© 2015 IJSRSET | Volume 1 | Issue 2 | Print ISSN : 2395-1990 | Online ISSN : 2394-4099
Themed Section: Engineering and Technology

Identity Based Encryption for Securing Publish Subscribe System

Harismita. H, Laavanya.R, Meenaa.R, Kalai Selvi
Information Technology, Dhanalakshmi College of Engineering, Chennai, Tamilnadu, India

ABSTRACT

We have reachable a new approach to provide authentication and confidentiality in a broker-less content-based publish/subscribe system. Security is highly challenging in this system. In the project, security is provided by adapting the Cipher text policy attribute based encryption. This over all approach provides fine-grained key management and the efficient cost for encryption, decryption, and routing in the order of subscribed attributes, based upon credentials.

Keywords: Publish/subscribe, P2P, Security

I. INTRODUCTION

Publishers sends information into the publish/subscribe system, and subscribers precise is the topic for subscriptions. Published events are clustered to their relevant subscribers, without the publishers knowing the subscriber. This decoupling is traditionally ensured by intermediary routing over a broker network. In current systems the publish/subscribe system uses content based routing which does not provide authentication. Access control in the context of publish/subscribe system means that only authenticated publishers are allowed to disseminate events in the network and only those events are delivered to authorized subscribers. These security issues are not trivial to solve in a content-based publish/ subscribe system and pose new challenges. For instance, end-to-end authentication using a public key infrastructure (PKI) conflicts with the loose coupling between publishers and subscribers.

Publishers must maintain the public keys of the interested subscribers in order to encrypt events. Subscribers on the other hand, must know the public keys of all the relevant publishers in order to verify the authenticity of the received events. Event message conflicts with the content-based routing therefore. In this paper, we present a new approach to provide authentication publish/subscribe system. Their subscriptions. Private keys assigned to the subscribers are labelled A publisher associates each encrypted event with a set of credentials. We adapted identity based encryption mechanisms to ensure that a particular subscriber can decrypt an event only if there is match between the credentials associated with the event and the key. A weaker notion of subscription confidentiality is defined and a secure connection protocol is designed to preserve the weak subscription confidentiality.

II. METHODS AND MATERIAL

A. System Model

Content-based publish/subscribe For the routing of events from publishers to the relevant subscribers we use the content-based data model. The event space, denoted by , is composed of a global ordered set of distinct attributes (Ai): = fA1; A2; : : : ;Adg. Each attribute Ai is characterized by a unique name, its data type and its domain An event is matched against a subscription f (and subsequently delivered to the subscriber), if and only if the value of attributes in the event stases the corresponding\constraints imposed by the subscription. Let Ef1 and Ef2 denote the sets of events matching subscription f1 and f2, respectively. Then f1 is said to be covered by another subscription f2 exits no dedicated broker infrastructure. Publishers and Subscriber contribute as peers to the maintenance of an self-organizing overlay structure. Peers can join the overlay by contacting an arbitrary peer and thereafter subscribe and publish events. In order to authenticate publishers we use the concept of advertisements in which a publisher announces beforehand the set of events which it intends to publish. There are three major goals for the proposed secure publish/ subscribe system, namely to support authentication, confidentiality and scalability:

Authentication: In order to avoid non-eligible publications only authorized publishers should be able to publish events in the system. Similarly, subscribers should only receive those messages to which they are authorized to subscribe.

Confidentiality: In a broker-less environment, two aspects of confidentiality are of interest: i) the events are only visible to authorized subscribers and are protected from illegal modifications; ii) the subscriptions of subscribers are condential and unforgivable.

B. Identity Based Encryption

While a traditional PKI infrastructure requires to maintain for each identity a private/public key pair which has to be known between communicating entities to encrypt and decrypt messages, Identity-based encryption it provides a promising alternative to reduce the amount of keys to be managed. In identity-based encryption (IBE), any valid string which uniquely identifies a user can be the public key of the user. A key server maintains a single pair of public and private master keys. The master public key can be used by the sender to encrypt and send the messages to a user with any identity, e.g. an email address. To successfully decrypt the message, a receiver needs to obtain a private key for its identity from the key server.

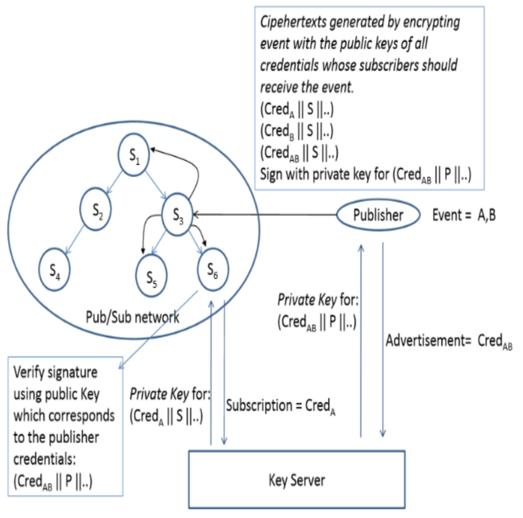
Cipehertexts generated by encrypting event with the public keys of all credentials whose subscribers should receive the event.
(Cred$_A$ || S ||..)
(Cred$_B$ || S ||..)
(Cred$_{AB}$ || S ||..)
Sign with private key for (Cred$_{AB}$ || P ||..)
S$_1$
S$_2$
S$_3$
S$_4$
S$_5$
S$_6$
Pub/Sub network
Publisher
Event = A,B
Private Key for: (Cred$_{AB}$ || P ||..)
Advertisement= Cred$_{AB}$
Verify signature using public Key which corresponds to the publisher credentials: (Cred$_{AB}$ || P ||..)
Private Key for: (Cred$_A$ || S ||..)
Subscription = Cred$_A$
Key Server

Figure 1: Approach overview publisher has credentials with two attribute A and B. Subscriber s6 has a credential to receive events with attribute A.

We want to stress here that although identity-based encryption at the first glance, appears like a highly centralized solution, its properties are ideal for highly distributed applications. A sender needs to know only a single master public key in order to communicate with any identity. Similarly, a receiver only obtains private keys for its own identities. Furthermore, an instance of central key server can be easily publisher/subscriber network

1. **Bilinearity:** e(ux; vy) = e(uy; vx) = e(u; v)xy, for all u; v 2 G1 and x; y 2 Zp.
2. **Non-degeneracy**: e(u; v) 6= 1, for all u; v 2 G1.
3. **Computability**: e can be efficiently computed.

C. Approach Overview

For providing security mechanisms in publish/subscribe we rely on the methods of identity-based encryption and adapt it in order to support many-to-many interactions between subscribers and publishers. Publishers and subscribers interact with a key server by providing credentials to the key server. In this case we say the credential is authorized by the key server. Consequently, a credential consists of two parts: first a binary string which describes the capability of a peer in publishing and receiving events, and second a proof of its identify While this can happen in a variety of ways, e.g. relying on challenge response, hardware support, etc., we pay attention mainly at expressing the capabilities of a credential, i.e. how subscribers and publishers can create a credential. This process needs to account the many possibilities to partition the set of events expressed by an advertisement or subscription and exploit overlaps in subscriptions and publications. Subsequently, we use the term credential only for referring to the capability string of a credential. The keys assigned to publishers and subscribers, and the cipher texts are labelled with credentials.

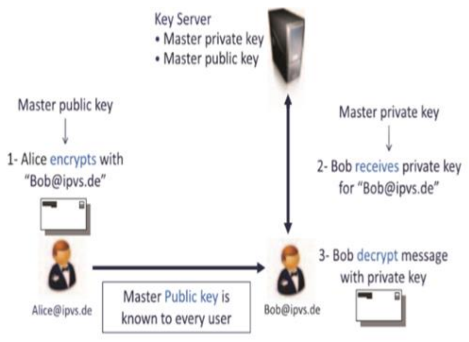
Key Server
• Master private key
• Master public key
Master public key
1- Alice encrypts with "Bob@ipvs.de"
Master private key
2- Bob receives private key for "Bob@ipvs.de"
3- Bob decrypt message with private key
Alice@ipvs.de
Master Public key is known to every user
Bob@ipvs.de

Figure 2: Identity-based encryption

D. Creation of Credentials

First of all we have to find a systematic way of decomposing the event space for a content-based subscription model. Later we show how subscriptions and advertisements are mapped to the subspaces of the event space and credentials are created. Further extensions like considering string attributes and complex subscriptions are discussed subsequently.

a) Event Space Numeric Attributes

Decomposition of the event space by focusing on numeric attributes takes place. Later we will discuss also how other types of attribute supported. The event space, composed of d distinct numeric attributes can be geometrically modelled as a d-dimensional space such that each attribute represents a dimension in the space. Subscriptions and advertisements are represented by hyper rectangles in the space, whereas published events represent points. With the spatial indexing approach, the event space is hierarchically decomposed into regular subspaces, which serve as enclosing approximation for the subscriptions and advertisements. Each tree level represents one step of the recursive process, starting with the root where the event space is still undivided.

b) Mapping To Credentials

Subscription or advertisement of a peer can be composed of several subspaces. A credential is assigned for each of The mapped subscriptions. An event can be approximated by the smallest subspace that encloses the point represented by it. To deliver the encrypted event a cipher text must be generated for each subspace that encloses the event so that the peer whose subscription mapped to any of these subspaces should be able to successfully decrypt the event. An event dze matches (is enclosed in) a subspace dzs if dze is covered by dzs. In general, the number of subspaces matched by an event dze is in the order of log2 (Qdi=1 Ti) and is equal to jdzej+1

c) 5.3 Complex Subscriptions

For a complex subscription with predicates on different attributes, a subscriber receives separate credentials and thus keys for each attribute. Using these keys, a subscriber should be able to successfully decrypt any event with the corresponding Attributes, if he is authorized to read the values associated with the attributes. Any cryptographic primitive can be easily used for this purpose. In a content-based publish/subscribe system. A subscription defines a conjunction on predicates. An event matches a subscription if and only if all of the predicates in the subscription are satisfied. To ensure event confidentiality, a subscriber must not be able to successfully decrypt any event which matches only parts of its subscriptions. However, assigning keys for individual attributes and XOR based decryption does not prevent this behaviour. For example, consider a subscriber with two subscriptions

d) Methods For Security

In this section we will describe the construction of security mechanisms to achieve authentication of publishers and subscribers as well as confidentiality of events. One naive solution would be to directly use the techniques from PKI by assigning public/private key pair to each credential. Publishers and subscribers can contact key server to obtain the public/private key pairs that corresponds to their credentials. However, PKI does not provide a mechanism to bound together the public/private key pairs associated with the same subscription and therefore, cannot be used. The security mechanisms described in this section are adapted from attribute-based encryption schemes.

In particular, our modifications,
i) Allow publishers to sign and encrypt events at the same time by using the idea of identity-based signcryption.
ii) Include some additional cipher texts that increase the efficiency of the system and,
iii) Allow subscribers to verify the signatures associated with all the attributes simultaneously. Our modifications do not change the basic structure of the schemes and therefore preserves the same security strength.

e) Publishing Event

When a publisher wants to publish an event M, it chooses at random for each attribute Ai of the Event, such that q = Pdi=1 qi. These random values ensure that only the subscribers who have matching credentials for each of the attributes should be able to decrypt the event. Furthermore, it generates a length random key SK. To encrypt an event, a publisher uses master public key and performs the following steps:

Step1: Compute The cost of asymmetric encryption generally increases with the size of the plaintext. Therefore, only a fixed length random key SK is encrypted using the private keys of publisher. The actual event message M is encrypted with a symmetric encryption algorithm such as AES, using the key SK. The cipher text C2 also includes the public keys of the credentials which authorizes the publisher to send the event. The inclusion of these public keys increases the efficiency of signature verification process at the expense of a small increase in the cipher text size (one public key per attribute).
Step2: For each attribute, a cipher text should be send for each credential that matches its value. For example, in case of a numeric attribute with value mapped to 0000, a cipher text should be disseminated for the credentials 0000,000,00 and 0. For each credential Cred i;j that matches the value of the attribute Ai, compute Ci;j = (u0Q k2□i;j uk)qi , where □i;j is calculated as described above. The cipher texts are ordered according to the containment relationship (in descending order) between their associated credentials.

f) Receiving Event

On receiving the cipher texts, a subscriber trie to decrypt them using its private keys. The cipher texts for each attribute are strictly ordered according to the containment relation between their associated credentials, therefore a subscriber only tries to decrypt the cipher text whose position coincides with the position of its credential in the containment hierarchy of corresponding attribute. The position of a credential can be easily determined by calculating its length. For example, for a numeric attribute, credential 0000 occupies 4th position in the containment hierarchy i.e. after 0,00 and 000. Subscribers decrypt the cipher text in the following manner. and event dissemination mechanisms (Section 6) ensure that subscriber knows the exact credential needed to decrypt the event. Verification. A subscriber will only accepts the message if it is from an authorized publisher. The received event is authentic if the following identity holds: VL = VR1 _ VR2 _ VR3 Remember the cipher text C2 contains the public keys of the credentials which authorize the publisher to send the event. If no such public keys are included in C2, then the subscriber should try to authenticate the event by checking for all possible credentials which a publisher might hold to publish the event. For example, an event with a single Numeric attribute and a value mapped to 0000 can be published by the publisher with credentials 0000, 000,00 or 0. In this case our approach is as follows: a subscriber checks the authenticity of the event for each attribute Ai separately2, by verifying that for one of the possible credentials Credi;j the following identity holds: In this case the total verification cost is Pd i=1 log2(Ti)..

E. Subscription Confidentiality

In this section, we address subscription confidential in broker-less publish/subscribe system, where publishers and subscribers are responsible for maintaining the overlay network and forwarding events to relevant subscribers. First, we describe the maintenance of the publish/subscribe overlay network. Later we done a weaker notion of subscription confidentiality and detail the mechanisms behind.

The publish/subscribe overlay is a virtual forest of logical trees, where each tree is associated with an attribute A subscriber joins the trees corresponding to the attributes of its subscription. Similarly, a publisher sends an event on all the trees associated with the attributes in the event. Within each attribute tree, subscribers are connected according to the containment relationship between their credentials associated with the attribute. The subscribers with coarser credentials (e.g. the ones mapped to coarser subspaces in case of numeric attributes) are placed near the Tree of Attribute A1 Tree of Attribute A2 Figure 6: Publish/Subscriber system with two numeric attributes of the tree and forward events to subscribers with credentials. A subscriber with more than one credentials can be handled by running multiple virtual peers on a single physical node, each virtual peer maintaining its own set of tree links. For example in Figure 6, the subscriber s3 has two credentials f000; 010g and is connect to two places in the tree.

In order to connect to an attribute tree, a newly arriving subscriber sn sends the connection request along with its credential to a random peer sr in the tree. The peer sr compares the request credential with its own; if the peer's credential covers the request credential and the peer can accommodate more children, it accepts the connection. Otherwise, the connection request is forwarded to all the children with covering credentials and the parent peer with exception of the peer, from which it was received. In this way the connection request is forwarded by many peers in the tree before it reaches the suitable peer with covering credential and available connection.

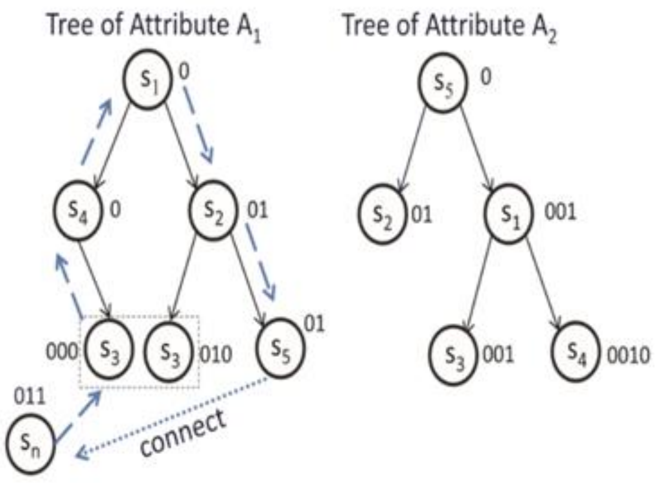

Figure 3: Pub/Sub System with two attributes

Figure 3 shows the path followed by a request from a subscriber until it reaches the desired parent subscriber. The drawback of maintaining separate trees for each attribute is that the subscribers also receive events that match only a part of their subscription (false positives). However, it cannot receive event confidentially because false positives cannot be decrypted without having required credentials for each attribute.

a) Weak Subscription Confidentiality

It is infeasible to provide strong subscription confidentiality in a broker-less publish/subscribe system because the maintenance of the overlay topology requires each peer to know the subscription of its parent as well as its children.
To address this issue, a weaker notion of subscription confidentiality is required.

Definition 6.1. Let s1 and s2 denote two subscribers in a publish/subscribe system which both possess credentials for an attribute Ai. Weak subscription confidentiality ensures that at most the following information can be inferred about the credentials of the subscribers:

1. The credential of s1 is either coarser or equal to the credentials of s2.
2. The credential of s1 is either _ner or equal to the credentials of s2.

b) Secure Connection Protocol

In the following, we propose a secure connection protocol, which maintains the desired overlay topology without violating the weak subscription confidentiality. For simplicity and without loss of generality, here we discuss the secure connection protocol with respect to a single tree associated with a numeric attribute Ai and each of the subscribers owns a single credential. The secure protocol is based on the idea that in the tree subscribers are always connected according to the containment relationship between their credentials, e.g. a subscriber with credential 00 can only connect to the subscribers with credentials 0 or 00. A new subscriber s encrypts secret words3 with the public keys Pus i;j for all credentials that cover its own credential e.g. a subscriber with credential 00 will generate cipher texts by applying the public keys Pus i;0 and Pus i;00. The generated cipher texts are added to a connection request (CR) and the request is forwarded to a random peer in the tree. A connection is established if the peer can decrypt any of the cipher texts using its private keys.

Filling the security gaps: By looking at the number of cipher texts in the connection request the peer could detect the credential of the requesting subscribers. For example, a subscriber with credential 00 can only connect to 0 or 00 and therefore, a connection request will have two cipher texts, whereas the connection request for 000 will have three cipher texts. In the worst case, a subscriber has a credential of the _nest granularity. This can be covered by log2(Ti) other credentials and therefore a connection request contains in the worst case that many cipher texts. To avoid any information leak, cipher texts in the connection request are always kept in O(log2 Ti) (O(L) for pre_x matching) by adding random cipher texts if needed. Furthermore, the cipher texts are to avoid any information leak from their order.

Algorithm: Secure overlay Maintainer Protocol

```
1: upon event Receive(CR of $s_{new}$ from sp) do
2: if decrypt request(CR) == SUCCESS then
3: if degree(sq) == available then // can have child peers
4: connect to the $s_{new}$
5: else
6: forward CR to fchild peers and parentg □ sp
7: if decrypt request(CR) == FAIL then
8: if sp == parent then
9: Try to swap by sending its own CR to the snew
10: else
11: forward to parent
```

A child peer sq receives CR (of subscriber s_{new}) from the parent only if the parent cannot accommodate more children. If sq cannot be the parent of s_{new}, i.e., s_{news} credentialis coarser than that of sq, then it tries to swap its position with s_{new} by sending its own connection request (cf. Algorithm 1, lines 7-9). However, if none of the children of parent sp can connect or swap with s_{new}

then there is no containment relationship between the credentials of the children and . In this case a parent should disconnect one of its children in order to ensure the new subscriber is connected to the tree.

6.4 Discussion for an attribute Ai, let S_ be the set of peers in the system whose credentials covers the credential of the subscriber s1. Let S_ denote the set of subscribers whose credentials.

III. RESULTS AND DISCUSSION

Secure Event Dissemination

The secure connection protocol ensures that the credential of a parent peer covers the credentials of its children. Therefore, a parent peer can decrypt every event, which it forwards to the children. Regardless of the cryptographic primitives, a parent can eventually discover the credentials of its child peers e.g. by maintaining histories. In our approach we used one hop flooding to avoid this problem. In one hop flooding, a parent assumes that the children have the same credentials as its own and forwards each successfully decrypted event to all of them. In turn the children forward each event which was successfully decrypted to all of their children and so on. In this strategy, a child may have credentials then its parent and may receive false positives. The detailed mechanism works as follows: To publish an event, a publisher forwards the cipher texts of each attribute to a randomly selected subscriber on the corresponding attribute tree. All the cipher texts of an event are labelled with a unique value such as sequence number of the event. This helps subscribers to identify all the cipher texts of an event(as cipher text for each attribute are received on the separate tree).

Evaluations

Similar to Event Guard we evaluated our solution in two aspects: i) quantifying the overhead of our cryptographic primitives, and ii) evaluating the performance of our secure publish/subscribe system by benchmarking it with an Unsecured system. 8.1 Performance of Cryptographic primitives In this section, we measure the computational overhead of our security mechanisms. The security mechanisms are implemented by Pairing-Based Cryptography (PBC) library [12]. The implementation uses a 160-bit elliptic curve group based on the super singular curve y2 all reporting values are averaged over 1000 measurements. The message size is kept 128 bytes as this good enough for most symmetric encryption algorithms4. One-hop flooding (OHF) Table 3 shows the overhead from the perspective of publishers and subscribers in our system. In 4In our system pairing based encryption is used to encrypt a random key SK, which is later used to decrypt actual event using symmetric encryption with a uniform event distribution. Publish/subscribe overlay construction. We measured the average latency experienced by each subscriber to connect to a suitable position in an attribute tree. Latency is measured from the time subscriber sends connection request message to a random peer in the tree till the time the connection is actually established.

IV. CONCLUSIONS

In this paper, we have presented a new approach to provide authentication and confidentiality in a broker-less content-based publish/subscribe system. The approach is highly scalable in terms of number of subscribers and publishers in the system and the number of keys maintained by them. In particular, we have developed mechanisms to assign credentials to publishers and subscribers according to their subscriptions and advertisements. Private keys assigned to publishers and subscribers, and the cipher texts are labelled with credentials events. Furthermore, we developed a protocol to preserve the weak subscription confidentiality in the presence of semantic clustering of subscribers.
Example: application includes news distribution, stock exchange, environmental monitoring, traffic control, and public sensing.

V. REFERENCES

[1]. Muhammad Adnan Tariq, Boris Koldehofe, and Kurt Rothermel IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 25, NO. 2, FEBRUARY 2014.

[2]. W.C. Barker and E.B. Barker, “SP 800-67 Rev. 1. Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher,” technical report, Nat’l Inst. of Standards & Technology, 2012.

[3]. S. Choi, G. Ghinita, and E. Bertino, “A Privacy-Enhancing Content-Based Publish/Subscribe System Using Scalar Product Preserving Transformations,” Proc. 21st Int’l Conf. Database and Expert Systems Applications: Part I, 2010.

[4]. J. Bacon, D.M. Eyers, J. Singh, and P.R. Pietzuch, “Access Control in Publish/Subscribe Systems,” Proc. Second ACM Int’l Conf. Distributed Event-Based Systems (DEBS), 2008.