specifies the subscription policy over the requested event type *tp*, and then the subscriber encodes its subscription policy and each attribute into BF_j^k, respectively.

Phase 2: For each allowed requested event type *tp*, the publisher P chooses a random number r_{tp} to randomize each attribute conjunction $w_{i1} \wedge \ldots \wedge w_{im}$ $(1 \leq i \leq n)$ in Γ_{tp}, such that $h_{tp} = h(w_{i1}|w_{i2}|\ldots|w_{im}|r_{tp})$, where "|" is a connector. Even though other types of events may have the same attribute conjunctions in their policies, h_{tp} will be different given the random number r_{tp}. Meanwhile, the

publisher P creates an access credential AC that will be sent to the broker, the generation of the credential is independent of each attribute conjunction $w_{i1} \wedge \ldots \wedge w_{im}$ $(1 \leq i \leq n)$ in Γ_{tp} and the subscriber S's attribute conjunction γ. AC is denoted by $AC \triangleq (SW_{p\to s}, \{C_{p1}, C_{p2}, (BF_i^a, BF_i^{a-m}), C_{s3}\})$, it includes a switching key $SW_{p\to s}$, the encoded attribute sets BF_i^a, BF_i^{a-m}, as well as three ciphertexts C_{p1}, C_{p2} and C_{s3}. In AC,

$$\begin{aligned} SW_{p\to s} &= \texttt{KeySwitch}(pk_s, sk_p), \\ C_{p1} &= Enc(sk_p, r_{tp}), \\ C_{p2} &= Enc(sk_p, h_{tp} + r_e - r_p \cdot (h(w_{i1}) + \ldots + h(w_{im}))). \end{aligned}$$

According to the procedure of policy encoding, P generates BF_i^a and BF_i^{a-m} by encoding the auxiliary attributes set Set_i^a and the mask set Set_i^{a-m}. The results of BF_i^a and BF_i^{a-m} are obtained by using the operation $\texttt{genBF}(w_{i1} \wedge \cdots \wedge w_{im})$.

For each attribute conjunction $\gamma_j = w'_{j1} \wedge \ldots \wedge w'_{jm}$ in γ, P chooses a new random number r_p,

$$C_{s3} = Enc(pk_s, r_p \cdot (h(w'_{j1}) + \ldots + h(w'_{jm}))).$$

At the end of the authorization phase, the access credential AC is sent to the broker at which S is registered.

Phase 3: Before P publishes an event e with the type tp to a broker, P encrypts the event e under a new random number r_0 for e to get the ciphertext C_p, then

$$C_p = Enc(sk_p, e + r_e \cdot r_0).$$

Phase 4: When the broker receives an encrypted event C_p from P to S, it checks for matching by the equation $BF_i^a \wedge (BF_i^{a-m} \vee BF_j^k) = BF_i^a$. That is to say, the broker checks whether the subscriber has an attribute conjunction γ_j in γ that satisfies $\Gamma_i \in \Gamma$. If the matching result is positive, then the subscription request is allowed. The subscriber is a qualified (or authorized) user.

Phase 5: For the qualified subscriber S, the broker converts C_p under P's public key into the event ciphertext C'_p under S's public key by using the keyswitching operation in the FHE. Beyond that, the broker converts the access credential AC for enabling S's access by using the keyswitching operation and homomorphic operations in the FHE, where the converted event ciphertext

$$C'_p = ReEnc(SW_{p\to s}, C_p) = Enc(pk_s, e + r_e \cdot r_0).$$

The converted results of the access credential include C_s and C'_s, where

$$\begin{aligned} C_s &= ReEnc(SW_{p\to s}, C_{p2}) \oplus C_{s3} = Enc(pk_s, h_{tp} + r_e), \\ C'_s &= ReEnc(SW_{p\to s}, C_{p1}) = Enc(pk_s, r_{tp}). \end{aligned}$$

Phase 6: The broker sends event ciphertexs (C'_p, C_s, C'_s) to its subscriber S for enabling S's access.

The event e can be recovered by decrypting C_p to obtain $e + r_e \cdot r_0$, and then conducting the modulo operation $e + r_e \cdot r_0$ $(\texttt{mod}\ r_e)$ to remove $r_e \cdot r_0$.

Phase 7: Upon receiving C'_p, C_s, and C'_s from the broker, S can easily recover r_e from the access credentials. S first decrypts C_{p1} and obtains the random r_{tp}, meaning that the identifier r_{tp} of the event type tp should be used in the decryption. Because γ_j satisfies Γ_i, γ_j is equal to $w_i = w_{i1} \wedge \ldots \wedge w_{im}$, S can decrypt ciphertext $C_{s'}$ to obtain the random number r_e. The steps are

$$\begin{aligned} h_{tp} &= h(w'_{j1} | \ldots | w'_{jm} | r_{tp}), \\ r_e &= Dec(sk_s, C_s) - h_{tp}. \end{aligned}$$

At least, S decrypts C'_s to obtain $e + r_e \cdot r_0$, and recovers e through a modulo operation with respect to r_e.

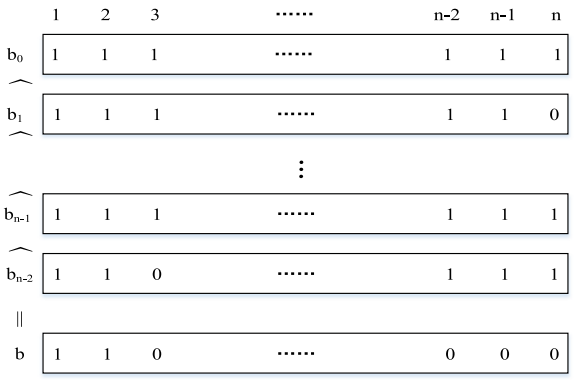


FIGURE 6. Illustration of matching.

IV. ANALYSIS OF THE PROPOSED FRAMEWORK

In this section, we first analyze the correctness of our CACF, and then analyze the properties of the security and privacy of our proposed CACF with the following theorems.

A. CORRECTNESS ANALYSIS

Theorem 1 (Matching Correctness): Given the encoded authorization policy BF_i^a, BF_i^{a-m} and the encoded attribute BF_j^k, if the matching equation $BF_i^a \wedge (BF_i^{a-m} \vee BF_j^k) = BF_i^a$ holds, then $W'_j \subseteq W_i$.

Proof: According to the construction of BF_x $(x \in \{i^a, i^{a-m}, j^k\})$, We will present an example to illustrate the matching correctness. Let $\Gamma_i = w_{i1} \wedge w_{i2}$. Based on the random number r, its masked attributes set is $Set_i^a = \{w_{i1}, w_{i2}, rw_{i1}, rw_{i2}\}$, and the masked set is $Set_i^{a-m} = \{rw_{i1}, rw_{i2}\}$. Set_i^a and Set_i^{a-m} are encoded to BF_i^a and BF_i^{a-m}, respectively. Each BF_j^k is appended by an n-bit binary string b_i. Each position in b_i corresponds w_i, that is to say, if w_i appears in the attribute set, the $i-th$ position in b_i is 1; otherwise, the $i-th$ position in b_i is 0. Thus, if $BF_j^1, BF_j^2, \ldots, BF_j^k, \ldots, BF_{n-2}$ make $BF_i^a \wedge (BF_i^{a-m} \vee BF_j^k) = BF_i^a$ hold, then the computation of b is as shown in Fig. 6. If $BF_i^a \wedge (BF_i^{a-m} \vee BF_j^k) = BF_i^a$, then $b = b_0 \wedge b_0 \wedge \ldots \wedge b_{n-1} \wedge b_{n-2}$. From Fig. 6, we can obtain $b = 1100\ldots00$, that is $\{w_{i1}, w_{i2}\} \subseteq Set_a$, i.e., $W'_j \subseteq W_i$. Therefore, the bi-directional policy matching scheme is correct.

Theorem 2 (Authorization Correctness): For the policy Γ of an event e with a type tp and an attribute conjunction γ of

a subscriber S, if $w_{i1} = w'_{j1}, \ldots$, and $w_{im} = w'_{jm}$, then S can access all events of topic tp in CACF.

Proof: Let $\Gamma_i = w_{i1} \wedge \cdots \wedge w_{im}$ and $\gamma_j = w'_{j1} \wedge \cdots \wedge w'_{jm}$. Because $w_{i1} = w'_{j1}, \ldots$, and $w_{im} = w'_{jm}$, the broker can determine the equation $BF_i^a \wedge (BF_i^{a-m} \vee BF_j) = BF_i^a$ is true. Based on the definition of access credentials $(SW_{p\to s}, \{C_{p1}, C_{p2}, (BF_i^a, BF_i^{a-m}), C_{s3}\})$, the broker will create $SW_{p\to s}, \{C_{p1}, C_{p2}, C_{s3}\}$ for S. According to the keyswitching and decryption operations, S can get r_{tp}. Next, S can recover the event e by decrypting C_p to obtain $e + r_e \cdot r_0$, and then conducting the modulo operation $e + r_e \cdot r_0 \ (\mathtt{mod}\ r_e)$ to remove $r_e \cdot r_0$.

The subscriber can successfully obtain the requested event if its attributes match the publisher's authorization policy. Analogously, we can prove that the subscriber accepts the subscribed event from the published event type if the event attributes match the subscriber's subscription policy.

B. SECURITY & PRIVACY ANALYSIS

For the application layers, the bi-directional policy matching scheme keeps non-interaction CPA security. That is to say, if two different attribute conjunctions have non-overlapping attributes, then any adversary is unable to distinguish the encodings of the two attribute conjunctions. The detailed proof of non-interaction CPA security can be referred to [19].

For the data layer, our CACF ensures that only authorized subscribers can access the events. In other words, only subscribers whose attributes satisfy the authorization policy of publishers, can access the events. Besides, CACF can resist network attacks such as spoofing attacks, man-in-the-middle attacks and compromised-key attacks. Thus, we will analyze the following types of possible security attacks: (1) collusion attacks: two subscribing services collude to recover the event that they cannot individually access, (2) a broker uses the stored access credentials to access the secret event, (3) spoofing attacks, (4) man-in-the-middle attacks, and (5) compromised-key attacks.

Theorem 3: CACF resists collusion attacks.

Proof: For two subscribing services S_1 and S_2, if both S_1 and S_2 cannot access the event e, then S_1 and S_2 will not collude to recover the event e. Since S_1's and S_2's attributes do not satisfy the access policy of e individually, the broker cannot verify policy matching equation $BF_i^a \wedge (BF_i^{a-m} \vee BF_j^k) = BF_i^a$ successfully. Thus, the publisher will not generate access credentials for them, and the broker will not switch the encrypted event. If the collusion of S_1 and S_2 makes the publisher P generate C_{p1}, C_{p2} and C_{s3} for S_1, and C'_{p1}, C'_{p2} and C'_{s3} for S_2; S_1 and S_2 will not recover e successfully. Because the attribute conjunction of S_1 (or S_2) is not equal to the corresponding attribute conjunction $w_i = w_{i1} + w_{i2} + \ldots + w_{im}$, the broker will not obtain $C_s = ReEnc(SW_{p\to s}, C_{p2}) \oplus C_{s3} = Enc(pk_s, h_{tp} + r_e)$ through homomorphic operations. r_e and r_p are secret, thus S_1 and S_2 cannot recover the event e.

Theorem 4: The broker resists using the stored access credentials to access the secret event.

Proof: The broker can perform the keyswitching operations by using the stored switching key $SK_{p\to s}$. It first switch event ciphertext $C_p = Enc(sk_p, e + r_e \cdot r_0)$ to the ciphertext $C'_p = Enc(pk_s, e + r_e \cdot r_0)$. The broker then removes the random number r_p and embedded attributes from the event ciphertext C_p through the homomorphic addition operations $C_s = ReEnc(SW_{p\to s}, C_{p2}) \oplus C_{s3} = Enc(pk_s, h_{tp} + r_e)$ Last, the broker converts the secret r_{tp} under P's public key into the secret r_{tp} under S's public key, i.e., $C'_s = ReEnc(SW_{p\to s}, C_{p1}) = Enc(pk_s, r_{tp})$. However, the broker does not have the private key of the subscriber S, and it cannot receive the random number r_{tp} and recover the random h_{tp} which is used in the hash function on attributes. Thus, the broker will not obtain the random number r_e and recover the event e, preventing the curious behaviour of the broker from leaking secret information.

Theorem 5: CACF resists spoofing attacks.

Proof: Our framework adopts a general authentication method to assess the identity of subscribers. There are four types of spoofing attacks. First, a subscriber S cannot fake its identity to get the information of event e, because the system needs to verify the identity of S when S sends a subscription request with an event type tp. Second, the broker cannot fake the access credentials stored on it to recover event e, because the broker does not have any private keys of the subscribers to remove random numbers and decrypt ciphertexts. Third, two subscribers S_1 and S_2 cannot collude with each other to get the unauthorized event e, because P generates the switching key for each authorized subscriber. Event if S_1 and S_2 have qualified attributes according to the policy, the broker will not have a switching key to convert an encrypted event for them. Lastly, the broker colludes with the subscribers to mislead publishers to get some secret information by itself or send any ciphertext to any broker. For this spoofing attack, we cannot prevent it. Thus, we assume that the broker is honest-but-curious and does not collude with subscribers.

Theorem 6: CACF resists man-in-the-middle attacks.

Proof: First, the attacker cannot replace the broker with access credentials with a broker without any credential to make an unauthorized subscriber to get secret information of event e. If P authorizes S_1 to access to the event e and does not authorize S_2 to access e, the broker of S_2 replaces the broker with access credentials $(SW_{p\to s}, \{(C_{p1}, C_{p2}, (BF_i^a, BF_i^{a-m}), C_{s3}\})$ for S_1. $SW_{p\to s}$ does not convert the encrypted event to S_2. S_2 cannot get the information of e. Second, multiple brokers team up to act as a broker to forge the access credential of S_1, because there are different random numbers for different attribute conjunctions, i.e., for each attribute conjunction $w_{i1} \wedge \ldots \wedge w_{im}$ from the policy of P. P chooses a new random number r_{tp} and generates $h_{tp} = h(w_{i1}|w_{i2}|\ldots|w_{im}|r_{tp})$. For each attribute conjunction $\gamma_j = w'_{j1} \wedge \ldots \wedge w'_{jm}$ in γ, P chooses a new random number r_s and calculates $h_s = h(w'_{j1}|\ldots|w'_{jm}|r_s)$. If the broker combines different attribute conjunctions from multiple

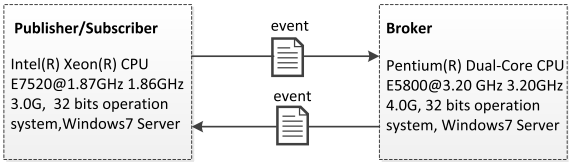


FIGURE 7. Testing environment.

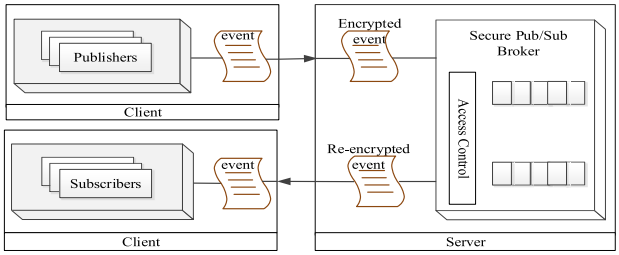


FIGURE 8. Testing design.

brokers, the notification broker will have unsuccessful bloom filter checks. Hence, the broker will not convert the encrypted event and access credentials for them. Thus, S_1 still cannot get additional information.

Theorem 7: CACF resists compromised-key attack.

Proof: When the private key of a subscriber is compromised, this does not have an impact on other subscribers. If both S_1 and S_2 are authorized to subscribe the event e published by P, the broker generates switching keys $SW_{p\to s_1}$ for S_1 and $SW_{p\to s_2}$ for S_2; i.e., S_1 and S_2 do not need to share keys. Thus, if the secret key of S_1 is compromised, it will not affect receiving event e for S_2.

V. PERFORMANCE EVALUATION

Based on Apache ActiveMQ [26] which is a message-oriented Java Message Service (JMS) broker, we have prototyped CACF. The performance of the prototype is evaluated.

A. IMPLEMENTATION & CONFIGURATION

All experiments are performed in a distributed setup, where the detailed configuration used in our tests is shown in Fig. 7. To evaluate precisely in a publish/subscribe system, both publisher clients and subscriber clients ran on the same computer with the configuration of 3.0G of RAM, Intel 1.87GHz CPUs. Windows7_32 operating system. The broker ran on another server equipped with 4.0G of RAM and Intel 3.2GHz CPUs, again Windows7_32 operating system was used. All servers are connected via a standard 100Mbps LAN.

Due to the real-time communication requirement in a publish/subscribe-based IoT service system, we adopt a fully homomorphic encryption scheme described in [17], which can encrypt large integers efficiently. In this scheme, we chose a random prime as the encryption modulus, where the size of the random prime is larger than 5K bytes. Thus, the encrypted event data can be 5K bytes.

In our experimental tests, we extended Apache ActiveMQ with the policy matching capability and the key switching capability. The framework is shown in Fig. 8. The broker is responsible for: (1) checking matching between the authorization policies and attributes of each published event and each subscribing service; (2) key switching for converting the encrypted events from the publisher to the re-encrypted events, which is then decrypted by the subscriber; (3) filtering event by routing events to the authorized subscribers.

To save effort, we extend the publish/subscribe system with building in CACF as the ''secure publish/subscribe system''. *Latency* and *throughput* are chosen as the main performance metrics in our evaluation. Here, we consider the following three types of latency:

- *Pub-to-Sub Latency* without *CACF (Baseline)*, which is defined as the entire devotion in generating an event from its publisher to its subscriber, including the time spent in finding the event type on the broker.
- *Pub-to-Sub Latency* with *CACF* (Sec Pub/Sub), which consists of the whole time spent in the basic event processing (i.e., baseline), as well as the time spent in building CACF. The latency includes (1) the latency of performing encryption operations on the event publisher; (2) the latency of performing key switching operations and performing matching operations on the broker; and (3) the latency of the decryption operation on the subscriber.
- *Throughput*, is defined to be the average number of the published events in one second.

To evaluate the overhead of event communication from publishers to subscribers and the scalability of the secure publish/subscribe system, we measure the latency in the baseline and the secure publish/subscribe system by the experiments in three specified test cases:

1) Evaluating the latency in terms of the increasing size of a published event;
2) Evaluating the latency in terms of the increasing number of authorization policies in an event;
3) Evaluating the latency in terms of the increasing number of attributes in policy for the subscriber.

B. EVALUATION RESULTS

In our tests, we measure the average latency for an event under different sizes of event data, different numbers of policies in an event and different numbers of rules at a subscriber. We compare these latency metrics between the baseline and the proposed secure publish/subscribe system (sec pub/sub). All test cases run 1000 times. The other parameters of each experimental test are summarized in Table 1.

In the first test, we vary the sizes of event data from $1KB$ to $5KB$, and fix one policy and one attribute at the broker, the test result is shown in Fig. 9. From the figure, we can see that the Pub-to-Sub latency of sec pub/sub is about $37ms \sim 42ms$, the latency increases by about $12ms$, as compared to the baseline.

In the second test, for $1KB$ data event, we measure the average latency with two attributes on the subscriber, the average

TABLE 1. Test Cases.

Test Cases	Description	Event Size (KB)	No. of Policies	No. of attributers in one subscriber
(1)	For 1 policy and 1 rule in an event, evaluate the latencies with different event publishing rates.	1, 2, 3, 4, 5	1	1
(2)	For 1KB data event size, evaluate the latencies with different numbers of policies in an event, where the number of the rule in one policy is 2.	1	1, 2, 4, 6, 8, 10	2
(3)	For 1KB data event size, evaluate the latencies with different numbers of rules in one policy, where the number of policy in one event is 1.	1	1	1, 5, 10, 15, 20, 25

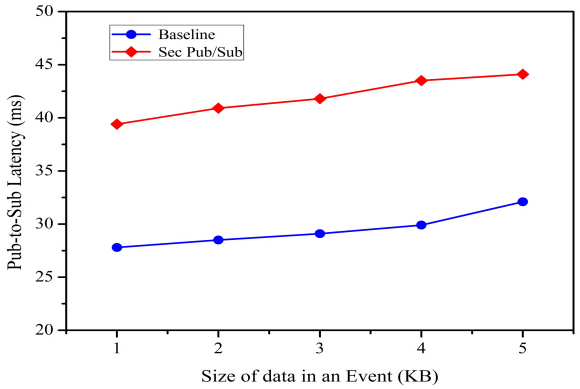


FIGURE 9. Latency with different size of one event (KB).

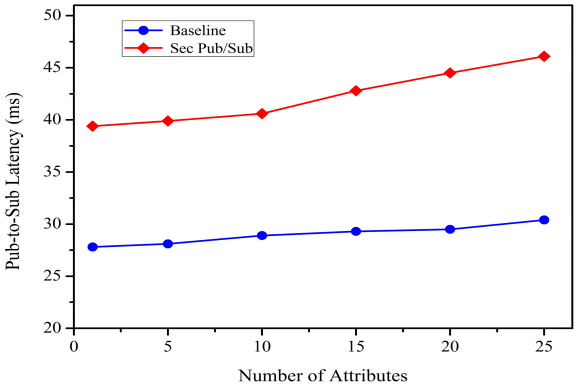


FIGURE 11. Latency with different number of attributes on one Subscriber.

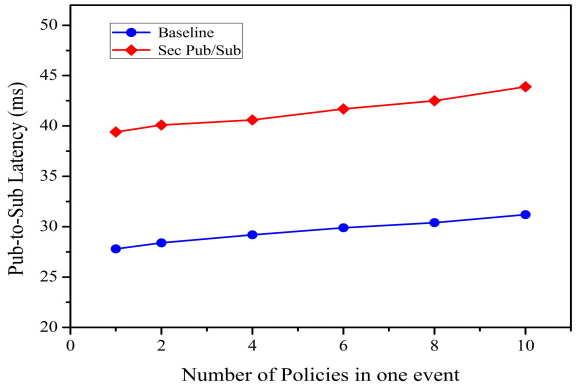


FIGURE 10. Latency with different number of policies in one event.

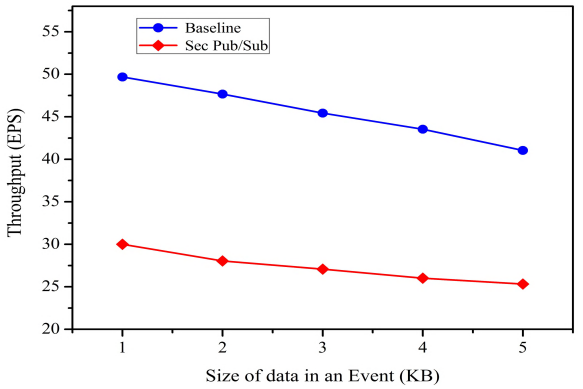


FIGURE 12. Throughput for different data event sizes in KB.

latency with different numbers of policies 2, 4, 6, 8, 10 on the horizontal axis is shown in Fig. 10. From the figure, we can see that the Pub-to-Sub latency of sec pub/sub is about $38ms \sim 45ms$, the latency increases by about $14ms$, as compared to the baseline. This latency is a little higher than the latency of the first test, since the second test has more attributes than the first test.

In the third test, for $1KB$ data event with 10 policies, we measure the average latency with different numbers of attributes on one subscriber, as shown in Fig. 11. From Fig. 11, we can see that the Pub-to-Sub latency of sec pub/sub is about $39ms \sim 46ms$, the latency increases by about $16ms$, as compared to the baseline. As the number of attributes increases, the latency exhibits a continuous curve. This latency is increasingly faster than the previous two test cases, since it takes more time to check matching.

By means of the "Little's Law", the throughput in Events Per Second (EPS) can be derived as "$Throughput = \frac{1}{Latency}$". Based on the average pub-to-sub latencies and the latencies of the baseline, we present the average sustainable pub-to-sub throughput results in Fig. 12.

The pub-to-sub throughput results are based on the average pub-to-sub latencies with or without access control. Fig. 12 shows the average sustainable throughput in processing events per second using different event sizes; the horizontal axis is in base-10 logarithm. The size of the

data event is one of the main factors that affect pub-to-sub latencies. With the growth of data event sizes, the pub-to-sub throughput decreases, that is to say, fewer data events per second can be sent from the publisher to the subscriber.

From the above security analysis and latency evaluation results, the overhead incurred for preserving the publish/subscribe system is reasonable and acceptable. The overall latency comparison shows that our access control framework has higher policy matching efficiency and higher scalability.

VI. RELATED WORK AND DISCUSSION

In the context of publish/subscribe systems, the security requirements mainly include the privacy of participant services, the data confidentiality and access control. There are extensive studies [9], [18]–[21], [27], [28] on these security issues of publish/subscribe services.

The cryptographic encryption solution is one of privacy-preserving techniques used in distributed systems. Depending on Ciphertext Policy Attribute based Encryption (CP-ABE), Tariq *et al*. [29] propose an access control scheme for a brokerless publish/subscribe system, in which each publisher and subscriber has a separate private key and a public key for the authorization credential. The work described in [30] studies on preserving subscription privacy in publish/subscribe systems, which is limited to fine-grained access control for the published event. Opyrchal and Prakash [31] focus on publication privacy in publish/subscribe systems by providing access control on publications. Ion *et al*. [32] propose an access control framework by combining an attribute-based encryption scheme and a multi-user searchable data encryption scheme. This framework allows multiple users to encrypt data and to make queries. However, these solutions require participant services to coordinate to establish the encryption and decryption keys, and the network broker should decrypt the event from its publisher and then encrypt it again for sending to the subscribers.

Homomorphic encryption [33] is a novel approach for preserving privacy in publish/subscribe systems. It supports complex computation conducted on the broker, but it is not practical. With the Ring-LWE (RLWE) key switching approach from the homomorphic encryption, Yuriy *et al*. [34] propose two IND-CPA-secure multi-hop unidirectional Proxy Re-Encryption (PRE) schemes. The schemes keep the decoupling features of publishers and subscribers, as well as preventing the broker access to the unprotected event. These security schemes focus on the privacy of subscription or the privacy of publishers, and rarely support comprehensive privacy protection of published events and the subscription at the same time.

Goyal *et al*. [35] provide a Key-Policy ABE scheme, which allowed the policies (attached to keys) to be expressed by any monotonic formula over encrypted attributes (ciphertext). Waters [36] propose the Ciphertext-Policy Attribute Encryption (CP- ABE) scheme, where any encryptor is allowed to specify access control by using any access formula on the attributes in the system. However, in these approaches, the CP-ABE scheme embeds authorization policies into ciphertexts. Such security schemes in publish/subscribe systems require that a publisher has many keys, where each publisher gives the subscriber a key. It does not allow for using notification brokers to reduce the key management burden of the participant, and does not preserve the decoupling feature between service providers and consumers.

The differential privacy technique is adopted to resolve the privacy and security of users' data in [28], with focus on uncertain datasets of users. The DDS security standards [15] include an access control DDS-AC scheme for publish/subscribe systems. A publisher in DDS-AC can set up topic domains, and decide whether a subscriber can join the domain to access events. DDS-AC adopts the traditional access control method and assumes that there is an omnipotent supervisor to supervise access to data. This requires authorization delegation. In our solution, there is no such an assumption or limitation.

This paper is a continuation of our work [27], where a data-centric access control framework (DCACF) is given and data confidentiality is thoroughly addressed. Based on DCACF, we add the description of embedding policy and preserving policy. Different from these studies, in this paper, we focus on security requirements of privacy-preserving and confidentiality-protecting aspects. In publish/subscribe-based IoT services communication, we adopt policy encoding and matching techniques, as well as fully homomorphic encryption scheme, to achieve our security goals.

VII. CONCLUSION AND FUTURE WORK

The publish/subscribe paradigm provides a loosely-coupled and scalable communication model for collaborative IoT services. In this paper, we propose a comprehensive access control framework, CACF, for publish/subscribe-based IoT services communication. Our framework preserves the data confidentiality by adopting the FHE scheme which supports homomorphic operations on encrypted data. Then, we adopt privacy-preserving matching technology on the access policy specified by publishing services and the subscription policy specified by subscribing services. In our framework, our security solution preserves the direct, anonymous, and multicast features of publish/subscribe services. There is no need to share keys between collaborative services, which reduces the burden of key management. We demonstrate that CACF is correct and secure in the standard model. The performance evaluation results show that our comprehensive framework can provide the privacy-preserving and confidentiality-protecting capabilities with acceptable latency.

For future work, we will extend the framework to address the security aspects in SDN (Software Defined Networking) based publish/subscribe services.

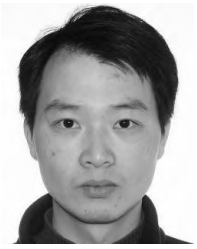
REFERENCES

[1] S. Li, L. Xu, and S. Zhao, "The Internet of Things: A survey," *Inf. Syst. Frontiers*, vol. 17, no. 2, pp. 243–259, 2015.
[2] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A survey on enabling technologies, protocols, and applications," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 2347–2376, 4th Quart., 2015.
[3] P. T. Eugster, P. A. Felber, R. Guerraoui, and A.-M. Kermarrec, "The many faces of publish/subscribe," *ACM Comput. Surv.*, vol. 35, no. 2, pp. 114–131, 2003.
[4] V. Karagiannis, P. Chatzimisios, F. Vazquez-Gallego, and J. Alonso-Zarate, "A survey on application layer protocols for the Internet of Things," *Trans. IoT Cloud Comput.*, vol. 3, no. 1, pp. 11–17, 2015.
[5] F. T. El-Hassan and D. Ionescu, "Design and implementation of a hardware versatile publish-subscribe architecture for the Internet of Things," *IEEE Access*, vol. 6, pp. 31872–31890, 2018.
[6] W. K. Chai *et al.*, "An information-centric communication infrastructure for real-time state estimation of active distribution networks," *IEEE Trans. Smart Grid*, vol. 6, no. 4, pp. 2134–2146, Jul. 2015.
[7] J. J. Q. Yu, A. Y. S. Lam, D. J. Hill, and V. O. K. Li, "A unified framework for wide area measurement system planning," *Int. J. Elect. Power Energy Syst.*, vol. 96, pp. 43–51, Mar. 2018.
[8] E. Onica, P. Felber, H. Mercier, and E. Rivière, "Confidentiality-preserving publish/subscribe: A survey," *ACM Comput. Surv.*, vol. 49, no. 2, pp. 1–43, 2016.
[9] R. Barazzutti, P. Felber, H. Mercier, E. Onica, and E. Riviere, "Efficient and confidentiality-preserving content-based publish/subscribe with prefiltering," *IEEE Trans. Dependable Secure Comput.*, vol. 14, no. 3, pp. 308–325, May 2017.
[10] S. Kraijak and P. Tuwanut, "A survey on IoT architectures, protocols, applications, security, privacy, real-world implementation and future trends," in *Proc. 16th Int. Conf. Commun. Technol.*, 2015, pp. 18–20.
[11] M. A. Márquez, R. S. Herrera, A. Mejías, F. Esquembre, and J. M. Andújar, "Controlled and secure access to promote the industrial Internet of Things," *IEEE Access*, vol. 6, pp. 48:289–48:299, 2018.
[12] C. Esposito and M. Ciampi, "On security in publish/subscribe services: A survey," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 2, pp. 966–997, 2nd Quart., 2015.
[13] V. C. Hu, D. R. Kuhn, and D. F. Ferraiolo, "Attribute-based access control," *Computer*, vol. 48, no. 2, pp. 85–88, Feb. 2015.
[14] F. Paci, M. Mecella, M. Ouzzani, and E. Bertino, "Acconv–an access control model for conversational web services," *ACM Trans. Web*, vol. 5, no. 3, pp. 13:1–13:33, 2011.
[15] *DDS Security Specification Version 1.0—Beta 1*, OM Group, Jaipur, India, 2014.
[16] Z. Wang, D. Huang, Y. Zhu, B. Li, and C.-J. Chung, "Efficient attribute-based comparable data access control," *IEEE Trans. Comput.*, vol. 64, no. 12, pp. 3430–3443, Dec. 2015.
[17] D. Liu, "Practical fully homomorphic encryption without noise reduction," IACR Cryptol. ePrint Arch., Santa Barbara, CA, USA, Tech. Rep. 2015/468, 2015, pp. 1–18.
[18] E. Onica, P. Felber, H. Mercier, and E. Rivière, "Efficient key updates through subscription re-encryption for privacy-preserving publish/subscribe," in *Proc. 16th Annu. Middleware Conf.*, 2015, pp. 25–36.
[19] L. Duan, Y. Zhang, S. Chen, S. Wang, B. Cheng, and J. Chen, "Realizing IoT service's policy privacy over publish/subscribe-based middleware," *SpringerPlus*, vol. 5, no. 1, p. 1615, 2016.
[20] E. Onica *et al.*, "Efficient key updates through subscription re-encryption for privacy-preserving publish/subscribe," in *Proc. 16th Annu. Middleware Conf.*, 2015, pp. 25–36.
[21] M. Li, S. Yu, Y. Zheng, K. Ren, and W. Lou, "Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 1, pp. 131–143, Jan. 2013.
[22] M. Richards, R. Monson-Haefel, and D. A. Chappell, *Java Message Service: Creating Distributed Enterprise Applications*. Newton, MA, USA: O'Reilly Media, 2009.
[23] D. Liu, "Efficient processing of encrypted data in honest-but-curious clouds," in *Proc. IEEE 9th Int. Conf. Cloud Comput. (CLOUD)*, Jun. 2016, pp. 970–974.
[24] Z. Brakerski and V. Vaikuntanathan, "Efficient fully homomorphic encryption from (standard) LWE," *SIAM J. Comput.*, vol. 43, no. 2, pp. 831–871, 2014.
[25] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "(Leveled) fully homomorphic encryption without bootstrapping," *ACM Trans. Comput. Theory*, vol. 6, no. 3, pp. 13:1–13:36, 2014.
[26] B. Snyder, D. Bosanac, and R. Davies, "Introduction to apache activemq," in *Proc. Active MQ Action*, 2017, pp. 6–16.
[27] L. Duan *et al.*, "Secure data-centric access control for smart grid services based on publish/subscribe systems," *ACM Trans. Internet Technol.*, vol. 16, no. 4, pp. 1–17, 2016.
[28] Q. Wang, D. Chen, N. Zhang, Z. Ding, and Z. Qin, "PCP: A privacy-preserving content-based publish–subscribe scheme with differential privacy in fog computing," *IEEE Access*, vol. 5, pp. 17962–17974, 2017.
[29] M. A. Tariq, B. Koldehofe, and K. Rothermel, "Securing broker-less publish/subscribe systems using identity-based encryption," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 2, pp. 518–528, Feb. 2014.
[30] W. Rao, L. Chen, and S. Tarkoma, "Toward efficient filter privacy-aware content-based pub/sub systems," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 11, pp. 2644–2657, Nov. 2013.
[31] L. Opyrchal and A. Prakash, "Secure distribution of events in content-based publish subscribe systems," in *Proc. 10th Conf. USENIX Secur. Symp.*, vol. 10, 2001, pp. 1–21.
[32] M. Ion, G. Russello, and B. Crispo, "Design and implementation of a confidentiality and access control solution for publish/subscribe systems," *Comput. Netw.*, vol. 56, no. 7, pp. 2014–2037, 2012.
[33] M. Van Dijk *et al.*, "Fully homomorphic encryption over the integers," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 2010, pp. 24–43.
[34] Y. Polyakov, K. Rohloff, G. Sahu, and V. Vaikuntanathan, "Fast proxy re-encryption for publish/subscribe systems," *ACM Trans. Privacy Secur.*, vol. 20, no. 4, pp. 1–31, 2017.
[35] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proc. 13th ACM Conf. Comput. Commun. Secur.*, 2006, pp. 89–98.
[36] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," in *Proc. Int. Workshop Public Key Cryptogr.* Berlin, Germany: Springer, 2011, pp. 53–70.

LI DUAN received the Ph.D. degree from the Beijing University of Posts and Telecommunications, in 2016. She is currently a Research Fellow with the Nanyang Technological University, and the University of Science and Technology Beijing. Her research interests are services computing, the Internet of Things, and network service security and privacy.

CHANG-AI SUN received the bachelor's degree in computer science from the University of Science and Technology Beijing, China, and the Ph.D. degree in computer science from Beihang University, China. He was an Assistant Professor with Beijing Jiaotong University, China; a Postdoctoral Fellow with the Swinburne University of Technology, Australia; and a Postdoctoral Fellow with the University of Groningen, The Netherlands. He is currently a Professor with the School of Computer and Communication Engineering, University of Science and Technology Beijing. His research interests include software testing, program analysis, and service-oriented computing.

YANG ZHANG received the Ph.D. degree in computer applied technology from the Institute of Software, Chinese Academy of Sciences, in 2007. He is currently with the State Key laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing, China. His research interests include service oriented computing, the Internet of Things, and service security and privacy.



WEI NI received the B.E. and Ph.D. degrees in electronic engineering from Fudan University, Shanghai, China, in 2000 and 2005, respectively. He is currently the Team Leader of CSIRO, Sydney, Australia, and also an Adjunct Professor with the University of Technology Sydney. He also holds adjunct positions at the University of New South Wales and Macquarie University. Prior to this, he was a Postdoctoral Research Fellow with Shanghai Jiaotong University, from 2005 to 2008; the Deputy Project Manager with the Bell Labs R&I Center, Alcatel/Alcatel-Lucent, from 2005 to 2008; and a Senior Researcher with Devices R&D, Nokia, from 2008 to 2009. His research interests include stochastic optimization, game theory, graph theory, as well as their applications to network and security. He has been serving as the Vice Chair of the IEEE NSW VTS Chapter and the Editor of the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, since 2018; the Secretary of the IEEE NSW VTS Chapter, from 2015 to 2018; the Track Chair of the VTC-Spring 2017l; the Track Co-Chair of the IEEE VTC-Spring 2016; and the Publication Chair of BodyNet 2015. He also served as the Student Travel Grant Chair of WPMC 2014; a Program Committee Member of CHINACOM 2014; and a TPC member of the IEEE ICC'14, ICCC'15, EICE'14, and WCNC'10.

JUNLIANG CHEN is currently a Professor with the Beijing University of Posts and Telecommunications. His research interests are in the area of service creation technology. He was elected as a member of the Chinese Academy of Science, in 1991, and a member of the Chinese Academy of Engineering, in 1994.