Privacy-Preserving Content-Based Publish/Subscribe Networks

Abdullatif Shikfa, Melek Önen, and Refik Molva
EURECOM, France

Abstract Privacy and confidentiality are crucial issues in content-based publish/subscribe (CBPS) networks. We tackle the problem of end-user privacy in CBPS. This problem raises a challenging requirement for handling encrypted data for the purpose of routing based on protected content and encrypted subscription information. We suggest a solution based on a commutative multiple encryption scheme in order to allow brokers to operate in-network matching and content based routing without having access to the content of the packets. This is the first solution that avoids key sharing among end-users and targets an enhanced CBPS model where brokers can also be subscribers at the same time.

1 Introduction

Publish-subscribe paradigm allows for flexible and dynamic communication among a large number of participants. As opposed to classical messaging systems, in publish-subscribe, communicating parties are loosely coupled in that the source of the information does not need to know potential recipients of the information and the recipients do not need to know where the information originates from. In a content-based publish-subscribe system the forwarding of data segments between the sources and the recipients does not take into account the addresses of communicating parties but is performed based on the relationship between the content of each message and the interest of recipients. The recipients who inform the publish-subscribe system about the messages they are interested in through subscription messages are thus called subscribers. Publish-subscribe applications range from large scale content distribution applications such as stock-quote distribution to dynamic messaging between loosely-coupled parties in on-line social networks.

The flexibility of publish-subscribe comes on the other hand with a high cost in increased exposure in terms of data security and privacy. Apart from classical data security concerns such as the confidentiality and integrity of messages, the authentication of the source, access control and authorization of subscribers, publish-subscribe also raises new challenges inherent to the collapsed forwarding scheme that is the underpinning of publish-subscribe. In classical layered communication systems, the application layer information can be protected with various security mechanisms like encryption and message authentication without affecting the underlying data forwarding mechanisms implemented in the network layer. In case of publish-subscribe, protection of the content with similar security mechanisms would conflict with the forwarding functions since the latter rely on the very content that is being transmitted for their basic operations. Publish-subscribe therefore calls for new solutions to allow intermediate nodes to perform routing operations based on data protected with encryption and integrity mechanisms. The first requirement is for a secure forwarding

mechanism that would achieve the look-up in forwarding tables using encrypted content as the search key. Furthermore, an important privacy requirement in content-based publish-subscribe is the confidentiality of the messages through which subscribers inform the network about their interests. Whilst encryption of these messages appears to be a suitable solution for subscriber privacy, such encryption operation raises an additional challenge for the forwarding mechanism. Hence not only the search key for the look-up mechanism but also the forwarding table itself would be based on encrypted data. Some existing security primitives such as keyword search with encrypted data or private information retrieval seem to partially meet the new requirements raised by secure and privacy preserving data forwarding in publish-subscribe but none of the existing security mechanisms actually address both the problem of secure look-up and the secure building of forwarding tables in a comprehensive manner.

In this paper, we suggest a set of security mechanisms that allow for privacy-preserving forwarding of encrypted content based on encrypted subscriber interest messages. The main advantages of this solution are that it achieves both data confidentiality from the point of view of the publishers and the privacy of the subscribers with respect to their interests in a potentially hostile model whereby the publishers, the subscribers and the intermediate nodes in charge of data forwarding do not trust one another. The solution relies on a scheme called multi-layer encryption that allows intermediate nodes to manage forwarding tables and to perform content forwarding using encrypted content and based on encrypted subscriber messages without ever accessing the cleartext version of those data. Our solution further avoids key sharing among end-users and targets an enhanced CBPS model where brokers can also be subscribers at the same time.

2 Reference Model and problem statement

2.1 Content-Based Publish/Subscribe (CBPS)

We consider a classical CBPS model as described in many papers like [6, 20]. In this model the CBPS consists of:

- **end-users** divided in:
  - **publishers** which publish information in the form of event notifications,
  - **subscribers** which express their interests in a certain content in the form of subscription filters,
- the CBPS infrastructure composed of **brokers** (intermediate nodes) whose task is to disseminate notifications sent by publishers to the interested subscribers.

We assume that the CBPS infrastructure can be viewed, from the perspective of each publisher, as a tree whose root node is the publisher itself and whose leaf nodes are the subscribers (whether interested in the content published by the publisher or not). Based on this model, we only consider the case of a network with one publisher for the sake of simplicity.

Information contained in each event should fit within an event schema, and the subscription filters are predicates against this schema. Our model of subscription is equality filters with only one keyword and events are composed of two parts: one routable attribute and a second part which is the payload. The equality matching is the mostly used filtering function in the literature since it can be used as a basis to support range queries as introduced in [17]. Brokers use this matching operation between filters and routable attributes to route published content. If we take as an example the commonly used stock quote dissemination problem, a subscription filter could be $(price = 120)$ which would match an event like $(price = 120, [symbol = \text{"}STM\text{"}, price = 120, volume = 1000])$.

In [6], authors show that content-based routing and in-network matching are vital for the performance and scalability of the CBPS system. To this extent, if two subscriptions match the same content, then only one of them should be propagated in the network. We thus need to define a notion of equivalence between filters in order to aggregate them: we say that two filters f_1 and f_2 are equivalent if they match the same events.

As mentioned in the introduction, this paper focus on privacy issues in CBPS, hence we now describe our security assumptions to complete our reference model.

2.2 Threat Model and Security Assumptions

As in many papers (e.g. [20]), we assume a honest-but-curious model for the publishers, the subscribers and the brokers. Publishers, subscribers and brokers are computationally bounded and do not deviate from the designed protocol, but they may be interested in learning more than needed to correctly run the protocol to break subscriber privacy. A curious publisher may indeed be interested in knowing which subscribers are interested in the content it publishes. Subscribers may try to sneak on other subscribers to determine what their interest are or at least if they have some common interests. The same goes for curious brokers which may eavesdrop on the messages routed through them to discover the content of an event notification or a subscription filter.

However, all the nodes are honest and do not deviate from the designed protocol, meaning for instance that brokers correctly route the information they receive as indicated by the protocol, they do not drop packets or forward packets in a wrong way. Denial of service attacks are thus out of the scope of this paper. Furthermore nodes reveal neither their secret nor received data to other nodes. We also take into account malicious but passive nodes outside of the network, which can overhear communications and try to break end-users' privacy.

We now show the link between privacy and confidentiality issues and how the required confidentiality can be achieved through secure routing.

2.3 Privacy, Confidentiality and Secure Routing

In this paper, we focus on the problem of subscriber and publisher privacy. As pointed out in [14] privacy is expected to be a significant concern for acceptance of pervasive environments like CBPS systems. Privacy from the subscriber point of view refers to the fact that subscribers do not want any other nodes, be it brokers, publishers, other subscribers or even nodes outside the CBPS infrastructure, to spy on their interests and be able to profile them in any way. There are several ways of ensuring privacy; one of the classical approaches is to guarantee data confidentiality with cryptographic primitives.

Confidentiality in CBPS networks has first been analyzed in [21] where the authors identify three confidentiality issues, defined as follows:

- **Information confidentiality**: Can the infrastructure perform content-based routing, without the publishers trusting the infrastructure with the content?
- **Subscription confidentiality**: Can subscribers obtain dynamic, content-based data without revealing their subscription filters to the publishers or to the infrastructure?

- **Publication confidentiality**: Can publishers control which subscribers may receive particular publications?

Subscription confidentiality is obviously a must to preserve subscribers privacy but it is not sufficient: we also need to take information confidentiality into consideration, otherwise adversaries could infer the subscription filter by analyzing the information which matches it. From a publisher's perspective privacy may not be as crucial. Publishers publish some content which is meant to be received by some nodes, hence they often do not require a full-fledged privacy but they require publication confidentiality that we mentioned earlier.

Publication confidentiality is an access control rather than privacy issue: publishers want to be able to authorize certain subscribers to be able to access the content they publish while preventing unauthorized ones from learning valuable information about it. Since publication confidentiality is not necessary to ensure privacy, we do not consider it in the sequel of the paper, especially that orthogonal solutions can be developed to ensure it.

To ensure privacy we hence have to fulfill two main confidentiality requirements, namely:

- **Information confidentiality:** this confidentiality requirement may look paradoxical : content-based routing is indeed, by definition, based on evaluations of the content of notifications against subscription filters. The challenge is to be able to perform these evaluations on encrypted data without leaking information on the corresponding content or subscription filter. In the stock quotes example, this corresponds to be able to do content-based routing on an encrypted event where the brokers cannot discover in the event the value of the symbol, price or volume.
- **Subscription confidentiality:** this is the dual problem of information confidentiality. Here, subscribers do not want to reveal their interests either to brokers or publishers or other subscribers but they still want to receive the content they are interested in and only this one. So the challenge in this case is to match a content with an encrypted subscription without disclosing the subscription filter. In the stock quotes example, this requirement corresponds to the ability to find which events match which filter without accessing it in clear; it is a problem of secure function evaluation, where a broker has to evaluate a hidden function (the filter which was encrypted by the subscriber).

In summary, information and subscriber confidentiality in CBPS call for new mechanisms to achieve secure routing of encrypted data with the capability of matching encrypted event notifications against encrypted subscription filters in order to ensure end-users privacy. In further analysis routing of encrypted data in CBPS involves two separate operations:

- **Building routing tables:** Brokers have to build routing tables using routing information -subscription filters- which is classically propagated upwards (from subscribers to publishers) by intermediate nodes to subsequently allow for the routing of content in a possibly optimized fashion. The challenge in our case is that subscription filters are encrypted, hence nodes have to build their routing tables with encrypted filters (to satisfy the subscription confidentiality constraint) and to aggregate theses encrypted routing information. Aggregation of routing tables' entries is not strictly a security concern but is nonetheless a strong requirement from the point of view of performance.
- **Look-up:** Once routing tables are built, nodes can forward data -events' notification- downwards (from publishers to subscribers) in an optimized way through the CBPS infrastructure. The challenge for brokers in the dissemination process is to be able to perform the look-up of encrypted data (to fulfill the information confidentiality requirement) in routing tables where entries include encrypted subscription filters.

CBPS privacy thus calls for a solution that achieves secure routing of encrypted data based on encrypted routing information. One naturally turns to searchable encryption and keyword search [19, 5] that are cryptographic techniques most likely to meet the requirements of secure routing in CBPS. Unfortunately none of the existing searchable encryption and keyword schemes address both the secure forwarding and the table building requirement of CBPS. We tailor in this paper a dedicated solution to meet the specific requirements of CBPS.

3 Secure routing with multiple layer encryption

3.1 Multiple layer commutative encryption (MLCE)

The basic idea behind our solution is to use a MLCE in order to meet the privacy requirements raised by CBPS systems. MLCE allows intermediate nodes in charge of routing secure traffic to perform secure transformations without having access to the data that is being transferred. This feature of MLCE lends itself very well to solving the problem of routing encrypted data as raised by CBPS.

In multiple layer encryption data is encrypted several times with different keys. In the case where the encryption layers all use the same cryptosystem, and if this cryptosystem is commutative, then the layers can be added and removed in any order. An encryption mechanism $\mathscr{E}$ is commutative if, for any data d, any keys k_1, k_2 we have :

$$\mathscr{E}_{k_2}(\mathscr{E}_{k_1}(d)) = \mathscr{E}_{k_1}(\mathscr{E}_{k_2}(d)).$$

We propose to use multiple layer commutative encryption in order to ensure secure routing in CBPS applications where the publisher publishes encrypted events and the subscriber sends its encrypted subscription filter to the source over untrusted brokers. The idea is for the subscriber to encrypt its subscription filter with r layers corresponding to the $r \geq 2$ next hops, and for the publishers to do the same with their event notifications. Brokers en-route remove one encryption layer and add a new one without destroying the other layers so that the data is always protected by at least $r-1$ layers of encryption. Thus brokers do not have access to data in cleartext. Still, this mechanism allows secure look-up as well as efficient and secure routing table building thanks to the commutativity of the layers. The number of layers r is a security parameter that has a performance impact, yet, for the sake of simplicity, we present our scheme only for the case $r = 2$ and discuss the choice of the parameter r in section 5.

To further introduce the solution, let us consider a minimalist example. In this example, we consider three nodes in line, namely a subscriber denoted by S, then a broker denoted by B and finally a publisher denoted by P. We denote by k_{ij} a key shared between node i and j. S encrypts its data x_S with $\mathscr{E}_{k_{SP}}(\mathscr{E}_{k_{SB}}(x_S))$ and so does P with its data x_P: $\mathscr{E}_{k_{SP}}(\mathscr{E}_{k_{BP}}(x_P))$. The broker now can remove the layers corresponding to k_{SB} and k_{BP} respectively to obtain $\mathscr{E}_{k_{SP}}(x_S)$ and $\mathscr{E}_{k_{SP}}(x_P)$. Hence, it cannot access the data directly but it is able to perform a matching operation for the secure look-up since x_S and x_P are encrypted under the same keys.

Therefore, given a commutative cryptosystem we are able to do secure routing and hence protect the privacy of publishers and subscribers. Yet, commutative cryptosystems are very rare, and although many security solutions assume the existence of a commutative cipher, few of them deal with a concrete commutative cryptosystem. We developed a scheme based on the Pohlig-Hellman cryptosystem, that we carefully adapted to our case in order to provide a complete and concrete solution. Privacy-preserving routing with MLCE is achieved through four security primitives that are detailed in the next section.

3.2 Security Primitives

To further refine the privacy-preserving routing using MLCE we identify the generic operations required for secure event dissemination as follows:

- **ENCRYPT_FILTER:** used by subscribers to generate encrypted subscription filters. On input a subscription filter and some keying material it outputs an encrypted version of the subscription filter.
- **ENCRYPT_NOTIFICATION:** used by the publisher to encrypt its notifications. On input an event notification and some keying material it outputs an encrypted version of the subscription filter.
- **SECURE_LOOK_UP:** allows a broker to decide whether an encrypted notification matches one of the encrypted subscriptions of its routing table. This primitive should only return the boolean result of the matching operation.
- **SECURE_TABLE_BUILDING:** allows the broker to build a routing table and to compare two encrypted subscriptions. If two subscriptions match the same content there is indeed no need to forward both of them to the broker's parent. The broker only needs to store both of them with the corresponding child in its routing table and it forwards one to its parent. As the previous primitive, this one should only return the boolean result of the matching operation, but it should not leak any additional information about the subscriptions. The aggregation is optional from a pure privacy point of view (it even induces additional difficulties) but it is vital from a performance point of view to comply with some content-based routing optimizations.

All brokers use the same general message processing to preserve MLCE and manage the security primitives at the same time. This processing is summarized in Table 1. In this table, the broker is denoted by B, its grandparent by G and its grandchild by C, the encryption algorithm is $\mathscr{E}$ and the corresponding decryption algorithm is $\mathscr{D}$. On the left B receives an encrypted subscription filter SF and on the right an encrypted event notification EN. We now formally describe our solution in the next section.

Upwards: filters propagation	**Downwards: event dissemination**
Remove an encryption layer: $\mathscr{D}_{k_{BC}}(SF)$	Remove an encryption layer: $\mathscr{D}_{k_{BG}}(EN)$
Update the routing table RT_B: $SECURE\_TABLE\_BUILDING(RT_B, \mathscr{D}_{k_{BC}}(SF))$	Secure look-up: $SECURE\_LOOK\_UP(RT_B, \mathscr{D}_{k_{BG}}(EN))$
Add an encryption layer: $\mathscr{E}_{k_{BG}}(\mathscr{D}_{k_{BC}}(SF))$	Add an encryption layer: $\mathscr{E}_{k_{BC}}(\mathscr{D}_{k_{BG}}(EN))$
Forward the message upwards	Forward the message downwards

Table 1 Message processing at a broker

4 Privacy and confidentiality in the Hybrid Model

We propose a new solution based on the Pohlig-Hellman cryptosystem whereby subscribers do not need to share a unique and common key K with the publisher. This solution does not differentiate brokers from subscribers and therefore allows brokers to also act as subscribers by subscribing to events and sending their own subscription filters while performing the routing operation.

In order to formally define the four security primitives required to securely forward filters and events, we first remind the Pohlig-Hellman cryptosystem and then describe the dedicated key distribution mechanism that allows multiple layer encryption.

4.1 The Pohlig-Hellman cryptosystem & Key distribution

The Pohlig-Hellman cryptosystem [16] is defined as a tuple $(p, \mathscr{K}, \mathscr{E}, \mathscr{D})$ as follows:

- p is a large prime known by all nodes (it is a system parameter)
- $\mathscr{K}$ outputs a pair of keys (k_i, d_i) such that $k_i d_i \equiv (1 mod (p-1))$;
- $\mathscr{E}(p, k_i, x)$ returns $x^{k_i} mod p$;
- $\mathscr{D}(p, d_i, y)$ returns $y^{d_i} mod p$

Since $k_i d_i \equiv (1 mod (p-1))$, we have $x^{k_i d_i} mod n \equiv x mod n$.
The encryption operation is based on an exponentiation and is therefore inherently commutative. Indeed:

$$\begin{aligned}\mathscr{E}(p, k_i, \mathscr{E}(p, k_j, x)) &= (((x^{k_i}) mod p)^{k_j}) mod p \\ &= (x^{k_j k_i}) mod p \\ &= \mathscr{E}(p, k_j, \mathscr{E}(p, k_i, x))\end{aligned}$$

Thanks to the commutative property of the Pohlig-Hellman cryptosystem, any broker is able to add and suppress encryption layers if it stores the corresponding keys. The addition and subtraction of a layer in this new hybrid model respectively correspond to a Pohlig-Hellman encryption and decryption operation. Since the security of this cryptosystem relies on the hardness of the Discrete Logarithm Problem the key k_i can be used to encrypt several different messages. Moreover, this cryptosystem is asymmetric in the sense that the encryption key differs from the decryption key. However, as opposed to classical asymmetric cryptosystems such as RSA [18], if a node knows one of the keys, it can automatically deduce the remaining key. Therefore there is no "public key"; all keys are secret and they are only revealed to authorized nodes. We therefore need to define a proper key distribution mechanism that is processed in a self-organized way. In this new mechanism, each node N_i simply receives two pair of Pohlig-Hellman keys $(k_{i,j}, d_{i,j})$ and $(k_{m,j}, d_{m,j})$ which it repectively shares with its 2-level ancestors: $N_j = Parent(N_i, 1)$ and $N_m = Parent(N_i, 2)$.

The Pohlig-Hellman cryptosystem and the dedicated key distribution being described, we formally define the four security primitives in the next sections.

4.2 Propagation of subscription filters and building of routing tables

4.2.1 ENCRYPT_FILTER

$ENCRYPT\_FILTER$ used by subscriber S_i only requires the filter and two encryption keys k_{S_i,B_j} and k_{S_i,B_l} where B_j and B_l are respectively S_i's parent node and grandparent node.
Therefore:

$$\begin{aligned}ENCRYPT\_FILTER(f, k_{S_i,B_j}, k_{S_i,B_l}) = SF_{S_1} &= \mathscr{E}(p, k_{S_i,B_j}, \mathscr{E}(p, k_{S_i,B_l}, f)) \\ &= f^{k_{S_i,B_j} k_{S_i,B_l}} mod p\end{aligned}$$

Given this result, S_i sends the message $[SF_{S_i} = f^{k_{S_i,B_j} k_{S_i,B_l}} mod p; S_i]$ to its parent node B_j.

4.2.2 SECURE_TABLE_BUILDING

Whenever an intermediate node B_j receives an encrypted filter $f^{k_{B_i,B_j}k_{B_i,B_l}}modp$, it first removes one encryption layer with the use of d_{B_i,B_j} as follows:

$$\mathscr{D}(p,d_{B_i,B_j},f^{k_{B_i,B_j}k_{B_i,B_l}}modp) = f^{k_{B_iB_l}}modp$$

The $check-equivalence$ operation is very simple, since B_j only checks in its routing table RT_j if there is an equality with some rows. There is no need for an additional information to use this operation. If there is an equality, then the destination in SF_{B_i} is added in the corresponding row and the message is not forwarded; otherwise, B_j creates a new row with SF_{B_i}, adds another encryption layer with the key k_{B_j,B_m} shared with its grand-parent node B_m and finally sends the following message to B_l: $f^{k_{B_i,B_l}k_{B_j,B_m}}$.

4.3 Content distribution and secure look-up

Symmetrically, the Publisher P first uses the $ENCRYPT\_NOTIFICATION$ to encrypt the event notification with the corresponding keys and forwards the packet to the next broker. Then, the broker, after removing one encryption layer, runs the $SECURE\_LOOKUP$ primitive and accordingly it adds another encryption layer and forwards the packet.

4.3.1 ENCRYPT_NOTIFICATION

$ENCRYPT\_NOTIFICATION$ that is used by a publisher P, takes the routable attribute ra in the event notification and two keys $k_{B_m,P}$ and $k_{B_l,P}$ that are respectively shared with its child node B_m and its grandchild nodeB_l. Moreover, P defines a payload encryption key k_P in order to encrypt the payload with a symmetric encryption algorithm. $ENCRYPT\_NOTIFICATION$ returns:

$$EN = ENCRYPT\_NOTIFICATION(ra,k_{B_m,P},k_{B_l,P},k_P,P) = [EN_1;EN_2;EN_3]$$

$$EN_1 = ra^{k_{B_m,P}k_{B_l,P}}modp; EN_2 = k_P^{k_{B_m,P}k_{B_l,P}}modp; EN_3 = \mathscr{F}_{k_P}(\mathscr{P})]$$

4.3.2 SECURE_LOOK_UP

When an intermediate node B_m receives the encrypted event notification $ra^{k_{B_m,P}k_{B_l,P}}modp$ and the encryption payload encryption key $k_P^{k_{B_m,P}k_{B_l,P}}modp$, it will first suppress a decryption layer with the use of $d_{B_m,P}$ in order to obtain $EN_{1m} = ra^{k_{B_l,P}}modp$ and $k_P^{k_{B_l,P}}modp$. Given this partially decrypted routable attribute and the routing table RT_m, $SECURE\_LOOKUP(EN_{1m},RT_m)$ returns the list of children nodes where the corresponding packet will be forwarded. The look-up in this case simply consist in an equality check between EN_{1m} and each of the rows of RT_m. Then, B_m adds a new encryption layer and forward the following packet to the correct destination:

$$[ra^{k_{B_l,P}k_{B_m,B_j}}modp; k_P^{k_{B_l,P}k_{B_m,B_j}}modp; \mathscr{F}_{k_P}(\mathscr{P})]$$

Only the encryption key k_P of the payload is modified at each node. The payload itself is never modified while being forwarded. The protocol is illustrated with an example in appendix for a better understanding.

5 Analysis

In this section, we evaluate the security and the performance of the scheme.

We first show that the proposed encryption mechanism with multiple encryption layers ensures confidentiality against external attackers that do not participate to any networking or security operation and further show that it is reaching its privacy goal.

In a work evaluating the security of cryptosystems in the multi-user setting [3], Bellare et al. have essentially shown that if a cryptosystem is secure in the sense of indistinguishability, then the cryptosystem in the multi-user setting, where related messages are encrypted using different keys, is also secure. When a message is encrypted with two independent keys it is at least as secure as any individual encryption. Thus, the scheme is at least as secure as a one layer encryption.

Furthermore, thanks to the use of multiple encryption layers, the confidentiality of messages relies on the use of keys belonging to different users. Messages are namely forwarded and continuously modified by the addition and removal of encryption layers but they remain unaccessible to brokers or eavesdroppers at all times. Even if two subscribers are subscribing with the same filter they are not able to tell so because each one encrypts it with different keys.

Our protocol hence preserves privacy thanks to secure and efficient routing, moreover it provides the following features:

1. The security of the Pohlig-Hellman cryptosystem is based on the discrete logarithm problem in a finite field of prime order which is hard when the exponent is unknown. Hence we can use the same key several times and key management is very simple and local.
2. The secure aggregation operation is very simple as well since it is a simple equality test between two filters. Thanks to this secure aggregation of filters at each broker, routing tables are optimized.
3. Since there is no need for a shared secret, any node can be a subscriber. Brokers for instance can be subscribers and the privacy of all subscribers is still preserved which is a very interesting feature especially in a peer-to-peer environment.

Our protocol relies on the use of two encryption layers in order to simplify the description. In the honest-but-curious model this is optimal. However if two consecutive nodes, a node and its parent, collude and hence share their own keying material, they can decrypt their children nodes' subscriptions. Our scheme allows for a protection against collusion attack -which can occur only in a malicious broker model- by increasing the number of encryption layers as described in [12]. Therefore, the privacy of the scheme and its resistance to collusion attacks depends on the choice of the number of encryption layers denoted by r. The larger values for r imply a larger number of nodes to collude to break it. However, with large r, key storage per node becomes a burden and the key distribution overhead can have an impact on the performance of the protocol. Also aggregation occurs only after r hops so the larger the r the less efficient the aggregation mechanism. The choice of r is hence a trade-off that depends on the scenario and the topology of the network.

6 Related work

Publish subscribe is a messaging paradigm that allows the creation of flexible and scalable distributed systems. SIENA ([6]) is an example of a popular content-based publish subscribe system, but many others have been developed ([4, 2, 8]). Most of the efforts in this area concern pure networking issues, like performance or scalability.

Wang et al. [21] analyze the security issues and requirements that arise in CBPS systems. They mainly identify classical security problems (like authentication, integrity or confidentiality) and adapt them to the CBPS case. Yet they do not provide concrete or specific solutions to these new problems.

In [13] Opyrchal and Prakash focus on the confidentiality issue only on the last leg from end-point brokers to subscribers in a way that is more efficient than group security in terms of key management. Yet their scheme assumes that brokers are completely trustworthy.

Recently two interesting works concerning confidentiality in CBPS have been published. First, in [17], authors focus on notification and subscription confidentiality only. They define the confidentiality issues in a formal model and propose then few solutions depending on the subscription and notification format. Yet the proposed matching techniques are quite costly, since even the basic one featuring equality tests only is six times slower than the protocol without confidentiality. Moreover they assume that publishers and subscribers share a secret which reduces the decoupling of CBPS. Furthermore, in their attacker model, only the brokers are honest-but-curious, the publishers and subscribers are assumed to be trustworthy. This assumption is very strong because the group of publishers and subscribers may be very large. Such a scheme does not protect subscribers' privacy against other curious subscribers for example, let alone against malicious subscribers. Second, in [20], authors propose a specific key management scheme and then a probabilistic multi-path event routing to prevent frequency inferring attacks. In their threat model all nodes (publishers, subscribers and brokers) are assumed to be honest-but-curious. The main weakness of the scheme is the requirement for a KDC which is a centralized authority that is trusted not to be curious and decipher all the communication messages. Concerning content-based event routing, this scheme considers that events have some routable attributes which are tokenized in order to become pseudorandom chains and prevent dictionary attack. Like in [17] they adapt the protocol of Song et al. [19] but they do not motivate the use of this particular solution rather than easier and lighter ones. Furthermore their way of ensuring privacy is through multiple path routing thus affecting the performance, whereas we protect privacy by cryptographic means.

Finally, in [14], Opyrchal et al. deal with privacy in CBPS, but the focus of the paper is mainly on privacy policy management and not on the design of a cryptographical protocol to achieve it.

Multiple encryption was previously proposed in [10] where authors propose onion routing, to limit a network's vulnerability to traffic analysis. It provides anonymous communication for HTTP through proxies using the RSA commutative encryption scheme. Independently, Pannetrat and Molva use multiple layer encryption in [15] for the distribution of confidential data from 1 source to a group of n nodes. This particular algorithm ensures multicast confidentiality and it also prevents the compromise of the whole group whenever a subset of nodes are compromised. In [12], authors proposed a similar approach for data collection in wireless sensor networks where in this case, n nodes are sending some data to 1 source, the sink. In addition to confidentiality, authors also take the advantage of the inherent homomorphic property in the underlying encryption technique in order to ensure aggregation over encrypted data. Our scheme combines both of these approaches to ensure secure routing and hence subscriber privacy in the $n-to-n$ model akin to CBPS.

Private matching: the underpinning of the secure look-up and secure table building primitives is a matching operation using encrypted data. Private matching has been introduced for equality matches [1, 11] and extended to more general settings [9, 7]. Yet a careful study of the problem shows that there is a subtle but important difference between private matching and the requirements of our scheme. Private matching is

indeed a two-party protocol between a client and a server where the client learns at the end the information that he shares with the server, whereas in our case the matching operation has to be performed by a third party which has no control over the data.

7 Conclusion

In this paper, we analyzed privacy issues in content-based publish/subscribe networks. In order to solve this problem with cryptographic tools we analyzed the link between privacy and confidentiality and identified two necessary confidentiality requirements, namely publisher and information confidentiality. This led us to the more general problem of routing encrypted events using encrypted subscription filters. This problem of secure routing requires two main primitives, namely **building of encrypted routing tables** with aggregation of encrypted filters and **secure look-up** of encrypted events with encrypted routing tables to disseminate the events efficiently. These two primitives have to be designed together with the other classical primitives in order to solve the privacy-preserving routing which had no existing solution.

We then presented a solution to this problem based on multiple layer commutative encryption. MLCE allows brokers to perform secure transformations without having access to the data that is being transferred. Brokers can indeed remove or add an encryption layer without destroying the others and hence perform aggregation, routing tables building or look-up on private data protected by the other layers. Privacy is thus guaranteed among all nodes, including subscribers and eavesdropping outsiders.

Our solution uses the Pohlig-Hellman cryptosystem, and is the first scheme which enables privacy-preserving routing with no shared secret between end-users. Hence, key management is easy and local. Another key feature of this protocol is that it allows brokers to be subscribers at the same time while preserving privacy of all nodes which is appealing for peer-to-peer applications. This protocol can also be tailored to withstand collusion attacks at a certain performance cost.

As future work, we intend to develop these schemes by improving their flexibility regarding the network topology and the subscription filter format. We would like indeed to extend subscription filters encompass logical expressions.

Acknowledgments

The authors would like to thank Erik-Oliver Blaß for fruitfull discussions and comments.

References

1. R. Agrawal, A. V. Evfimievski, and R. Srikant. Information sharing across private databases. In A. Y. Halevy, Z. G. Ives, and A. Doan, editors, *SIGMOD Conference*, pages 86–97. ACM, 2003.
2. G. Banavar, T. Chandra, B. Mukherjee, and Nagarajarao. An efficient multicast protocol for content-based publish-subscribe systems. *Proceedings of 19th IEEE International Conference on Distributed Computing Systems*, pages 262–272, 1999.
3. M. Bellare, A. Boldyreva, and S. Micali. Public-key encryption in a multiuser setting: Security proofs and improvements. In *Eurocrypt 2000*, pages 259–274. Springer Verlag, 2000.
4. K. P. Birman. The process group approach to reliable distributed computing. *Commun. ACM*, 36(12):37–53, 1993.
5. D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano. Public key encryption with keyword search. In *EUROCRYPT*, pages 506–522, 2004.

6. A. Carzaniga, D. S. Rosenblum, and A. L. Wolf. Design and evaluation of a wide-area event notification service. *ACM Trans. Comput. Syst.*, 19(3):332–383, 2001.
7. L. Chmielewski and J.-H. Hoepman. Fuzzy private matching (extended abstract). In *ARES*, pages 327–334. IEEE Computer Society, 2008.
8. A. K. Datta, M. Gradinariu, M. Raynal, and G. Simon. Anonymous publish/subscribe in p2p networks. In *IPDPS '03: Proceedings of the 17th International Symposium on Parallel and Distributed Processing*, Washington, DC, USA, 2003. IEEE Computer Society.
9. M. J. Freedman, K. Nissim, and B. Pinkas. Efficient private matching and set intersection. In *EUROCRYPT*, pages 1–19, 2004.
10. D. M. Goldschlag, M. G. Reed, and P. F. Syverson. Hiding routing information. In *in Information Hiding*, pages 137–150. Springer-Verlag, 1996.
11. Y. Li, J. Tygar, and J. M. Hellerstein. Private matching. technical report IRB-TR-04-005, February 2004.
12. M. Önen and R. Molva. Secure data aggregation with multiple encryption. In *Wireless Sensor Networks, 4th European Conference, EWSN 2007*, Lecture Notes in Computer Science, pages 117–132, Delft, The Netherlands, 29-31 January 2007. Springer.
13. L. Opyrchal and A. Prakash. Secure distribution of events in content-based publish subscribe systems. In *SSYM'01: Proceedings of the 10th conference on USENIX Security Symposium*, pages 21–21, Berkeley, CA, USA, 2001. USENIX Association.
14. L. Opyrchal, A. Prakash, and A. Agrawal. Supporting privacy policies in a publish-subscribe substrate for pervasive environments. *JNW*, pages 17–26, 2007.
15. A. Pannetrat and R. Molva. Multiple layer encryption for multicast groups. In *The proceedings of CMS'02*, Portoroz, Slovenia, September 2002.
16. S. Pohlig and M. Hellman. An improved algorithm for computing logarithms over gf(p) and its cryptographic significance. *IEEE Transactions on Information Theory*, 24(1):106–110, Jan 1978.
17. C. Raiciu and D. S. Rosenblum. Enabling confidentiality in content-based publish/subscribe infrastructures. *Securecomm and Workshops, 2006*, pages 1–11, 28 2006-Sept. 1 2006.
18. R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21:120–126, 1978.
19. D. X. Song, D. Wagner, and A. Perrig. Practical techniques for searches on encrypted data. *Proceedings of the IEEE Symposium on Security and Privacy*, pages 44–55, 2000.
20. M. Srivatsa and L. Liu. Secure event dissemination in publish-subscribe networks. In *ICDCS '07: Proceedings of the 27th International Conference on Distributed Computing Systems*, page 22, Washington, DC, USA, 2007. IEEE Computer Society.
21. C. Wang, A. Carzaniga, D. Evans, and A. Wolf. Security issues and requirements for internet-scale publish-subscribe systems. In *HICSS '02: Proceedings of the 35th Annual Hawaii International Conference on System Sciences (HICSS'02)-Volume 9*, page 303, Washington, DC, USA, 2002. IEEE Computer Society.

Appendix: An example

In order to illustrate our solution we define a simple CBPS network with one publisher (P), four brokers (B_1 to B_4) and five subscribers (S_1 to S_5). This tree is presented in figure 1. We assume that S_1 to S_4 subscribe to a common filter f while S_5 subscribes to a different filter f'.

Key management is quite simple. For example, B_1 shares four pairs of keys $(k_{S_1B_1}, d_{S_1B_1})$, $(k_{S_2B_1}, d_{S_2B_1})$,$(k_{B_1B_3}, d_{B_1B_3})$ and $(k_{B_1B_4}, d_{B_1B_4})$ respectively with S_1, S_2, B_3 and B_4.

The subscribers first encrypt their filters and send them upwards. For example S_2 sends to B_1 the following: $[f^{k_{S_1B_1}k_{S_1B_3}} mod p; S_1]$.

When subscription filters are propagated brokers build their routing table. We take the example of broker B_3. In this solution, aggregation is not performed directly but after two hops. Hence, B_3 receives encrypted subscription filters from all subscribers. Let us assume it first receives the message $[f^{k_{S_1B_3}k_{B_1B_4}} mod p, S_1]$. B_3 removes the encryption layer with S_1 and then stores in its forwarding table $f^{k_{B_1B_4}} \rightarrow B_1(S_1)$. It also forwards $[f^{k_{B_1B_4}k_{B_3P}} mod p; B_1]$ to B_4. Then if B_3 receives $[f^{k_{S_2B_3}k_{B_1B_4}} mod p, S_2]$ it removes the layer with S_2

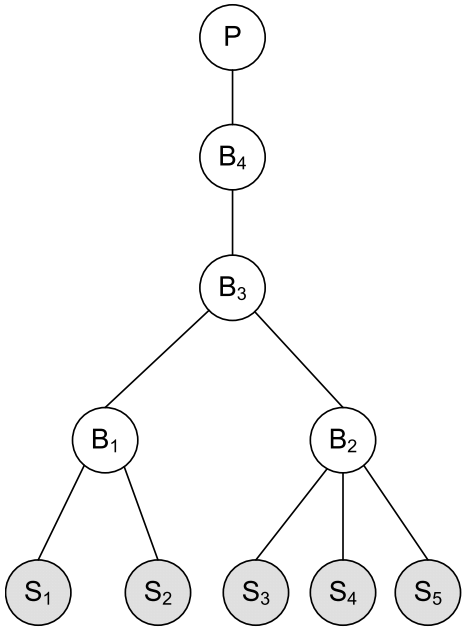

Fig. 1 Network used as illustration

to get $f^{k_{B_1B_4}}$ which is equal to the first row of RT_3, hence it just updates the list of destination and does not forward the message. When it receives $[f^{k_{S_3B_3}k_{B_2B_4}} mod\, p, S_3]$ it also removes one layer (with S_3) to get $f^{k_{B_2B_4}}$ which is different from the entry in its forwarding table and hence is not aggregated. A new row is added in FT_3 and the message $[f^{k_{B_2B_4}k_{B_3P}} mod\, p; B_2]$ is forwarded to B_4. B_3 also receives messages from S_4 and S_5 and *in fine* its forwarding table is represented in Table 2.

R_{3-1}	$f^{k_{B_1B_4}} \rightarrow B_1(S_1), B_1(S_2)$
R_{3-2}	$f^{k_{B_2B_4}} \rightarrow B_2(S_3), B_2(S_4)$
R_{3-3}	$f'^{k_{B_2B_4}} \rightarrow B_2(S_5)$

Table 2 Routing table RT_3 of B_3

In order to illustrate the propagation of a filter until the last hop before the publisher, we use the same example as in the previous scheme where S_1 wishes to send a filter f. Table 3 illustrates this propagation.

When P wants to publish a content $\mathscr{P}$ with routable attribute f, it chooses a key k_P and then performs $ENCRYPT\_NOTIFICATION(f, k_{B_3P}, k_{B_4P}, k_P, \mathscr{P})$ to get:

$$[f^{k_{B_3P}k_{B_4P}} mod\, p; k_P^{k_{B_3P}k_{B_4P}} mod\, p; \mathscr{F}_{k_P}(\mathscr{P})].$$

This message is then forwarded to its child B_4.

The content is forwarded by brokers according to their routing tables. For example when B_3 receives from B_4 the message $[f^{k_{B_3P}k_{B_1B_4}} mod\, p; k_P^{k_{B_3P}k_{B_1B_4}} mod\, p; \mathscr{F}_{k_P}(\mathscr{P})]$, B_3 removes the encryption layer with P in

S_1	f
$S_1 \rightarrow B_1$	$[f^{k_{S_1B_1}k_{S_1B_3}} mod p; S_1]$
B_1	$f^{k_{S_1.B_3}} mod p$
$B_1 \rightarrow B_3$	$[f^{k_{S_1B_3}k_{B_1B_4}} mod p; S_1]$
B_3	$f^{k_{B_1.B_4}} mod p$
$B_3 \rightarrow B_4$	$[f^{k_{B_1B_4}k_{B_3P}mod p}; B_1]$
B_4	$f^{k_{B_3,P}} mod p$

Table 3 Propagation of a filter f from S_1 to B_4

the routable attribute to get $f^{k_{B_1B_4}} mod p$ and then perform the look-up in RT_3. This look-up shows that this content corresponds to the first row of RT_3 and that two grandchildren are interested in it. Hence B_3 sends two messages to B_1, namely:

$$[f^{k_{B_1B_4}k_{S_1B_3}} mod p; k_P^{k_{B_1B_4P}k_{S_1B_3}} mod p; \mathscr{F}_{k_P}(\mathscr{P})],$$

$$\text{and } [f^{k_{B_1B_4}k_{S_2B_3}} mod p; k_P^{k_{B_1B_4P}k_{S_2B_3}} mod p; \mathscr{F}_{k_P}(\mathscr{P})]$$

Table 4 illustrates the propagation of an event notification related to filter f from publisher P to subscriber S_1.

Step	Event notification
P	$[f, k_P, P]$
$P \rightarrow B_4$	$[f^{k_{B_3P}k_{B_4P}} mod p; k_P^{k_{B_3P}k_{B_4P}} mod p; \mathscr{F}_{k_P}(\mathscr{P})]$
B_4	$[f^{k_{B_3P}} mod p, k_P^{k_{B_3P}} mod p, \mathscr{F}_{k_P}(\mathscr{P})]$
$B_4 \rightarrow B_3$	$[f^{k_{B_3P}k_{B_1B_4}} mod p, k_P^{k_{B_3P}k_{B_1B_4}} mod p, \mathscr{F}_{k_P}(\mathscr{P})]$
B_3	$[f^{k_{B_1B_4}} mod p; k_P^{k_{B_1B_4}} mod p, \mathscr{F}_{k_P}(\mathscr{P})]$
$B_3 \rightarrow B_1$	$[f^{k_{B_1B_4}k_{S_1B_3}} mod p, k_P^{k_{B1.B4}k_{S_1B_3}} mod p, \mathscr{F}_{k_P}(\mathscr{P})]$
B_1	$[f^{k_{S_1B_3}} mod p, k_P^{k_{S_1B_3}} mod p, \mathscr{F}_{k_P}(\mathscr{P})]$
$B_1 \rightarrow S_1$	$[f^{k_{S_1B_3}k_{S_1B_1}} mod p, k_P^{k_{S_1B_3}k_{S_1B_1}} mod p, \mathscr{F}_{k_P}(\mathscr{P})]$
S_1	$[f, k_P, P]$

Table 4 Evolution of a message published by P on its path to S_1