A Privacy-Enhancing Content-Based Publish/Subscribe System Using Scalar Product Preserving Transformations

Sunoh Choi[1], Gabriel Ghinita[2], and Elisa Bertino[2]

[1] Department of Electrical and Computer Engineering, Purdue University, USA
choi39@purdue.edu
[2] Department of Computer Science, Purdue University, USA
{gghinita,bertino}@cs.purdue.edu

Abstract. Users of content-based publish/subscribe systems (CBPS) are interested in receiving data items with values that satisfy certain conditions. Each user submits a list of subscription specifications to a broker, which routes data items from publishers to users. When a broker receives a notification that contains a value from a publisher, it forwards it only to the subscribers whose requests match the value. However, in many applications, the data published are confidential, and their contents must not be revealed to brokers. Furthermore, a user's subscription may contain sensitive information that must be protected from brokers. Therefore, a difficult challenge arises: how to route publisher data to the appropriate subscribers without the intermediate brokers learning the plain text values of the notifications and subscriptions. To that extent, brokers must be able to perform operations on top of the encrypted contents of subscriptions and notifications. Such operations may be as simple as equality match, but often require more complex operations such as determining inclusion of data in a value interval. Previous work attempted to solve this problem by using one-way data mappings or specialized encryption functions that allow evaluation of conditions on ciphertexts. However, such operations are computationally expensive, and the resulting CBPS lack scalability. As fast dissemination is an important requirement in many applications, we focus on a new data transformation method called Asymmetric Scalar-product Preserving Encryption (ASPE) [1]. We devise methods that build upon ASPE to support private evaluation of several types of conditions. We also suggest techniques for secure aggregation of notifications, supporting functions such as sum, minimum, maximum and count. Our experimental evaluation shows that ASPE-based CBPS incurs 65% less overhead for exact-match filtering and 50% less overhead for range filtering compared to the state-of-the-art.

Keywords: Publish/Subscribe Systems, Privacy, Confidentiality, Security.

1 Introduction

In Content-based Publish/Subscribe Systems (CBPS), data sources (or publishers) disseminate contents to users (or subscribers) using an infrastructure of intermediate

P. García Bringas et al. (Eds.): DEXA 2010, Part I, LNCS 6261, pp. 368–384, 2010.

routing entities called *brokers*. Each user creates a subscription that specifies constraints on the data items that s/he is interested to receive. Users register subscriptions with brokers. When a broker receives a notification from a data source, it examines which subscriptions satisfy the notification, and routes the data item to the appropriate users.

A typical application of CBPS is the dissemination of stock market quotes. Several large stock market exchanges act as data publishers, and generate high-rate data streams with prices of individual stocks. Each stock price update, e.g., "*Google, $600*" represents a notification. Users interested in an individual stock may formulate subscriptions with conditions such as "*Google,* $\geq$ *$550*", stating that only updates on the stock price of Google that are above $550 should be sent to the user. Figure 1 illustrates this scenario. When a broker receives a notification, it should be able to determine which subscriptions satisfy the notification and forward it accordingly. In order to do so, a broker has to support various evaluation functions, such as exact match (or equality) filtering, inequality filtering (e.g., "$>$" or "$\leq$"), range filtering, etc.

On the other hand, it is not always feasible to deploy a trusted, secure broker infrastructure dedicated to each CBPS application. Instead, an existing content distribution network such as Akamai [6], or an ad-hoc P2P/Grid computing environment may be used, in order to reduce operation and maintenance costs. In this scenario, brokers can no longer be trusted. Therefore, the contents of subscriptions and notifications should no longer be revealed in plaintext to the brokers.

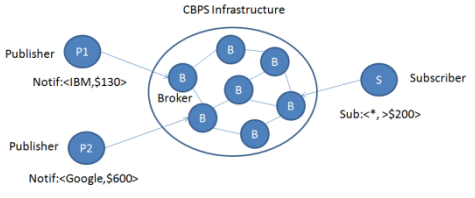

Fig. 1. Content-Based Publish/Subscribe System

A non-trusted broker that has access to plaintext subscriptions and notifications may cause significant damage to the CBPS participants. For instance, stock quotes dissemination is typically an expensive service to maintain, and users must pay a subscription fee. If the brokers do not route the data properly, unauthorized users may gain data access, causing financial losses to the publishers. On the other hand, subscription data may also be sensitive. Brokers may collect data subscriptions and infer certain sensitive information: for instance, untrustworthy brokers who notice that a large number of users are interested in a given stock at a particular price may use this knowledge to place unfair market orders at the disadvantage of other participants. In other application areas, subscription contents may disclose private details about users, such as shopping habits, political or religious affiliations, etc.

The conventional method for preventing unauthorized disclosures in a context like CBPS is to apply encryption on the plaintext data and to allow only authorized parties

to perform decryption. Unauthorized parties, such as the provider of the routing infrastructure, are unable to gain access to the plaintext data even though they have access to the ciphertext. However, encryption greatly complicates the process of data filtering. A naïve solution, based on encryption, would require broadcasting all the encrypted data from publishers to all subscribers. But, such a solution is not scalable, as the amount of transferred data is huge, and would quickly congest communication networks. Moreover, a user must have access only to the part of the data s/he has subscribed for. To this end, data items must be encrypted with different keys, and an alternate channel used to send appropriate encryption keys to each user is required. This way, each subscriber can only decrypt the data items s/he is authorized to. Still, managing such a large number of keys may also pose serious scalability concerns.

In order to achieve notification and subscription confidentiality in CBPS, a broker should be allowed to access only encrypted data. Several techniques that address this problem have been proposed previously. For instance, the method in [5] uses a partitioning method and builds an index of encrypted subscriptions. Certain types of conditions can be evaluated on values encrypted in this fashion. However, the method incurs false positives. The solution in [14] uses properties similar to those of homomorphic encryption [21] to evaluate conditions on encrypted data. However, the cost incurred may be large.

In this paper we propose a novel approach to encryption-based access control for CBPS. Our methods support efficient and precise evaluation of conditions based on the ciphertexts of subscriptions and notifications. Our solution relies on Asymmetric Scalar-Product Preserving Encryption (ASPE) [1], a geometric transformation that supports comparisons between pairs of data items. ASPE has been proposed in [1] in the context of evaluating nearest-neighbor queries. We adapt ASPE to support a broad range of conditions evaluations, such as exact match, inequality, range, as well as conjunction of single-attribute conditions. Our specific contributions are:

(i) We propose a novel secure CBPS method for condition evaluations using encrypted values. Our solution has a reduced computational overhead and does not incur false positives.

(ii) We outline a mechanism for secure CBPS aggregation under functions such as sum, minimum, maximum and count.

(iii) We developed a prototype of the proposed method on top of the well-established CBPS system SIENA [4], and we performed an extensive experimental evaluation in comparison with the state-of-the–art in secure CBPS [5]. The experimental results show that our proposed method clearly outperforms existing work in terms of overhead, while offering similar security features.

The remainder of the paper is organized as follows: Section 2 presents background information and surveys related work. Section 3 presents our solution for secure CBPS, whereas Section 4 discusses the case of attacker collusion. In Section 5, we outline a method for secure aggregation in CBPS. Section 6 presents experimental results, and Section 7 concludes with directions for future work.

2 Background

2.1 CBPS Overview

The main characteristic of CBPS is the loose coupling between publishers and subscribers, which allows for high scalability. The participants in a CBPS system are publishers (or data producers), subscribers (or data consumers), and an infrastructure of brokers that route the data from publishers to subscribers.

Publishers continuously generate data items and forward them to data brokers. A data item, also called a notification, is a set of attributes, where an attribute is a triple: (type, name, value) [4]. After the publisher sends a notification to a broker, it is no longer concerned with the process through which notifications reach subscribers.

A subscriber has an ability to express its interest in a particular data item by generating a subscription. Sometimes subscriptions are also referred to as filters. A subscription is a set of constraints, where a constraint is as follows: (type, name, operator1, value1, operator2, value2). The operators include all the common equality and ordering relations ($=, \neq, <, >$, etc). In order to express its interest, a subscriber creates a subscription and registers it with a broker. Subsequently, the subscriber does not need to be concerned about how messages will be routed.

Brokers route notification and subscription messages, and *match* notifications against subscriptions. An attribute $\alpha=(types_\alpha,name_\alpha,value_\alpha)$ matches an attribute constraint $\phi=(types_\phi,name_\phi,,operator1_\phi,value1_\phi,operator2_\phi,value2_\phi)$ iff. $types_\alpha=types_\phi \wedge name_\alpha=name_\phi \wedge operator1_\phi(value_\alpha,value1_\phi) \wedge operator2_\phi(value_\alpha,value2_\phi)$. We denote the fact that an attribute α matches an attribute constraint ϕ by $\alpha \prec \phi$. All such constraints must be matched. Then, we say that a notification n matches a filter f. ($n \prec f$ for short) if:

$$n \prec f \Leftrightarrow \forall \phi \in f: \ \exists \alpha \in n : \alpha \prec \phi$$

A subscription s_1 is said to *cover* another subscription s_2 when the set of notifications matched by s_2 is a subset of the set of notifications matched by s_1. Deciding whether a subscription covers another can reduce the amount of match operations performed by brokers, as well as the network traffic overhead.

$$s_2 \prec s_1 \Leftrightarrow \forall n: \ n \prec s_2 \implies n \prec s_1$$

There are several kinds of subscriptions. For instance, an equality filter specifies that an exact value must be matched (e.g., "price = 150"). The inequality filter requires a notification to have a value that is greater (conversely, less) than the value of the subscription (e.g., "amount < 300"). Similarly, a range filter specifies a value range (e.g., "100 < price < 200"). In addition, a subscriber may want to receive a notification that satisfies simultaneously several conditions on distinct attributes (e.g., "price < 100 and amount > 300"). We refer to such a subscription as conjunction filter.

We assume that brokers are curious but honest. In the worst case, a malicious broker could simply refuse to forward notifications, staging a denial-of-service attack. Protection against such threats can be achieved if several alternate paths between each publisher and subscriber pair are enforced in the routing infrastructure. However, this

Table 1. Examples of $\prec$

Notification	Match	Subscription
integer price = 150	$\prec$	*integer* price = 150
integer amount = 500	$\not\prec$	*integer* amount < 300
integer price = 150	$\prec$	*integer* price > 100 *integer* price < 200
integer price = 100 *integer* amount = 500	$\not\prec$	*integer* price < 100 *integer* amount > 300

is outside our scope: we focus on protecting against disclosure of notification and subscription information to brokers. The problem of malicious brokers has been addressed in previous work under some restricted scenarios [10, 12].

2.2 Related Work

Our research is related to previous work on outsourced databases (ODB) [8]. In ODB, a data owner (or publisher) stores a database at a service provider (SP) site. Users (i.e., data consumers) send their queries to the SP which processes queries and returns the results to the user. However, the SP is not trusted, therefore the data owner must upload an encrypted version of the database to the service provider. Processing queries on top of encrypted data poses similar challenges to our problem of matching notifications to subscriptions without accessing the plaintext versions of either. In addition, ODB are also concerned with ensuring the completeness (i.e., integrity) of query results. We do not address integrity; however, previous results that address integrity in data streams [20] can be adapted to our scenario.

The Order Preserving Encryption Scheme (OPES) applies an encryption function to an ordinal domain, such that $E(x) < E(y)$ if $x < y$ [2]. Thus, OPES converts a distribution of values to another distribution of values, possibly in a distinct domain. So, we can use OPES for inequality filtering and range filtering. However, OPES requires the distribution of the data in advance, whereas data are dynamically generated in CBPS.

On the other hand, we can encrypt a database by splitting the data domain into several partitions and assigning indices to these partitions [5, 7, 19]. In this case, since a service provider may return a superset of the query result, the data owner should perform a post-processing to filter false positives. This can be a problem in CBPS because a subscriber should not be able to receive data that s/he does not have authorization for. Such approaches violate the confidentiality of notifications. The work in [19] assumes that subscribers know the secure index. However, this can give users extra information which may result in compromising confidentiality.

The work in [15] discusses security requirements in CBPS. However, no specific methods for notification and subscription confidentiality are given. Similar to our work, in [16] the brokers are also not trusted. However, it is required that the part of notifications used for routing is not encrypted. Hence, confidentiality is compromised. The work in [18] assumes that the brokers are trusted and are allowed to decrypt a notification for routing. Such an assumption is not reasonable in our model.

The work of Raiciu et al. [5] provides notification and subscription confidentiality for equality, inequality, and range filtering. However, it incurs false positives in inequality filtering and range filtering. Moreover, in order to reduce the ratio of false positives, it has to use more partitions in a domain. Hence, it will take more time for a broker to match a notification against subscriptions. In addition, since a notification should have information about all partitions for range filtering, it will not be practical in a real application. The work in [14] achieves privacy-preserving filtering and covering in CBPS, but it relies on cryptographic elements that incur high computational overhead.

The work in [11] used an additively homomorphic public-key cryptosystem to protect confidential data from intermediate aggregation nodes. The work in [10] verified the integrity of the sum by leveraging a homomorphic MAC scheme based on the discrete logarithm property. However, only secure additive aggregation is considered. We consider in addition methods for sum, min, max and count functions for secure aggregation supporting equality, inequality and range filtering.

2.3 Asymmetric Scalar-Product Preserving Encryption (ASPE)

Distance-preserving-transformations (DPT) [13] are an appealing construction to hide the data while allowing at the same time to perform certain operations on top of the ciphertexts. DPT are a subset of the more general Distance Recoverable Encryption (DRE). However, as shown in [1], DRE-based techniques are vulnerable to attacks when the adversary knows some of the plaintext data, or the mapping between some plaintexts and ciphertexts [1, 9]. In our work, we employ a superior transformation, ASPE, which is not distance-recoverable [1].

The weakness of DRE comes from the fact that the attacker is able to recover distance information from the encrypted data [1]. In contrast, ASPE does not reveal distance information. Instead, it only provides means for distance comparison. Given two points p_1 and p_2, it decides which of the two points is nearer to a query point q.

$$D(p_1, q) \geq D(p_2, q)$$
$$\sqrt{\|p_1\|^2 - 2p_1 \cdot q + \|q\|^2} \geq \sqrt{\|p_2\|^2 - 2p_2 \cdot q + \|q\|^2}$$
$$\|p_1\|^2 - \|p_2\|^2 + 2(p_2 - p_1) \cdot q \geq 0$$

The inequality is decomposed to a number of scalar product computations. There are three types of scalar products: (type-1) scalar product of a data point with itself, (type-2) scalar product of a data point with the query point, and (type-3) scalar product of two different data points p_1 and p_2. If an encryption preserves only type-1 and type-2 products but not type-3 products, which is essential in DRE, then we can compare the distances $d(p_1, q)$ and $d(p_2, q)$ by the above equation without the vulnerabilities of DRE.

The scalar product of p and q can be represented as $p^T Iq$. I is the unit matrix, and can be decomposed to MM^{-1} for any invertible matrix M. If we set $p' = M^T p$ and $q' = M^{-1} q$, it is not computationally feasible for an adversary to determine the value of p and q from p' and q' without knowing M. Also, $p'^T q' = p^T M M^{-1} q = p^T q$, i.e., type-2 scalar product is preserved. But, $p_1'^T p_2' = p_1^T M M^T p_2$ is not equal to $p_1^T p_2$ in general. The Asymmetric Scalar-product Preserving Encryption (ASPE) uses M and M^{-1} as the transformations for data points and queries respectively.

If the type-1 product $\|p\|^2$ is revealed to the attacker, he knows that p lies on a hypersphere that is centered at the origin with a radius $\|p\|$. It can partially compromise security. We need to hide this information by hiding the value of $\|p\|^2$. That is, the value of $-0.5\|p_1\|^2$ is treated as the (d+1)-st dimension of the point p since it will be used to get the distance difference. Given a d-dimensional data point p, a (d+1)-dimensional point p^* is created. Similarly, we need to extend a query q to a (d+1)-dimensional point q^*. The simplest way is to set the (d+1)-st dimension of q^* to 1. For security, we generate a random number $r>0$ and scale q^* by r.

Let p_1', p_2', and q' be the encrypted points of the data points p_1, p_2 and the query point q. We can determine whether p_1 is closer to q than p_2 by evaluating whether $(p_1' - p_2') \cdot q' > 0$. Note that

$$\begin{aligned}(p_1' - p_2') \cdot q' &= (p_1' - p_2')^T q' = (M^T p_1^* - M^T p_2^*)^T M^{-1} q^* = (p_1^* - p_2^*)^T q^* \\ &= (p_1 - p_2)^T (rq) + (-0.5\|p_1\|^2 + 0.5\|p_2\|^2) r \\ &= 0.5r(\|p_2\|^2 - \|p_1\|^2 + 2(p_1 - p_2)^T q) \\ &= 0.5r(D(p_2, q) - D(p_1, q))\end{aligned}$$

So, the condition is equivalent to

$$0.5r(D(p_2, q) - D(p_1, q)) > 0 \Leftrightarrow D(p_2, q) > D(p_1, q).$$

3 Secure CBPS Using ASPE

By using ASPE, we can compare the distance between a data point p_1 and a query point q with the distance between another data point p_2 and the same query point q. In this paper, we propose a CBPS framework for matching subscriptions and notifications encrypted with an ASPE-like technique. In the CBPS setting, the data point p corresponds to a notification, and the query point q to a subscription. Specifically, for each subscription the users generate a set of reference values encrypted according to transformation M, whereas the publishers transform notifications according to transformation M^{-1}.

There are several kinds of subscription conditions in CBPS. 1) In the equality filtering, a subscriber wants to receive a notification which is equal in value to its subscription. 2) In the inequality filtering, a subscriber wants to receive a notification which is larger or less than its subscription. 3) In the range filtering, a subscriber wants to get a notification which is in a certain range. 4) In the conjunction filtering, a subscription consists of several conditions expressed as multiple attribute constraints and a subscriber wants to receive a notification which satisfies all the conditions.

In addition, the *covering* operation allows a broker to determine if, given two subscriptions, one of them will always match a strict subset of the notifications matched by the other. Next, we show how ASPE can be used to support all the above operations while maintaining confidentiality of subscriptions and notifications.

3.1 Equality Filtering

Consider for example that a subscriber wants to receive a notification which is equal to some value a. The subscriber sends to a broker the encrypted subscription $E(a) = a'$.

When a publisher wants to disseminate a data item with value x, it sends to a broker an encrypted notification $E(x) = x'$. When a broker receives a' and x', it must be able to determine whether x is equal to a. But, the broker should not learn what are the values of x and a.

Our secure matching method works as shown in Fig 2. The subscriber selects c and d where $c = a - s$ and $d = a + s$, i.e., a is the middle of the interval *[c,d]* ($s > 0$ is a random number) and sends $c' = M^T c^* = M^T(c, -0.5\|c\|^2)^T$ and $d' = M^T d^* = M^T(d, -0.5\|d\|^2)^T$ as a subscription to the broker. On the other hand, the publisher sends a notification $x' = M^{-1}x^* = M^{-1}r(x, 1)^T$.

As explained in Section 2.3, a broker can evaluate based on the encrypted values which one of *distance*(*c,x*) and *distance*(*d,x*) is larger.

$$\begin{aligned}(c' - d') \cdot x' &= (c' - d')^T x' = (M^T c^* - M^T d^*)^T M^{-1} x^* \\ &= (c^* - d^*)^T x^* \\ &= (c - d)^T (rx) + (-0.5\|c\|^2 + 0.5\|d\|^2) r \\ &= 0.5r(\|d\|^2 - \|c\|^2 + 2(c - d)^T x) \\ &= 0.5r(D(d, x) - D(c, x))\end{aligned} \tag{1}$$

If the difference is equal to *0*, the broker can determine that *x* is equal to *a*. Otherwise, the values must be different. However, since the broker receives $x', c',$ and d' instead of x and a, it can not know what x and a are.

Theorem 1. The equality matching using ASPE is secure.

Proof. Suppose that a broker wants to compute *a* using $x', c',$ and d' when *x* is equal to *a*. If the broker can get $(c + d)^T x = 2\|a\|^2$, since $c + d$ is equal to *2a*, the equality filtering is not secure. We show that the broker cannot obtain $(c + d)^T x$ from $(c' + d')^T x'$.

$$\begin{aligned}(c' + d')^T x' &= (M^T c^* + M^T d^*)^T M^{-1} x^* \\ &= (c^* + d^*)^T x^* \\ &= (c + d)^T (rx) + (-0.5\|c\|^2 - 0.5\|d\|^2) r \\ &= 0.5r(2(c + d)^T x - \|c\|^2 - \|d\|^2) \\ &\neq (\mathrm{c} + \mathrm{d})^{\mathrm{T}} \mathrm{x}\end{aligned} \tag{2}$$

Since the (*d+1*)-st dimensions of c^* and d^* are $-0.5\|c\|^2$ and $-0.5\|d\|^2$ respectively, and there is a random number r, the broker can't get $(c + d)^T x$ by computing $(c' + d')^T x'$ when *x* is equal to *a*. Therefore, the equality test using ASPE is secure.

3.2 Inequality Filtering

In the inequality filtering, the broker should be able to determine whether the notification value *x* is greater than subscription value *a*. As in the case of equality filtering using ASPE, the subscriber selects c and d where $c = a - s$ and $d = a + s$ ($s > 0$ is a random number) and sends $c' = M^T c^* = M^T(c, -0.5\|c\|^2)^T$ and $d' = M^T d^* = M^T(d, -0.5\|d\|^2)^T$as a subscription to the broker. The publisher sends a notification $x' = M^{-1}x^* = M^{-1}r(x, 1)^T$. At that time, the broker can compute a

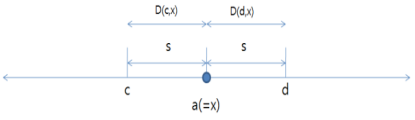

Fig. 2. Equality Filtering using ASPE

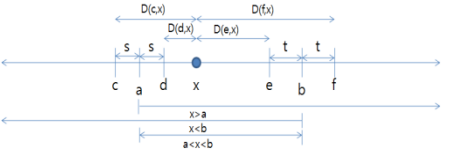

Fig. 3. Range Filtering using ASPE

difference between *distance*(*c,x*) and *distance*(*d,x*). If the difference is greater than *0*, the broker can know that *x* is greater than *a*. But, the broker can't know what the values of *x* and *a* are. The security proof is similar to the proof of Theorem 1.

$$D(c,x) - D(d,x) > 0 \Leftrightarrow (d^{'} - c^{'}) \cdot x^{'} > 0 \tag{3}$$

3.3 Range Filtering

When a subscriber wants to receive a value *x* between *a* and *b,* s/he sends to the broker two pairs of values. The first pair $c' = M^T c^* = M^T(c, -0.5\|c\|^2)^T$ and $d' = M^T d^* = M^T(d, -0.5\|d\|^2)^T$ encodes the value of a', whereas the second pair $e' = M^T e^* = M^T(e, -0.5\|e\|^2)^T$ and $f' = M^T f^* = M^T(f, -0.5\|f\|^2)^T$ encodes the value of b', where $e = b - t$ and $f = b + t$ ($t > 0$ is a random number), as shown in Figure 3. By using x', c', d', e', f', the broker can compute a difference *p* between *distance*(*c,x*) and *distance*(*d,x*) and a difference *q* between *distance*(*e,x*) and *distance*(*f,x*). If *p* is greater than *0*, *x* is greater than *a* and if *q* is less than *0*, *x* is less than *b*. Therefore, the broker can determine whether *x* is between *a* and *b* as follows:

$$\begin{aligned} &D(c,x) - D(d,x) > 0 \;\&\&\; D(e,x) - D(f,x) < 0 \\ &\Leftrightarrow (d' - c') \cdot x' > 0 \;\&\&\; (f' - e') \cdot x' < 0 \end{aligned} \tag{4}$$

In [2], it is suggested that the Order Preserving Encryption Scheme (OPES) can be used for inequality filtering and range filtering on encrypted data. However, OPES requires to know the data distribution in advance. But, in CBPS, since data are dynamically generated, it may be difficult to know the data distribution, especially when there can be several publishers. So, OPES is not suitable for CBPS. In [3], the ciphertext size of range filtering is $O(\sqrt{n})$. The value *n* is the number of data points in a domain. Such an approach would not scale well. Note that, the space complexity of range filtering using ASPE is $O(1)$.

3.4 Covering

Suppose that there are two subscribers. When the first subscriber wants to get a value greater than *a* and the second subscriber wants to receive a value greater than *b*, the broker should be able to determine that the set of notifications matched by one of the subscriptions will always be a subset of the notifications matched by the other. For instance, if *a* is less than *b*, the broker can send only (the encrypted) *a* to a parent broker. Also, when a parent broker evaluates the matching condition, it may only

need to evaluate a single condition rather than two, saving computation cost. Therefore, the covering operation helps to achieve scalability in CBPS.

Our idea is illustrated in Figure 4. Each subscriber selects a random number. Thus, the first subscriber chooses a random number y (y>>a+s=d) and the second subscriber picks up a random number z (z>>b+t=f). The first subscriber sends to the broker the triplet $c' = M^T c^* = M^T(c, -0.5\|c\|^2)^T$, $d' = M^T d^* = M^T(d, -0.5\|d\|^2)^T$, $y' = M^{-1}y^* = M^{-1}(y, 1)^T u$ (where u is also a random number) to encode the value of a' and the second subscriber sends $e' = M^T e^* = M^T(e, -0.5\|e\|^2)^T$, $f' = M^T f^* = M^T(f, -0.5\|f\|^2)^T$, $z' = M^{-1}z^* = M^{-1}(z, 1)^T v$ (where v is a random number) to encode the value of b'.

By computing the difference between *distance*(e,y) and *distance*(f,y), if the difference is greater than *0*, the broker can determine whether y is greater than b. If y is less than b, the broker can know that b is greater than a since y is greater than a+s(=d).

If y is greater than b and condition (5) is satisfied, the broker can determine that b is greater than a. However, the broker can't know what a and b are. On the other hand, in condition (5), we can use z' instead of y'.

$$\begin{aligned} D(a,y) > D(b,y) &\Leftrightarrow D(c,y) + D(d,y) > D(e,y) + D(f,y) \\ &\Leftrightarrow (D(c,y) - D(e,y)) + (D(d,y) - D(f,y)) > 0 \\ &\Leftrightarrow (e' - c') \cdot y' + (f' - d') \cdot y' > 0 \end{aligned} \tag{5}$$

For simplicity, the condition (5) and Figure 4 show covering for the inequality filtering. We can extend this easily to covering for range filtering. In this case, a subscriber sends c', d', i', j', y' in order to get the values between a and g and the other subscriber sends e', f', k', l', z' in order to get the values between b and h. If condition (6) is satisfied, we can say that the first subscription covers the second subscription. Thus, if x is between b and h, then x is between a and g (e.g., $b < x < h \rightarrow a < x < g$).

$$\begin{aligned} & D(a,y) > D(b,y) \;\&\&\; D(g,y) < D(h,y) \\ \Leftrightarrow\; & D(c,y) + D(d,y) > D(e,y) + D(f,y) \\ & \&\& \; D(i,y) + D(j,y) < D(k,y) + D(l,y) \\ \Leftrightarrow\; & (D(c,y) - D(e,y)) + (D(d,y) - D(f,y)) > 0 \\ & \&\& \big(D(i,y) - D(k,y)\big) + (D(j,y) - D(l,y)) > 0 \\ \Leftrightarrow\; & (e' - c') \cdot y' + (f' - d') \cdot y' > 0 \\ & \&\& \; (k' - i') \cdot y' + (l' - j') \cdot y' > 0 \end{aligned} \tag{6}$$

3.5 Conjunction Filtering

Suppose that a publisher sends a value m and a value n for two distinct attributes in a single notification. In this case, the publisher sends a column vector $X' = M^{-1}X = M^{-1}r(X, 1)^T$ to the broker(s). The column vector X contains the original values m

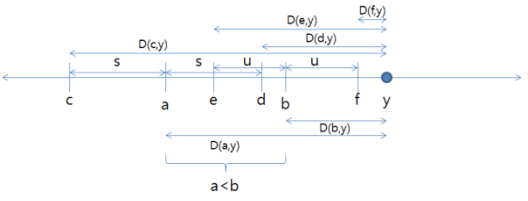

Fig. 4. Covering for Range Query using ASPE

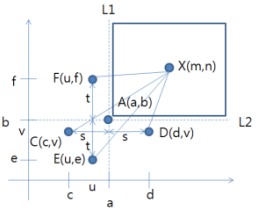

Fig. 5. Conjunctive Query using ASPE

and n. Suppose that a subscriber wants to receive a notification in which the value of the first attribute is greater than a and the value of the second attribute is greater than b. Then, the subscriber sends $C' = M^T C^* = M^T(C, -0.5\|C\|^2)^T, D' = M^T D^* = M^T(D, -0.5\|D\|^2)^T$, $E' = M^T E^* = M^T(E, -0.5\|E\|^2)^T$, $F' = M^T F^* = M^T(F, -0.5\|F\|^2)^T$as a subscription instead of $A' = M^T A^* = M^T(A, -0.5\|A\|^2)^T$. A is a column vector which contains the values a and b. As shown in Figure 5, C and D have the same distance from the line $L1$. E and F have the same distance from the line $L2$. If a condition (7) is satisfied, the broker can determine whether m is greater than a and n is greater than b. Thus, the broker can know that X is inside the rectangle specified by coordinates $L1$ and $L2$. But, the broker can't know what a, b, m, and n are. We can extend this method to a hyper-rectangle which has multiple attributes to handle multi-dimensional spaces.

$$\begin{aligned} & D(C,X) > D(D,X) \;\&\&\; D(E,X) > D(F,X) \\ \Leftrightarrow\; & (D' - C') \cdot X' > 0 \;\&\&\; (F' - E') \cdot X' > 0 \end{aligned} \tag{7}$$

In [3,17], the complexity of conjunctive condition evaluations is $O(nd)$. The value n is the number of the data points and the value d is the number of dimensions. In contrast, the complexity of conjunctive conditions using ASPE is only $O(d)$.

4 Preventing Collusion Attacks

In ASPE, a subscriber knows the matrix M. So, if the subscriber is malicious, the system may not be secure since the broker can decode the notification using the matrix learnt from the subscriber. In order to solve this problem, we can employ a trusted security manager component. Instead of creating their own encrypted subscriptions, users send plaintext subscriptions to the security manager. For example, the subscriber sends an original subscription x to the security manager, and the latter assembles the encrypted subscription. $y' = M^T y^* = M^T(y, -0.5\|y\|^2)^T$, $z' = M^T z^* = M^T(z, -0.5\|z\|^2)^T$ which can be then sent to a broker. This way, the subscriber does not learn the transformation matrix. The security manager architecture is depicted in Figure 6.

Since the subscriber does not know the matrix, it can't collude with a broker. Also, the subscriber can't get the matrix from the original subscription x and the encrypted subscription y' and z' by Theorem 1. Even if the subscriber can get the value a from the notification a' by decoding it using a secret key, since the value a is not a column vector but an integer value, the subscriber can't learn the matrix.

Moreover, if subscribers want to send too many subscriptions, it can make the CBPS system insecure. If a subscriber or a broker knows all the subscriptions, it may attempt to guess the matrix used for ASPE. In this case, the security manager may restrict the rate of subscriptions in order to protect against this sort of attacks.

In addition, we can use an access control method at the security manager. A subscriber should send a plaintext subscription to the security manager through a secure channel. If the subscriber has an access right, it can receive the encrypted subscription from the security manager and receive a notification from a broker by forwarding the encrypted subscription.

When a subscriber leaves its subscribing group, the security manager makes a publisher change the matrix which is used for notification encryption or sends a new matrix to the publisher such that the subscriber can't receive notifications any more. The notification made by the new matrix can't be filtered by the prior subscription. So, a subscriber which lost an access right can't receive a notification.

Also, the broker can manage the subscriptions using a soft-state method. Thus, it enables access control by deleting a subscription which expired. In order to prevent the subscription from expiring, a subscriber should contact the security manager to renew an access right and forward a new encrypted subscription to a broker.

Changing the matrix whenever a subscriber leaves the system can be a burden to a secure CBPS. If subscribers do not leave frequently, changing the matrix is reasonable. Also, using soft-state in the brokers can reduce the frequency of changing the matrix.

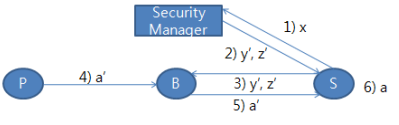

Fig. 6. Preventing collusion

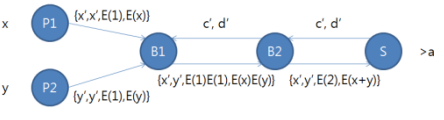

Fig. 7. Secure Aggregation using ASPE

5 Secure Aggregation Using ASPE

In this section, we suggest a method for secure aggregation in CBPS using ASPE in conjunction with homomorphic encryption. Specifically, we employ an additively homomorphic public-key cryptosystem for sum, such as the one in [21]. When there is a notification whose value is a and another notification whose value is b, two publishers can send the encrypted notification $E(a)$ and $E(b)$ by using the homomorphic encryption function. Then a broker computes the sum of the two values as follows: $E(a + b) = E(a) \cdot E(b)$

Note that, this encryption is applied to the notification content. The part of the notification that is used for routing will also be encrypted with the ASPE function, and this section of the notification is used by the brokers to make forwarding decisions. The format of the notification is the following:

Notification = <*min, max, count, sum*>

In order to do range filtering, a notification has an encrypted minimum value and an encrypted maximum value by using ASPE. For additive aggregation, it has a sum by

using homomorphic function. For example, a notification whose value is x is represented as $< x', x', E(1), E(x) >$. Since we used ASPE, x' is equal to $M^T x^* = M^T r(x, 1)^T$. Because the notification is a sum of only one notification, its count value is equal to $E(1)$. Figure 7 illustrates this process.

When a broker receives two notifications x' and y', if a subscriber wants to get a sum of notifications whose values are greater than a, the broker should check whether x and y are greater than a. As we mentioned in Section 3.2, we can use the inequality filtering method based on ASPE. The following condition should be examined where $c = a - s$ and $d = a + s$. (s is a random number.) First, if the following condition (8) is satisfied, we can verify whether x and y are greater than a. Then, the sum of the two notifications is computed as $E(x + y) = E(x) \cdot E(y)$.

$$\begin{aligned} &D(c, x) - D(d, x) > 0 \;\&\&\; D(c, y) - D(d, y) > 0 \\ &\Leftrightarrow (d' - c') \cdot x' > 0 \;\&\&\; (d' - c') \cdot y' > 0 \end{aligned} \tag{8}$$

Second, we have to determine which one is greater between the two values in order to get the minimum value and the maximum value. If the condition (9) is satisfied, we can know that x is less than y. Because x and y are greater than a in the first step, x and y are also greater than c.

$$D(x, c) < D(y, c) \Leftrightarrow (y' - x') \cdot c' < 0 \tag{9}$$

If we assume that x is less than y, the minimum value is x and the maximum value is y. So, the broker produces a new notification which has the following values.

$$\textit{<min, max, count, sum>} = < x', y', E(2),\ E(x + y) >$$

6 Experimental Evaluation

We implemented the proposed secure CBPS methods using ASPE in SIENA[4], a wide-area event notification service. SIENA is a content-based publish/subscribe infrastructure where brokers are vertices in a connected overlay acyclic graph. However, SIENA does not provide any security features. We evaluate the performance of our method in comparison with the C-CBPS solution proposed in [5].

As discussed in Section 2.2, C-CBPS supports secure equality filtering and range filtering using a re-mapping of subscriptions and notifications. It adapts a scheme from Song et al. [22] to support equality filtering and defines two schemes for inequality filtering and range filtering.

For equality filtering, C-CBPS computes the hidden value of an attribute by passing its plaintext value to a pseudorandom function with the secret key. The encrypted subscription is the hidden value of the plaintext. The encrypted notification has a random nonce r and the result of feeding the nonce r to a pseudorandom function. When the broker computes the value of feeding the nonce r to a pseudorandom function with the encrypted subscription, if the value is the same with the result contained in the notification, the notification satisfies the subscription.

For inequality filtering, C-CBPS chooses l points, $p_1, \ldots, p_l$ as reference points and considers the following dictionary: {"$>p_1$", "$>p_2$", … , "$>p_l$", "$<p_1$", "$<p_2$", … , "$<p_l$"}. Subscriptions will be approximated with one of these constraints. Each

notification is considered to be a document containing the words in the dictionary. For range filtering, in order to support $l_b < N < u_b$ subscriptions, C-CBPS has the publishers and subscribers agree on a partitioning $P = \{p_1, \ldots p_l\}$. The publisher encrypts the index of the subset which N belongs to by using equality filtering. The subscribers include as subscriptions encrypted versions of the indexes of the subsets in the partition they are interested in (i.e. all $p_i \in P$ such that $p_i \cap (l_b,u_b) \neq \emptyset$).

The security features of our method and C-CBPS are similar. However, our results show that the overhead incurred at the brokers by our method is clearly superior to the work in [5].

6.1 Evaluation Methodology

All the data used for testing are generated uniformly at random. In all the tests, a single instance of the enhanced SIENA matching engine was evaluated. All experiments were run on a 2.1Ghz Intel Core2 Duo CPU with 3GB of RAM running Windows Vista and Sun's JDK 1.6. Time is measured by using the function *System.nanoTime()*.

Matching time is measured as the time that the broker spends to identify the set of matching subscribers, when a notification is given. We measure the average matching time required to match a notification against 1000 subscriptions. Subscription and notification sizes are measured by total network bytes sent including SIENA's protocol overhead. We consider types of subscriptions which filter numeric attributes using arithmetic operators (i.e., =, <, >). We use the schemes with subscription covering enabled.

6.2 Matching Time Measurements

Notifications are integers generated uniformly at random from [0,10000]. To test *equality filtering*, we select subscriptions uniformly at random from [0,10000]. Figure 8(a) shows the results for equality filtering time. Our method using ASPE shows the similar performance with SIENA which uses the plaintext subscriptions and notifications. On the other hand, ASPE takes about 65% less time than C-CBPS for equality filtering. ASPE has a reference matching time of 2.7ms.

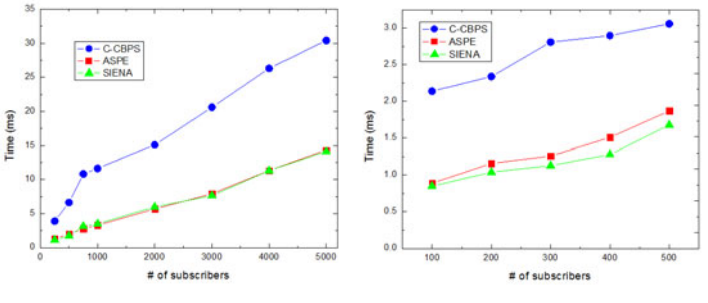

Fig. 8. (a) Equality filtering time (b) Range filtering time

Figure 8(b) shows the result for *range filtering*. ASPE takes about 50% less time than C-CBPS and shows similar performance with SIENA. C-CBPS uses subsets for range filtering. For example, each subset can have a range as follows: (0,50), (10,60), (20,70), etc. It used overlapped partitions for subscription confidentiality. Originally C-CBPS used 890 subsets which have values from [0,1000]. So, it had a false matching rate of 5%. There are about 1000000 subsets which have values from [0,1000]. A partitioning scheme with zero false matches for any range subscription has $|D|^2$ points, being quite expensive. So, it is not reasonable to use only 890 subsets. In order to compare the performance of C-CBPS with ASPE more precisely, we made C-CBPS use about 15000 subsets which have values from [0,10000]. However, it still has false positives.

Also, in C-CBPS, as there are more subsets, the size of a subscription is larger because the subscription should contain the information about the subsets when covering is enabled. When there are 890 subsets, the size of a subscription is 775 bytes. When we use about 15000 subsets, the size is 10728 bytes. So, when there are more than 500 subscribers in C-CBPS, the system runs out of memory. ASPE is scalable and does not incur false positives.

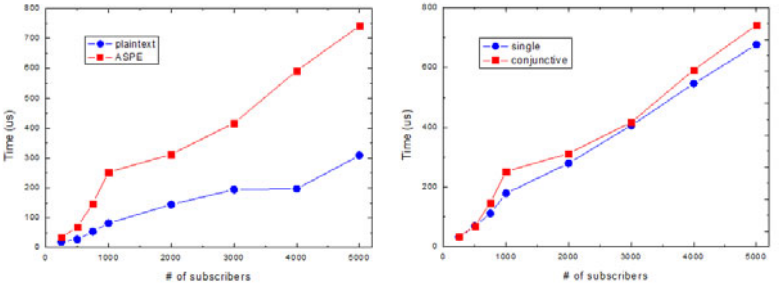

Fig. 9. (a) Conjunction filtering time (b) Single vs conjunctive condition filtering

Since SIENA and the C-CBPS do not support the conjunction filtering, we compared ASPE with plaintext filtering. Figure 9(a) shows the performance of ASPE which runs the conjunction filtering having two attributes. ASPE is about 2.5 times as expensive compared to plaintext filtering. But, since the reference matching time of ASPE for the conjunctive query is 0.25ms, it is still reasonable in practice.

Finally, Figure 9(b) gives the performance of the single condition filtering and conjunctive condition filtering using ASPE. Note that, the conjunctive condition filtering takes only about 10% more time than the single condition filtering. It shows that the conjunction filtering using ASPE is practical and scalable.

7 Conclusion

In this paper, we proposed a secure CBPS system based on Asymmetric Scalar-product Preserving Encryption in order to provide notification and subscription confidentiality and to reduce matching complexity. Our methods support equality filtering, inequality filtering, range filtering, covering, and conjunction filtering which

are essential in CBPS. In addition, our solution does not incur false positives, in contrast to existing work such as C-CBPS. Moreover, we suggested a new method for secure aggregation using ASPE and homomorphic functions. We can support sum, min, max and count functions for equality, inequality, and range filtering.

The experiment results show that our methods take about 65% less time in equality filtering and about 50% less time in range filtering than C-CBPS. Moreover, our secure conjunction filtering method incurs reasonable overhead when compared to plaintext conjunction filtering and single condition filtering. In future work, we intend to develop secure mechanisms for other types of subscriptions with more complex conditions.

Acknowledgments

The work reported in this paper has been partially supported by MURI award FA9550-08-1-0265 from the Air Force Office of Scientific Research.

References

1. Wong, W.K., Cheung, D.W., Kao, B., Mamoulis, N.: Secure kNN Computation on Encrypted Databases. In: ACM SIGMOD International Conference on Management of Data, pp. 139–152 (2009)
2. Agrawal, R., Kiernan, J., Srikant, R., Xu, Y.: Order Preserving Encryption for Numeric Data. In: ACM SIGMOD International Conference on Management of Data, pp. 563–574 (2004)
3. Boneh, D., Waters, B.: Conjunctive, Subset, and Range Queries on Encrypted Data. In: Theory of Cryptography Conference, pp. 535–554 (2007)
4. Carzaniga, A., Rosenblum, D.S., Wolf, A.L.: Design and evaluation of a wide-area event notification service. ACM Transactions on Computer Systems 19(3), 332–383 (2001)
5. Raiciu, C., Rosenblum, D.S.: Enabling Confidentiality in Content-Based Publish/Subscribe Infrastructures. In: International Conference on Security and Privacy in Communication Networks, pp. 1–11 (2006)
6. Akamai, `http://www.akamai.com`
7. Hacigumus, H., Iyer, B., Li, C., Mehrotra, S.: Executing sql over encrypted data in the database-service-provider model. In: ACM SIGMOD International Conference on Management of Data, pp. 216–227 (2002)
8. Hacigumus, H., Iyer, B., Mehrotra, S.: Providing database as a service. In: ICDE International Conference on Data Engineering, pp. 29–38 (2002)
9. Liu, K., Giannella, C., Kargupta, H.: An attacker's view of distance preserving maps for privacy preserving data mining. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) PKDD 2006. LNCS (LNAI), vol. 4213, pp. 297–308. Springer, Heidelberg (2006)
10. Minami, K., Lee, A.J., Winslett, M., Borisov, N.: Secure Aggregation in a Publish-Subscribe System. In: ACM Workshop on Privacy in the Electronic Society, pp. 95–104 (2008)
11. Ahmed, W., Khokhar, A.: Secure aggregation in large scale overlay networks. In: Global Telecommunications Conference, pp. 1–5 (2006)
12. Srivatsa, M., Liu, L.: Securing Publish-Subscribe Overlay Services with EventGuard. In: ACM Conference on Computer and Communications Security, pp. 289–298 (2005)

13. Oliveira, S.R.M., Zaiane, O.R.: Privacy preserving clustering by data transformation. In: 18th Brazillian Symposium on Databases, pp. 304–318 (2003)
14. Nabeel, M., Shang, N., Bertino, E.: Privacy-Preserving Filtering and Covering in Content-Based Publish Subscribe Systems. CERIAS Tech. Report 2009-15, Purdue University, West Lafayette, IN
15. Wang, C., Carzaniga, A., Evans, D., Wolf, A.L.: Security Issues and Requirements for Internet-Scale Publish-Subsribe Systems. In: Hawaii International Conference on System Sciences, pp. 303–310 (2002)
16. Khurana, H.: Scalable Security and Accounting Services for Content-based Publish/Subscribe Systems. In: ACM Symposium on Applied Computing, pp. 801–807 (2005)
17. Shi, E., Bethenourt, J., Hubert Chan, T.-H., Song, D., Perrig, A.: Multi-Dimensional Range Query over Encrypted Data. In: IEEE Symposium on Security and Privacy, pp. 350–364 (2007)
18. Pesonen, L.I.W., Eyers, D.M., Bacon, J.: Encryption-Enforced Access Control in Dynamic Multi-Domain Publish/Subscribe Networks. In: International Conference on Distributed Event-Based Systems, pp. 104–115 (2007)
19. Hore, B., Mehrotra, S., Tsudik, G.: A Privacy-Preserving Index for Range Queries. In: International Conference on Very Large Data Bases, pp. 720–731 (2004)
20. Papadopoulos, et al.: Continuous Authentication on Data Streams. In: International Conference on Very Large Data Bases, pp. 135–146 (2007)
21. Paillier, P.: Public-key cryptosystem based on composite degree residuosity classes. In: Advances in Cryptology, pp. 223–238 (1999)
22. Song, D., Wagner, D., Perrig, A.: Practical techniques for searches on encrypted data. In: IEEE Symposium on Security and Privacy, pp. 44–55 (2000)