Secure Publish-Subscribe Mediated Virtual Organizations

Richard Ssekibuule
Department of Computer Science
Faculty of Computing and IT, Makerere University
Kampala, Uganda
Email: rkayondo@cit.mak.ac.ug

***Abstract*—Digital technologies such as publish-subscribe systems present dynamic services support for inter-organizational activities. In order for these systems to achieve usage acceptance, various security requirements have to be met by the enabling technologies. In this article, we focus on confidentiality, privacy and integrity requirements for Publishers and Subscribers in a Publish-Subscribe mediated electronic market. We consider a virtual organization architecture, in which subscribers dynamically join and leave various organizations. We review techniques previously suggested in literature for providing confidentiality, privacy and integrity requirements and then present a new solution which is based on cryptographic hashes and public-key cryptography.**

***Keywords*-Publish-Subscribe, Confidentiality, Integrity, Hash, Cryptography**

I. Introduction

Publish/Subscribe systems are commonly used in highly dynamic environments and systems to support many-to-many service and communication requirements. Such electronic markets typically involve suppliers advertising their goods and services in a common electronic repository. Customers in the environment would then search for these goods and contact suppliers for completion of the transaction. A prominent model for supporting customers and suppliers in autonomic markets was previously been defined by Guttman et al. [1], [2] in a *Consumer Buying Behavior (CBB) model* mediated by agents. Figure 1 below presents the seven stages that constitute a CBB model. The seven stages of the CBB model are: (1) *need identification*, (2) *product brokering*, (3) *buyer coalition formation*, (4) *merchant brokering*, (5) *negotiation*, (6) *purchase and delivery* and (7) *product or service evaluation*. It is worth mentioning that not all stages described in the CBB model are mandatory for consumer transactions. Some transactions may-not for example involve negotiations or formation of coalitions. In this article, we consider privacy, confidentiality and integrity challenges for an electronic market in which two or more virtual organizations are mediated by publish-subscribe systems. This article focuses on *product identification* (by consumers) , *product brokering* and *merchant brokering* (by the Brokers) of the CBB model. Of course suppliers (publishers) have to send in their products (publications) too.

The other stages of the CBB model such as buyer coalition formation, negotiation, product delivery and evaluation are outside the scope of this article. Figure 2 below presents a high-level overview for the virtual organization architecture under consideration. The organizational setup is mediated by

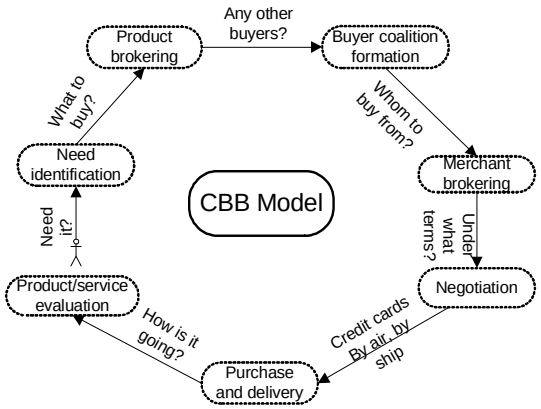

Figure 1. *Consumer Buying Behavior Model*

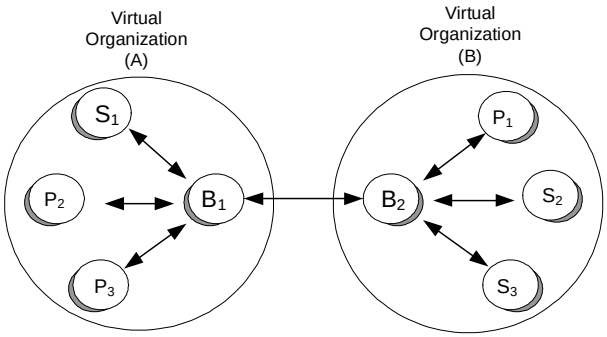

Figure 2. *Publish-Subscribe Mediated Market*

Publish-Subscribe systems. The Brokers (B_1 and B_2) are responsible for aggregating subscriptions and publications from Subscribers (S_1, S_2 and S_3) and Publishers (P_1, P_2 and P_3) respectively. Virtual organizations A and B collaborate on subscriptions and publications through Brokers B_1 and B_2. In case Subscriber S_1 is interested in a subscription that is not available in B_1, the request would be forwarded to B_2 by B_1. The presented scenario requires information to be handled and disseminated across virtual organizations containing Subscribers and Publishers with diverse interests;

978-1-4244-5494-5/10/$26.00 ©2010 IEEE

some of which could be malicious towards some Publishers or subscribers.

This article presents a security scheme for enforcing confidentiality and integrity of messages exchanged and privacy of participants in Publish-Subscribe mediated virtual organizations. Section II presents security requirements and assumptions for the scheme. Section III presents related work previously presented in literature. The proposed security scheme is presented in section IV and section V presents an analysis and evaluation of the solution. Concluding remarks are presented in section VI.

II. Security Requirements and Assumptions

This article considers a scenario in which Subscribers and Publishers do not wish to reveal their identities to other participants in a publish-subscribe mediated system.

An environment in which Publishers, Subscribers and Brokers are interested in learning about transaction details of other participants is assumed. This implies that curious Brokers may want to know what Publishers and Subscribers are respectively selling and buying. Publishers may want to access information concerning transactions for particular Subscribers. Conversely, curious Subscribers could be interested in knowing identities of Publishers selling particular categories of products.

We consider three categories of security requirements, namely; confidentiality, privacy and integrity. These are further explained in the enumeration below.

1) ***Confidentiality***: This requirement applies to the objects which are being traded in the publish-subscribe mediated systems. Publications (offers) and Subscriptions (requests) are supposed to be kept secret from all participants in the environment
2) ***Privacy***: This requirement applies to identities of Publishers and Subscribers. Publishers and Subscribers do not wish to have their identities revealed to any parties (i.e. Publishers, Subscribers and Mediators) in the environment.
3) ***Integrity***: Messages and objects exchanged in the trading protocols are supposed to be protected from tampering by unauthorized entities.

Use-case scenarios for such requirements can be found in the stock market or job recruitment agencies. Considering an example of recruitment agencies, organizations may desire their identities to be kept secret until subscribers or job applicants have sent their applications. Applicants may also desire their job searches to be kept private in the publish-subscribe system.

III. Related Work

This section presents a review of solutions previously suggested in literature to provide for confidentiality, privacy and integrity requirements in publish-subscribe mediated environments. Their weaknesses are highlighted in this section and a new scheme to mitigate these weaknesses is presented in section IV.

A. Multi-Layer Encryption Scheme

Shikfa et al. [3] suggested a technique for guaranteeing confidentiality and privacy in publish-subscribe networks by using commutative cryptography [4]. This scheme assumes that all the correspondents in the publish-subscribe environment have already received shared keys. The suggested scheme relies on the commutative property of the Pohlig-Hellman cryptosystem [5] to allow removal of encryption layers while preserving others in order to enforce data confidentiality. The encryption and decryption keys in the Pohlig-Hellman cryptosystems differ; thus having asymmetric properties.

This cryptographic scheme requires that the encryption and decryption keys are kept secret since anyone can generate a second key with knowledge of either of the keys. Security of the cryptographic system depends on hardness of the discrete logarithmic problem. The multi-layer encryption scheme suggested employs four security primitives, namely: (1) Encrypt-Filter; Used by subscribers to create filters. (2) Encrypt-Notification; Used by publishers to encrypt notifications. (3) Secure-Lookup; Used by Brokers to perform matching functions of filters to notifications. (4) Secure-Table-Building; Used by Brokers to build a routing table and compare encrypted subscriptions. The design of the four security primitives is based on commutativity properties of the Pohling-Hellman scheme. Brokers can add and suppress encryption layers on entries in the routing table and then remove some of them during lookup without accessing details of the subscriptions or publications.

There are several short-comings of secure routing with multi-layer encryption using a Pohlig-Hellman (commutative-homomorphic) scheme. Firstly, is the key-distribution problem. Addition and removal of encryption layers depends on the participants respectively having the required encryption and decryption keys. This is a challenge for commutative cryptography where both (encryption and decryption) keys are required to be kept secret. Unfortunately, the Pohlig-Hellman scheme by design allows a second key to be generated with knowledge of one of the keys. If an adversary or curious brokers gains access to one of the keys they can generate the corresponding second key.

This challenge is assumed to be non-existent by Shikfa et al. [3]. We find this assumption impractical. Key distribution and secrecy of all keys cannot be ignored for practical deployment of such a system.

Secondly, Shikfa et al. [3] assume non-dynamic subscribers. This is clearly a problem, since an on-line system cannot be expected to have a static number of users. Consequently, assumptions concerning publishers having secret keys for all available subscribers are highly flawed. The secrecy of keys in commutative cryptography requires a

978-1-4244-5494-5/10/$26.00 ©2010 IEEE

high degree of trust among entities possessing the keys of correspondents. Any accidental or malicious disclosure of one of the keys would lead to an absolute violation of all security requirements since corresponding keys can be generated from either of the encryption or decryption keys.

Lastly, practical deployment of cryptographic schemes requires implementation of the indistinguishability property so that a cryptanalyst may not gain useful information from a given cipher-text. This property when implemented in the scheme presented by Shikfa et al. [3], would make cipher-text comparisons absolutely impossible.

B. Secure Publish/Subscribe and Matching Schemes

This subsection presents an overview of Publish/Subscribe systems and matching schemes which have previously been presented in literature for secure content delivery.

1) An event notification service which has gained popularity in Publish/Subscribe literature was previously presented by Carzaniga et al. [6]. The architecture of the system is independent of application implementation and it aims to facilitate publication of events by their creators and subscriptions by interested recipients. The system provides routing services for notifications matching between publications and subscriptions. As far as security is concerned, this system puts emphasis on availability and robustness of service delivery. Little attention is given to confidentiality and anonymity (privacy) requirements. Chandramouli et al. [7] also presented a similar scheme for a wide-area based Publish/Subscribe system and their main security concern was still scalability given heavy usage conditions.
2) Raiciu et al. [8] presented a Publish/Subscribe content-based infrastructure in which confidentiality of notifications and subscriptions are enforced. This system considers brokers to be the only threat to confidentiality and privacy of participants. Publishers and Subscribers are considered to be honest. This is clearly a problem, since curious Publishers may abuse privacy of Subscribers and vise-versa.

IV. Proposed Solution

This section presents our suggestions for a solution to enforce privacy, confidentiality and integrity in publish-subscribe mediated virtual organizations. The proposed solution is built on well established security techniques such as cryptographic hashes and public-key cryptography. Trusted Anonymizers (TA) are also used to facilitate processes required for enforcing security requirements specified in section II. A principle for separation of privileges [9] is used to enforce subscriber and publisher privacy.

The remainder of this section is organized as follows: The design of the proposed system is presented in subsection IV-A and details of the proposed security protocol are discussed in subsection IV-B. In subsection IV-C, an analysis of cryptographic hash functions and their implications to security requirements of the systems is presented.

A. Proposed System Design

An overview of interactions between Brokers, Subscribers and Publishers was previously presented in subsection IV-A without details of security techniques deployed in the system. Figure 3 below presents a new architecture for security in publish-subscribe mediated virtual organizations. The Trusted Anonymizers (TA_i) are used to enforce privacy requirements by acting as intermediaries for message exchanges between the Broker (B_l), Subscribers (S_j) and Publishers (P_k). The crossed arrow in figure 3 is an indicator

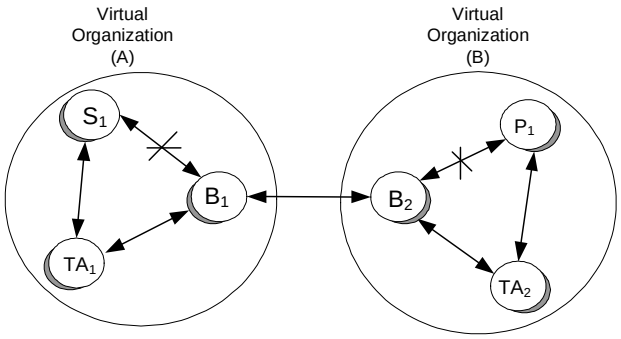

Figure 3. *Secure Publish-Subscribe Mediated Market*

that no direct interactions should happen between the two entities. The Brokers (B_l) should never communicate directly with the Publishers (P_k) or Subscribers (S_j). Section IV-B below presents details of the security protocol used to enforce confidentiality, privacy and integrity requirements for participating parties.

B. Security Protocol

The TA is the only trusted element in the security protocol for this publish-subscribe mediated system. As previously indicated in the introduction section, security requirements are enforced using public-key cryptography and cryptographic hashes. The assumptions presented below are considered for the security protocol supporting secure interactions between the Publishers (P), Subscribers (S), Trusted Anonymizers (TA) and Brokers (B).

1) All participating entities can obtain verifiable public-keys for their correspondents. That is to say, a public-key server is available to all entities to obtain valid digitally signed keys.
2) We also assume that the system inherits generic assumptions from public-key cryptography. Such assumptions include secret-key confidentiality.
3) The scheme trusts that the TAs will not reveal routing information to adversaries. This implies that routing information collected by the TAs must be protected

978-1-4244-5494-5/10/$26.00 ©2010 IEEE

and their computations are assumed to take place on a trusted platform.

4) The software executing on the Publishers' and Subscribers' terminals is assumed to be trusted. This software is responsible for generating the salt to be concatenated with the requests and offers. The process of concatenating a salt to the messages is strictly automated (achieved by software). Publishers and Subscibers cannot choose whether or not to concatenate a salt to their requests or offers. This operation is mandatory and is enforced through software on the Publishers and Subscribers terminals.

1) Publications: This subsection presents the component of the protocol for securely publishing content in the publish-subscribe system. In (1) and (2) formal components of the protocol are presented.

$$P_1 \rightarrow TA_2 : [hash(offer||salt)||TS_1]_{KU_{TA_2}} \quad (1)$$

$$TA_2 \rightarrow B_2 : [hashed\_offer||TS_2]_{KU_{B_2}} \quad (2)$$

Where $hashed\_offer = hash(offer||salt)$

TS_1 and TS_2 are timestamps which are supposed to be checked by the recipient in order to detect replay attacks. Furthermore, offers submitted by the publishers are sent as hashed values after concatenating them with a "salt". The concept of a salt being concatenated with the requests and offers is important for preventing dictionary attacks on hashes by either malicious Mediators or Trusted Anonymizers. The factors concerning the choice of ***hash functions*** and ***salt management scheme*** deserve much attention. The details of these choices are further discussed in subsection IV-C. This component of the security protocol achieves three objectives:

i. Privacy of the Publisher (P_1) is enforced by the Trusted Anonymizer (TA_2) eliminating direct interaction with a possibly curious Broker.
ii. Integrity of the messages is also enforced by encrypting messages using public-keys (KU_{TA_2} and KU_{B_2})
iii. Message confidentiality is achieved by sending hashed offers to entities participating in the protocol.

2) Subscriptions: This subsection presents the component of the protocol for securely requesting for content from the publish-subscribe system by the Subscribers (S_j).

$$S_1 \rightarrow TA_1 : [hash(request||salt)||TS_3]_{KU_{TA_1}} \quad (3)$$

$$TA_1 \rightarrow B_1 : [hashed\_request||TS_4]_{KU_{B_1}} \quad (4)$$

Where $hashed\_request = hash(request||salt)$

Again this component of the security protocol is used to enforce privacy for the subscriber and to protect message confidentiality and integrity using principles similar to those presented in subsection IV-B1 above.

3) Routing Tables: The Trusted Anonymizers (TAs) and Brokers (Bs) are supposed to keep secured routing tables which would be used to identify correct destinations of responses or feedback for participants.

The routing tables (I and II) created by the TAs are used to match Publishers or Subscribers with their respective hashed offers or requests. In case Brokers send feedback to the TA concerning a request which can be served (i.e. a match has been found), the concerned TA has to search in the routing table for the subscriber corresponding to a stored hash request for feedback to be sent.

Publisher	Hash Value
P_1	$[hash(offer\|\|salt)]_1$
P_2	$[hash(offer\|\|salt)]_2$
P_3	$[hash(offer\|\|salt)]_3$
...	...
P_i	$[hash(offer\|\|salt)]_i$

Table I
TAs' TABLE OF PUBLICATIONS

Subscriber	Hash Value
S_1	$[hash(request\|\|salt)]_1$
S_2	$[hash(request\|\|salt)]_2$
S_3	$[hash(request\|\|salt)]_3$
...	...
S_j	$[hash(request\|\|salt)]_j$

Table II
TAs' TABLE OF SUBSCRIPTIONS

The routing tables created by the Brokers (Bs) are used to match TAs to the hashed requests or offers they forwarded so that a route for feedback can always be established when a match is found. The brokers use the routing table to identify a given TA_l that sent them a particular hashed request or offer. Table III below shows a sample *Brokers' table of subscriptions*.

Trusted Anonymizer	Hash Value
TA_1	$[hash(request\|\|salt)]_1$
TA_2	$[hash(request\|\|salt)]_2$
TA_3	$[hash(request\|\|salt)]_3$
...	...
TA_l	$[hash(request\|\|salt)]_l$

Table III
BROKERS' TABLE OF SUBSCRIPTIONS

Likewise, the Broker has to store a table of publications which will be used to match TAs to corresponding requests from subscribers. Table IV below is a sample *Broker routing table* for publications. After building Tables III and IV, Brokers B_1 and B_2 will possess hashed values of requests and offers received from Trusted Anonymizers.

Trusted Anonymizer	Hash Value
TA_1	$[hash(offer\|\|salt)]_1$
TA_2	$[hash(offer\|\|salt)]_2$
TA_3	$[hash(offer\|\|salt)]_3$
...	...
TA_k	$[hash(offer\|\|salt)]_k$

Table IV
BROKERS' TABLE OF PUBLICATIONS

The presented sample tables do not have any optimization features such as indexing. This would of course be a relevant feature in a practical deployment. For example, tables III and IV could be combined by a Broker to create a match-making table consisting of a Primary-Key, Requesting-TA, Offering-TA and Hashed-Match columns.

The Brokers do not communicate directly with the Publishers and Subscribers, but they have the ability to perform matching functions between requests and offers received through the TAs. A ***request*** will be matched to an ***offer*** by a Broker if the hashes in protocol sessions (2) and (4) are equivalent. Whenever a Broker receives a hashed request from a TA, this hash is compared with contents of the publications table. In case a match is not found, the Broker (e.g. $B1$) would forward the request to another Broker (e.g. $B2$) for a match to be found.

After finding matching hashes, the Broker would then notify both the *"requesting"* TA_i and the *"offering"* TA_j about the success. The notification procedure is formally presented below.

$$B_1 \rightarrow TA_i : [TA_j||hashed\_offer||TS_5]_{KU_{TA_i}} \quad (5)$$

Where $hashed\_offer = hash(offer||salt)$

The message sent to the *"requesting"* TA_i consists of the identity of the *"offering"* TA_j, hashed value of the offer (obtained from the Broker's table of publications) and a timestamp to prevent replay attacks.

$$B_1 \rightarrow TA_j : [TA_i||hashed\_request||TS_6]_{KU_{TA_j}} \quad (6)$$

Where $hashed\_request = hash(request||salt)$

Similarly, the message sent to the *"offering"* TA_j, consists of the identity of the *"requesting"* TA_i, hashed value of the request and a timestamp to prevent replay attacks. The *"offering"* TA_j then notifies the Publisher (P_i) about the demand for their offer as shown in the protocol session below.

$$TA_j \rightarrow P_i : [hashed\_offer||TS_7]_{KU_{P_i}} \quad (7)$$

Where $hashed\_offer = hash(offer||salt)$

This part of the protocol also enforces the required security requirements of privacy, confidentiality and integrity for the publisher. The Broker will not be able to tell the identity of the Publisher, due to the indirect communication through a Trusted Anonymizer. Additionally, the TA_i also notifies the Subscriber (S_i) about the successful discovery of an offer to request.

$$TA_i \rightarrow S_i : [hashed\_request||TS_8]_{KU_{S_i}} \quad (8)$$

Where $hashed\_request = hash(request||salt)$

Now the Subscriber is required to generate a new unsigned public-key which will be used to enforce confidentiality and integrity requirements for delivery of the required publication. Presented below is the last component of the protocol.

$$S_i \rightarrow TA_i : [KU^*_{S_i}||TS_9]_{KU_{TA_i}} \quad (9)$$

Here $KU^*_{S_i}$ is the new unsigned public-key to be shared with the Publisher.

$$TA_i \rightarrow P_i : [KU^*_{S_i}||TS_{10}]_{KU_{P_i}} \quad (10)$$

Now, the Publisher (P_i) has an unsigned public-key for the Subscriber (S_i) received through the Trusted Anonymizer (TA_i) and can encrypt messages which can only be retrieved by S_i.

$$P_i \rightarrow TA_i : [[Object||TS^*]_{KU^*_{S_i}}||TS_{11}]_{KU_{TA_i}} \quad (11)$$

Finally, S_i can receive the secret object encrypted by the new unsigned public-key after TA_i has removed encryption layer in protocol session (11).

$$TA_i \rightarrow S_i : [Object||TS^*]_{KU^*_{S_i}} \quad (12)$$

In the next section we present a discussion of issues surrounding cryptographic hashing functions and their implications to the overall security of the proposed protocol in a Publish-Subscribe Mediated System.

C. Cryptographic Hash Functions

The guarantee of security requirements for the proposed system is highly dependent on the strengths of the cryptographic hash functions. The enumeration below presents key properties for hash functions previously suggested in literature [10]. These properties are a prerequisite for providing various security requirements to any system considering to use cryptographic hash functions:

i. Given a message (m) it should be easy to compute a hash value ($hash(m)$) from the message.
ii. Given $x = hash(m)$, it should be computationally infeasible to retrieve m from x.
iii. It is computationally difficult for an adversary to modify m without the $hash(m)$ changing.
iv. Given two messages (m_1 and m_2) where $m_1 \neq m_2$, it should be infeasible finding $hash(m_1)$ = $hash(m_2)$.

Given the hash function properties above, the proposed scheme would always generate unique hash values for various offers and requests respectively submitted by the

publishers and subscribers. Comparisons for equality between hash values for offers and request would also be possible since hash functions are deterministic and will always generate the same hash value for a particular input message.

If the hash function used in the system is poorly designed, then the hash values could be easily cracked by attackers. Hashing schemes such as MD5-crypt [11], [12], [13] and Bcrypt [14] use slow algorithms in-order to minimize the number of guesses which can be made by an attacker. Basically, the attacker would need much more time to crack a hash value with a slower algorithm as compared to other faster alternatives such as SHA-2 [15] family of hash functions. We are also aware of vigorous efforts for developing a more secure hash function (SHA-3) [16] conducted by the United States National Institute for Standards and Technology (NIST)[1].

The guiding principles for creating and choosing the salt are further explained in the next subsection (IV-D).

D. Choosing a Salt

The main objective of concatenating a salt with the messages (offers and requests) is to minimize possibilities of successful dictionary attacks. Adversaries in the network, curious brokers, subscribers and publishers may want to run dictionary attacks on messages with the objective of compromising confidentiality and privacy requirements.

Concatenating a salt to messages would tremendously reduce possibilities of dictionary attacks on the hashed values. An attacker would not succeed in deriving the plain text message from comparisons of hashes with those from words in standard dictionaries. The random bits attached to the messages would make such attacks very difficult in terms of storage requirements and computation. The salt has to be kept secret from system participants and any-other external entities so that attackers have limited information for guessing messages corresponding to hashed values.

The salt to be concatenated with the request or offer has to be carefully chosen in the global environment of the system. This is important for feasibility of matches between hashes of requests and offers to be preserved. Implementing a scheme in which randomized salts are appended to messages would definitely fail all comparisons between requests and offers.

E. Secure Salt Generation

The proposed scheme considers a tamper-proof software solution embedded on a smartcard for salt generation at the Publishers' and Subscribers' locations. The requirements for tamper-proof code are limited to only the hash generation component of the protocol. Such a software module is assumed to be small enought to fit on a smartcard. Since salt generation and hash value generation can be computationally inexpensive, deployment of smartcards to these two tasks can be practically supported by smartcard technologies[17] to achieve the desired security requirements.

A tamper-proof solution would guarantee that users do not have control over the process of creating a salt. Equally, the timestamps appended to the messages are system generated and the system has to rely on a trusted time server for synchronizing all clocks on participating nodes.

The next subsection (IV-F) provides more details on considerations for formatting messages in the secure publish-subscribe system.

F. Message Formatting

In order for comparisons in the system to make sense, message formats are supposed to be kept consistent. The format in which requests are concatenated with a salt and then encrypted, should be kept consistent with the subscriptions/offers.

This condition is particularly sensitive for the salt. If the salt concatenated with requests differs from that concatenated with the offers, then matching between offers and requests would be infeasible.

The global system environment is supposed to agree on a salt to be used for a particular period of time. It would be desirable to frequently change the salt to minimize chances of successful guesses. Such correct guesses would also increase chances of successful dictionary attacks on exchanged messages. Furthermore, the salt has to be synchronized with the publishers and subscribers via a secure channel.

A poorly chosen salting mechanisms will either render the security schemes unusable due to computationally incompatible offers and requests or the hash values will get easily broken by a curious broker and consequently users' privacy broken.

V. Solution Evaluation

This section discusses the weaknesses and strengths of the proposed scheme for providing confidentiality, integrity and privacy in publish-subscribe mediated virtual organizations.

Confidentiality of the salt generated at the Publishers' and Subscriber's terminals is dependent on the trust in the software responsible for generating the salt. The Publishers and Subscribers should not be able to gain access to the salt. This implies that memory protection for the generated salt should be provided. The choice of programming language for implementing this software is critical for the security of the system. Requirements such as type-safety and memory protection should be guaranteed by the implementation.

The Trusted Anonymizers (TAs) may turn against the Publishers and Subscribers to violate security requirements of privacy, confidentiality and integrity. The seemingly easy requirement to violate among those presented is privacy of Subscribers and Publishers by the TAs. The TAs are

[1] http://csrc.nist.gov/groups/ST/hash/sha-3/

978-1-4244-5494-5/10/$26.00 ©2010 IEEE

the first point of contact for the subscriber and publishers when they want to respectively buy and sell items via the Publish-Subscribe environment. It is practically infeasible to have message encryption without sharing keys with TAs. Encryption with the TAs public-key preserves message confidentiality and integrity in the presence of attacks and threats in the environment.

Attaching time bounds on validity of messages would not improve the security situation. Confidentiality and privacy can still be attacked using off-line techniques by a malicious Trusted Anonymizer. The broker has a chance to compromise message confidentiality when they copy the message and use available time and resources to decipher the message. However, successful completion of off-line attacks may occur long after the transaction was completed.

If the brokers succeed in cracking the hashes, they will not know the publisher of the item or in case of buying, they will not be able to tell who wishes to buy. However, successful off-line attacks by the TA would break both confidentiality and privacy of the message and Publisher-Subscriber respectively.

It is difficult to eliminate the use of hashing technique due to the requirement of blind-matching requests with offers. Encryption of messages would have provided a solution less susceptible to off-line attacks, but it would render blind-matching impossible.

VI. Conclusion

The proposed scheme is rigid as far as message formatting is concerned. Messages should be created in a standardized format for all participants so that matching requirements can be consistently met.

Although it is not our objective to prevent the TAs from knowing details of requests and offers, the hashed messages make this implicit security objective possible. The TAs will know identities of participants but will not have access to message details.

Performance of hash functions is guaranteed to be faster than other deterministic cryptographic techniques such as homomorphic encryption schemes. The hash generation procedure is assumed not to impose a performance penalty on the system due to efficiency requirements for hash algorithms as stated in the enumeration of subsection IV-C. The generated hash-value is far easier to encrypt than the original message due to the significant size differences between a message and its fingerprint. This presents the benefit of fast encryption for the resultant light-weight content.

We have presented a scheme which offers a practical solution for confidentiality, privacy and integrity requirements in publish-subscribe mediated virtual organizations. The compromises made (stated in section V) are also practically sound for the target requirements.

Acknowledgment

Many thanks to Erik Poll and John Quinn for the insightful reviews and feedback.

References

[1] R. Guttman, A. Moukas, and P. Maes, "Agent-mediated electronic commerce: A survey," *The Knowledge Engineering Review*, vol. 13, no. 02, pp. 147–159, 2001.

[2] P. Maes, R. Guttman, and A. Moukas, "Agents that buy and sell," *Communications of the ACM*, vol. 42, no. 3, p. 81, 1999.

[3] A. Shikfa, M. Onen, and R. Molva, "Privacy-Preserving Content-Based Publish/Subscribe Networks," in *Emerging Challenges for Security, Privacy and Trust: 24th Ifip Tc 11 International Information Security Conference, SEC 2009, Pafos, Cyprus, May 18-20, 2009, Proceedings*. Springer, 2009, p. 270.

[4] A. Shamir, "On the power of commutativity in cryptography," *Automata, Languages and Programming*, pp. 582–595.

[5] S. Pohlig and M. Hellman, "An improved algorithm for computing logarithms over GF (p) and its cryptographic significance (Corresp.)," *IEEE Transactions on information Theory*, vol. 24, no. 1, pp. 106–110, 1978.

[6] A. Carzaniga, D. Rosenblum, and A. Wolf, "Design and evaluation of a wide-area event notification service," *ACM Transactions on Computer Systems (TOCS)*, vol. 19, no. 3, pp. 332–383, 2001.

[7] B. Chandramouli, J. Yang, P. Agarwal, A. Yu, and Y. Zheng, "ProSem: Scalable wide-area publish/subscribe," in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. ACM, 2008, pp. 1315–1318.

[8] C. Raiciu and D. Rosenblum, "Enabling confidentiality in content-based publish/subscribe infrastructures," *Securecomm and Workshops, 2006*, pp. 1–11, 2006.

[9] N. Provos, "Preventing Privilege Escalation," in *Proceedings of the 12th USENIX Security Symposium*, 2003.

[10] I. Damgård, "A design principle for hash functions," in *Advances in CryptologyCRYPTO89 Proceedings*. Springer, pp. 416–427.

[11] S. Alexander, "improving security with homebrew system modifications," *USENIX; login*, vol. 29, no. 6, pp. 26–32, 2004.

[12] N. Provos and D. Mazieres, "MD5-crypt," *USENIX*, 1999.

[13] Provos and D. Mazieres, "A future-adaptable password scheme," in *Proceedings of the Annual USENIX Technical Conference*. Citeseer, 1999.

[14] N. Provos and D. Mazieres, "Bcrypt," *USENIX*, 1999.

[15] H. Gilbert and H. Handschuh, "Security analysis of SHA-256 and sisters," in *Selected Areas in Cryptography*. Springer, 2003, pp. 175–193.

[16] E. Fleischmann, C. Forler, and M. Gorski, "Classification of the SHA-3 Candidates," *International Association for Cryptologic Research (IACR) ePrint archive*.

[17] O. Kömmerling and M. Kuhn, "Design principles for tamper-resistant smartcard processors," in *Proceedings of the USENIX Workshop on Smartcard Technology on USENIX Workshop on Smartcard Technology*. USENIX Association, 1999, p. 2.

978-1-4244-5494-5/10/$26.00 ©2010 IEEE