Chapter 24

ANONYMOUS AND SECURE MULTICAST SUBSCRIPTION

Indrajit Ray and Indrakshi Ray

Abstract In this paper we propose a protocol by which a user can subscribe to an electronic service in an anonymous manner and then access that service also in an anonymous manner. The service provider does not know who accesses the service at any time. Moreover, if the subscriber so desires, the subscriber can prevent the service provider from linking two (or more) of its subscriptions to itself.

Keywords: Secure on-line subscription, secure multicast, anonymous protocols

1. Introduction

With the rapid growth of the Internet more and more opportunities are becoming available for obtaining goods and services online. Examples of such services are online news reports, stock information, financial information, company profiles, MP3 music, etc. These goods and services are delivered electronically to the consumer over the Internet. A number of these online services are based on paid subscription that imposes certain restrictions on the delivery of the objects – namely (1) consumers who have subscribed to an object should be able to access it, (2) no one other than the subscriber should have access to the object, and (3) the access should be allowed only during the period of validity for the subscription. These objectives can be met with a simple solution: encrypt the data object with the consumer's public key (or a secret key shared between the service provided and the customer) and transmit the encrypted data. However, if many customers subscribe to an object, then with this simple solution, an object has to be encrypted multiple times and transmitted multiple times. The simple solution is clearly, therefore, inefficient in terms of computation and bandwidth. An alternate solution is to broadcast the object in an encrypted manner and provide an access control mechanism by

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: [10.1007/978-0-387-35697-6_26](https://doi.org/10.1007/978-0-387-35697-6_26)

E. Gudes et al. (eds.), *Research Directions in Data and Applications Security*

© IFIP International Federation for Information Processing 2003

which only legitimate subscribers can access the object – in short, a solution based on secure multicasting at the application level.

Paid subscription services, additionally, raise several privacy issues for the consumer. First and foremost, payment for such services often requires the subscriber to provide considerable personal information to the provider which may be of a sensitive nature. Moreover, it is astonishingly easy for the provider to monitor the online activities of the subscriber; the results can be mined later to create a customer profile that can be used for various nefarious purposes.

The above factors motivate us to propose an anonymous subscription service that is based on secure multicasting. The service is based on a core set of protocols that we propose that allow any user to purchase a subscription service in an anonymous manner from a service provider and then access the service, also anonymously. Thus the service provider cannot determine a subscriber's true identity or link one subscription to a particular individual. Nor can the service provider create any customer profile that is of use. Moreover, for some additional complexity on the part of the subscriber, the subscriber can prevent the service provider from linking two subscriptions to itself.

Secure multicasting for online subscription services have been investigated previously by a number of researchers [1, 8, 11, 12]. All of them address the problem of key distribution and key management for secure multicasting and none address anonymity of the customer. Secure anonymous protocols have also been investigated separately [3–5, 9, 10, 13–16, 18]. However, we are not aware of any work that addresses both simultaneously in a unified framework.

The rest of the paper is organized as follows. Section 24.2 presents the theory of compatible keys that forms the basis of the anonymous multicast facility. Section 24.3 describes the subscription facility. We describe the anonymous subscription portion of the protocol in Section 24.3.1, the anonymous transfer and receipt of the subscription object in Section 24.3.2 and the subscription termination protocol in Section 24.3.3. Section 24.4 concludes our paper.

2. Theory of Compatible Keys

The secure multicasting protocol is based on the theory of compatible keys which we had developed earlier [17]. For lack of space we just provide the relevant definitions and the theorems.

Definition 1 The set of *messages* $\mathcal{M}$ is the set of non negative integers m that are less than an upper bound N , i.e.

$$\mathcal{M} = \{m | 0 \leq m < N\} \quad (1)$$

Definition 2 For positive integers a , b and N , we say a is *equivalent* to b , modulo N , denoted by $a \equiv b \pmod{N}$, if $a \bmod N = b \bmod N$.

Definition 3 For positive integers a, x, n and $n > 1$, if $\gcd(a, n) = 1$ and $a.x \equiv 1 \pmod{n}$, then x is referred to as the *multiplicative inverse of a modulo n* . Two integers a, b are said to be *relatively prime* if their only common divisor is 1, that is, $\gcd(a, b) = 1$. The integers $n_1, n_2, \dots, n_k$ are said to be *pairwise relatively prime*, if $\gcd(n_i, n_j) = 1$ for $i \neq j$.

Theorem 1 Euler's theorem states that for every a and N that are relatively prime, $a^{\phi(N)} \equiv 1 \pmod{N}$, where $\phi(N)$ is the Euler's totient function, defined as the number of integers that are less than N and are relatively prime to N .

Definition 4 A key K is defined to be the ordered pair $\langle e, N \rangle$, where N is a product of distinct primes, $N \geq M$ and e is relatively prime to $\phi(N)$; e is the *exponent* and N is the *base* of the key K .

Definition 5 The *encryption* of a message m with the key $K = \langle e, N \rangle$, denoted as $[m, K]$, is defined as

$$[m, \langle e, N \rangle] = m^e \pmod{N} \quad (2)$$

Definition 6 The *inverse* of a key $K = \langle e, N \rangle$, denoted by K^{-1} , is an ordered pair $\langle d, N \rangle$, satisfying $ed \equiv 1 \pmod{\phi(N)}$, where $\phi(N)$ is the Euler's totient function.

Theorem 2 For any message m .

$$[[m, K], K^{-1}] = [[m, K^{-1}], K] = m \quad (3)$$

where $K = \langle e, N \rangle$ and $K^{-1} = \langle d, N \rangle$.

Definition 7 Two keys $K_1 = \langle e_1, N_1 \rangle$ and $K_2 = \langle e_2, N_2 \rangle$ are said to be *compatible* if $e_1 = e_2$ and N_1 and N_2 are relatively prime.

Definition 8 If two keys $K_1 = \langle e, N_1 \rangle$ and $K_2 = \langle e, N_2 \rangle$ are compatible, then the *product* key, $K_1 \times K_2$, is defined as $\langle e, N_1 N_2 \rangle$.

Lemma 1 For positive integers a, N_1 and N_2 ,

$$(a \pmod{N_1 N_2}) \equiv a \pmod{N_1}$$

Theorem 3 For any two messages m and $\hat{m}$, such that $m, \hat{m} < N_1, N_2$,

$$[m, K_1 \times K_2] \equiv [\hat{m}, K_1] \pmod{N_1} \text{ if and only if } m = \hat{m} \quad (4)$$

$$[m, K_1 \times K_2] \equiv [\hat{m}, K_2] \pmod{N_2} \text{ if and only if } m = \hat{m} \quad (5)$$

where K_1 is the key $\langle e, N_1 \rangle$, K_2 is the key $\langle e, N_2 \rangle$ and $K_1 \times K_2$ is the product key $\langle e, N_1 N_2 \rangle$.

The more general case for the above theorem is as follows.

$$[m, K_1 \times K_2 \dots K_p] \equiv [\hat{m}, K_1] \pmod{N_1} \text{ if and only if } m = \hat{m} \quad (6)$$

$$[m, K_1 \times K_2 \dots K_p] \equiv [\hat{m}, K_1] \pmod{N_2} \text{ if and only if } m = \hat{m} \quad (7)$$

$$\vdots$$

$$[m, K_1 \times K_2 \dots K_p] \equiv [\hat{m}, K_1] \pmod{N_p} \text{ if and only if } m = \hat{m} \quad (8)$$

where $K_1 = \langle e, N_1 \rangle$, $K_2 = \langle e, N_2 \rangle$, etc. and $K_1 \times K_2 \dots K_p$ is the product key $\langle e, N_1 N_2 \dots, N_p \rangle$.

3. Anonymous Subscription Service

The following entities are the players involved in the protocol. parties:

Merchant - The merchant M , also referred to as the service provider, is responsible for selling the object to the customer and managing the payment portion.

Agent - The agent A is responsible for multicast delivery of the subscription materials to the subscribers. The agent periodically receives customer information from the merchant and at the beginning of every multicast, A uses this customer information to prepare the multicast message.

Customer - The customer subscribes to objects at the merchant, specifies the subscription period for each object it subscribes to, and pays for the proper amount for the subscription. All of these the customer does anonymously.

Customer's Bank and Merchant's Bank - The customer's bank maintain the customer's financial account while the merchant's bank does the same for the merchant's account. Payment for a subscription is in the form of a debit from the customer's account and a credit of the same amount to the merchant's account.

Note that the merchant and the agent may be two separate entities or the same entity. If they are two different entities we allow collusion between them. In other words, whatever information one of them is able to get about the customer it shares with the other party. Same is the case for the two banks. This protocol guarantees the anonymity of the customer under all possible collusion between the merchant, the agent and the financial institutions.

The complete protocol can be divided into three phase – (i) the anonymous subscription and payment phase, (ii) the object transmission and receipt phase

and (iii) the subscription termination phase. All communication between the customer and the merchant take place over an anonymizing channel based on Chaumian mixes [2]. Both forward path and return path for a message can be specified in an anonymous manner over such channels. Unless all the mix agents along a path collude the anonymity of the sender is maintained.

For this discussion the following symbols are used.

Table 1. Symbols used in protocol description.

Symbol	Interpretation
C, M, A	Customer, Merchant and Agent
CB, MB	Customer's bank and Merchant's bank
X_{acct}	X 's account with its bank
X_{pub}, X_{prv}	An entity X 's public and private key pair
ψ_X	unique nonce for entity X
B_{cpub}, B_{cprv}	Common public and private key for banks
$A \xrightarrow{a} B: Y$	A sends message Y to B over an anonymous channel
$A \Rightarrow B: Y$	A sends Y to B over an ordinary channel
$CC(Y)$	Cryptographic message digest of Y
C'	pseudo identifier of customer C used in one single transaction
K_i, K_i^{-1}	Compatible keys and their inverses
$\mathcal{P}$	Payment token used for paying for a subscription
$\mathcal{R}$	refund token used for refunding money on a subscription

3.1 Anonymous Subscription and Payment Phase

The process begins when the customer contacts the merchant to initiate a new subscription. For simplicity we will assume that the customer subscribes to only one object at any subscription phase. If the customer wants to subscribe to more than one object with the same merchant, the customer initiates different subscription phases with new pseudo identifiers. Two other parties besides the customer and the merchant are involved in this phase. They are the customer's bank, CB , and the merchant's bank, MB . The respective public/private key pairs are CB_{pub}/CB_{prv} and MB_{pub}/MB_{prv} . We also assume that all financial institutions share a common public/private key pair (B_{cpub}, B_{cprv}) . If a bank signs a message using the key B_{cprv} , then the bank receiving the message will be able to verify that the message has been signed by a bank but will not be able to identify, on its own, which specific bank signed the message.

This idea is similar to the concept of a group signature scheme [6, 7]. A group signature is publicly verifiable but it maintains the anonymity of the signer.

The subscription phase proceeds as follows:

Message 1

$$C' \xRightarrow{a} M: [Subscribe(O_i), M_{pub}], [CC(Subscribe(O_i)), C'_{iprv}]$$

$Subscribe(O_i)$ is the subscription request message. It is sent encrypted with the merchant's public key and contains the following items: (i) the customer's pseudo identifier C' , (ii) the customer's one time public key, C'_{ipub} , (iii) the identifier of the object that the customer wants to subscribe to, D_i , (iv) duration of the subscription, D , and (v) a nonce for the customer, $\psi_{C'}$.

The merchant decrypts the subscription message with its public key and retrieves the customer's pseudo identifier and one time public key. The merchant authenticates the subscription request by verifying the signature on the checksum. It then creates a new subscription record for the customer with identifier C' for object D_i for the period D . The merchant flags the record as payment pending.

Message 2

$$M \xRightarrow{a} C': [([M_{acct}, MB_{pub}], D_i, val), C'_{ipub}], \\ [CC([M_{acct}, MB_{pub}], val), M_{prv}]$$

The merchant next requests payment from the customer for the subscription. The payment will be a credit to the merchant's account, M_{acct} , with its bank for an amount, val , which is the cost of the subscription for the desired period. To make the request the merchant prepares a message with its account number encrypted under the merchant's bank's public key, $[M_{acct}, MB_{pub}]$. In this manner, no entity other than the merchant's bank gets to know the encrypted account information. This encrypted account information is sent to the customer C' .

Message 3

$$C \Rightarrow CB: [MTI, C_{prv}]$$

On receiving the payment request from the merchant, the customer submits a money transfer instruction to the customer's bank. The money transfer instruction, MTI , is a message consisting of the following information: (i) the amount of money, val , that is being transferred; this is the same as the cost of the subscription; (ii) the merchant's encrypted account that is to be credited, $[M_{acct}, MB_{pub}]$; (iii) the customer's account with this bank that is to be debited, C_{acct} ; and (iv) the signed cryptographic checksum of $([M_{acct}, MB_{pub}], val)$ received from the merchant – $[CC([M_{acct}, MB_{pub}], val), M_{prv}]$.

Message 4

$$CB \Rightarrow C: [[\mathcal{P}, B_{cprv}], C_{pub}], [CC([\mathcal{P}, B_{cprv}]), CB_{prv}]$$
OR

$$CB \Rightarrow C: [Failure, C_{pub}]$$

On receiving the money transfer instruction from the customer, CB first checks if the customer has instructed the correct amount to be transferred. If the amount is not correct the bank sends a failure message to the customer, $Failure$. Otherwise the bank next checks to see if C 's account has enough money. If it does, the bank sends the payment token, $\mathcal{P}$, to the customer; otherwise it sends the failure message.

The payment token, $\mathcal{P}$, created by the customer's bank contains the following information: (i) the amount, val , that is being credited, (ii) the encrypted account, $[M_{acct}, MB_{pub}]$, that is to be credited, and (iii) a random number, r , to prevent replay attacks.

The payment token, $\mathcal{P}$ is signed by the banks' common private key, B_{cprv} . This is in order that the merchant's bank can authenticate that the payment token has been generated by some bank and that it is okay to honour the payment token. Note that the payment token does not contain any identifying information for the customer's bank or the customer. Thus the merchant's bank cannot collude with the customer's bank to trace the customer.

Message 5

$$C' \xRightarrow{a} M: [[[\mathcal{P}, B_{cprv}], C', D_i], M_{pub}], [CC([\mathcal{P}, B_{cprv}]), C'_{iprv}]$$
OR

$$C' \xRightarrow{a} M: [Abort, C'_{iprv}]$$

Depending on whether the customer is interested in successfully completing the transaction, the customer sends either the signed and encrypted payment token that it got from CB to the merchant or an abort message. The payment token has to be sent along with the customer's pseudo identifier and the identifier of the object that the customer is subscribing to. This is required because without these information the merchant has no way of determining which customer's record is to be credited with the payment for the subscription.

Message 6

$$M \Rightarrow MB: [[\mathcal{P}, B_{cprv}], MB_{pub}], [CC([\mathcal{P}, B_{cprv}]), M_{prv}]$$

If M receives an abort message from the customer then the merchant aborts the subscription process. Otherwise M forwards the payment token to its bank, MB after encrypting it with MB 's public key.

Message 7

$$MB \Rightarrow M: [ack, MB_{prv}]$$

MB after receiving the payment token authenticates it. Then it decrypts the payment token using its private key to get the account number that is to be credited and the amount that is to be credited to this account. Once done crediting M 's account, it sends an acknowledgment message back to M .

Message 8

$$M \xrightarrow{a} C': [K_i^{-1}, C'_{ipub}]$$

AND

$$M \Rightarrow A: [(C', D_i, K_i), A_{pub}]$$

When the merchant receives the acknowledgment message from its bank, MB , it enters a subscription paid for information against the customer's record. Next it generates the customer's subscription encryption/decryption key pair K_i/K_i^{-1} according to Algorithm 1. The merchant records the encryption key K_i against the customer's subscription record and sends it to the agent, A , together with the customer's pseudo identifier and the identifier of the object that this customer has subscribed to. It also sends the decryption key, K_i^{-1} , anonymously to the customer.

Algorithm 1 *Encryption/Decryption Key Generation for Customer C_i*

Input: (i) $\mathbf{K}$ – List of encryption keys that have already been assigned to existing customers, (ii) e – a large exponent chosen by the merchant which is common to all the customers.

Output: (i) $\langle K_i, K_i^{-1} \rangle$ – Encryption Key K_i /Decryption Key K_i^{-1} for customer C_i .

Procedure GenerateCustomerKeys($\mathbf{K}, e$)

begin

$factors := 1$

for each $K_j = \langle e, N_j \rangle$ **in** $\mathbf{K}$ **do**

$factors := factors * N_j$

 choose an N_i that is relatively prime to $factors$

$K_i := \langle e, N_i \rangle$

$K_i^{-1} := \langle d_i, N_i \rangle$ */* where $ed_i \equiv 1 \pmod{\phi(N_i)}$ */*

end

The agent, A , maintains a list L_j for each data object O_j . Corresponding to each customer with (pseudo) identifier C_i who has subscribed to the object O_j there is a record R_{kj} in the list L_j . On receiving a customer's information (for example (C', D_j, K_i) as in Message 8 above) from the merchant, it inserts a new record $R_{ij} = \langle C_i, D_j, K_i, (T_{ijk}, T_{ijl}) \rangle$ as specified by Algorithm 2 below.

Algorithm 2 *Insert Customer C_i 's Subscription*

Input: (i) C_i – the (pseudo) identity of the customer; (ii) $\mathbf{D}$ – the set of data objects that the customer is subscribing to. If the customer is just subscribing to only one object this set contains just one element D_j ; (iii) $\mathbf{T}$ – the set of time intervals describing the period of subscription for each data object in $\mathbf{D}$. Again, if the customer is subscribing to just one item during this subscription it will be the tuple (T_{ijk}, T_{ijl}) ; and (iv) $\mathbf{L}$ – A set of lists corresponding to the data objects in $\mathbf{D}$.

Output: (i) $\mathbf{L}$ – The set of updated lists. Each list L_j will be updated by inserting the record R_{ij} .

Procedure InsertCustomerSubscription $((C_i, \mathbf{D}, \mathbf{T}, \mathbf{L}))$

begin

for each data object D_j in $\mathbf{D}$ **do**

 get $\langle T_{ijp}, T_{ijq} \rangle$ from the set $\mathbf{T}$

if there is no record R_{ij} in list L_j

 insert record $R_{ij} = \langle C_i, K_i, (T_{ijp}, T_{ijq}) \rangle$ in L_j

end

This completes the subscription phase.

3.2 Object Transmission and Receipt Phase

In this phase the agent identifies a suitable encryption key, encrypts the subscription objects with the key and multicasts them to a previously announced (in a public place) multicast group address. The customer tunes in to the multicast message to retrieve the data it has subscribed to.

The multicast key that is used by the agent may be the same key as used in the previous multicast session (if the multicast group has not changed) or it may be a new key. The choice of keys guarantees that only those customers that have valid subscriptions to this object d at the current time, are the only ones that can access d . The key generation and encryption is based on the theory of compatible keys described in Section 24.2.

Figure 1. Structure of the broadcast message.

3.2.1 Agent's action to multicast data. At each multicast time t , the following activities are performed in sequence: (i) The agent finds out the list of data objects that must be transmitted at this time. (ii) For each of these objects, the agent is responsible for generating the appropriate object encryption key using the Algorithm 3. (iii) The data objects are encrypted with the correct key. (iv) Each encrypted data object corresponds to a data block of the multicast message. (v) The multicast index block is created and prepended to the message. For each object that is being multicast, a pointer to the data block for that object is given in the multicast index block. The structure of the multicast message is given in Figure 1. (vi) The multicast message is transmitted to the corresponding multicast group.

Algorithm 3 Object Encryption Key Generation

Input: (i) $\mathbf{D}$ – the set of data objects to be multicasted, (ii) $\mathbf{L}$ – the set of lists corresponding to all the objects. (iii) t – The broadcast time t .

Output: (i) The set of keys used to encrypt the data objects.

Procedure ObjectEncryptionKeyGeneration($\mathbf{D}, \mathbf{L}, t$)

begin

for each data object $D_j \in \mathbf{D}$ **do**

$factor_j := 1$

for each list L_j **in** $\mathbf{L}$ **do**

for each record R_{ij} **in** L_j **do**

if t **is within the interval** (T_{ijk}, T_{ijl}) **then**

$factor_j = factor_j * N_i$ /* N_i is the modulus of Key K_i */

if $(factor_j \neq 1)$ **then**

$Key_{D_j} := \langle e, factor_j \rangle$

else no key to generate for data D_j .

end

3.2.2 Customer action for retrieving data. In order to retrieve the data object during the subscription interval, the customer performs the following steps in order: (i) The customer joins the advertised multicast group and listens to the multicast message. (ii) The customer reads the multicast index to find out which data block(s) contain the data object(s), the customer is interested in. (iii) The customer then gets the encrypted data object(s) from the respective data block(s). (iv) To decrypt each data object, the customer C_i uses his decryption key K_i^{-1} .

3.3 Subscription Termination Phase

The third and final phase of the protocol deals with subscription termination. Subscription will terminate at the end of the subscription period unless renewed by the customer. Subscription can also terminate when the customer chooses to do so. In the former case there is no refund of subscription dues to the customer. However, in the latter case the customer is refunded for the unused portion of the subscription. The two subscription termination protocols are thus different.

3.3.1 Subscription termination at end of subscription period. Recall from Section 24.3.2.1 that the agent determines the appropriate multicast encryption key at the start of every multicast time for the object d . If the group has not changed since last time the key remains unchanged. If the group of customers subscribing to d changes then the agent generates a new key. As the agent generates the encryption key based on the current set of subscribing customers, no change in our protocol is required to handle standard subscription termination.

3.3.2 Subscription termination by customer. A customer can choose to terminate a subscription at any time. To do this, the customer contacts the merchant anonymously, using the same pseudo identifier that it used during the subscription process. The merchant changes the subscription time for the object the customer was subscribing to. It then computes an appropriate refund and sends it to the customer. The refund is via an electronic fund transfer from the merchant's account to the customer's account. Anonymity of the customer needs to be preserved in this transaction as well as unlinkability. The subscription termination phase is described below.

Message 1

$C' \xrightarrow{a} M: [Unsubscribe(O_i), M_{pub}], [CC(Unsubscribe(O_i)), C'_{iprv}]$

$Unsubscribe(O_i)$ is the customer's unsubscription request message. It is sent encrypted with the merchant's public key and contains the following items: (i) the customer's pseudo identifier C' ; this should be the

same as the one used by the customer for subscription purposes, (ii) the customer's one time public key, C'_{ipub} ; this should also be the same as the one used during the subscription phase, (iii) the identifier of the object that the customer wants to unsubscribe from, D_i , (iv) the customer's encrypted account information with the customer's bank, $[C_{acct}, CB_{pub}]$, and (v) a nonce for the customer, $\psi_{C'}$.

Message 2

$M \Rightarrow A: \text{Remove}(C')$

AND

$M \Rightarrow MB: [RI, M_{prv}]$

On receiving the *Unsubscribe* message from the customer, the merchant modifies the customer's subscription information and sends a message to the agent to do the same. The agent removes the customer's information from its database – given by Algorithm 4.

Algorithm 4 *Modify Customer C_i 's Subscription*

Input: (i) C_i – the identity of the customer. (ii) $\mathbf{D}$ – the set of data objects that the customer is subscribing to. (iii) $\mathbf{T}$ – the set of time intervals describing the period of subscription for each data object. (iv) $\mathbf{L}$ – A set of lists corresponding to the data objects in $\mathbf{D}$.

Output: (i) $\mathbf{L}$ – The set of updated lists. Each list L_j will be updated by modifying the record R_{ij} .

Procedure ModifyCustomerSubscription $((C_i, \mathbf{D}, \mathbf{T}, \mathbf{L}))$

begin

for each data object D_j in $\mathbf{D}$ **do**

 get $\langle T_{ijp}, T_{ijq} \rangle$ from the set $\mathbf{T}$

if there is record R_{ij} in list L_j

 delete existing record R_{ij}

end

The merchant also calculates the amount of money that has to be refunded to the customer and sends a money refund instruction, RI , to its bank, MB . The money refund instruction RI is signed by the merchant and contains the following information: (i) the amount of the refund, amt , (ii) the merchant's account with this bank, M_{acct} , that is to be debited for the amount amt , and (iii) the encrypted account that is to be credited, $[C_{acct}, CB_{pub}]$.

Message 3

$$MB \Rightarrow M: [[\mathcal{R}, B_{cprv}], M_{pub}], [CC([\mathcal{R}, B_{cprv}]), MB_{prv}]$$

The merchant's bank debits the merchant's account (we assume that the merchant's bank has the amount) and creates a refund token, $\mathcal{R}$. The refund token contains the following information: (i) the amount amt that is being refunded (ii) the encrypted account, $[C_{acct}, CB_{pub}]$, that is to be credited with this amount, and (iii) a random number, r to prevent replay attacks.

The refund token, $\mathcal{R}$, is signed by the banks' common private key, B_{prv} . Thus the customer's bank can authenticate that the refund token has been generated by some other bank.

Message 4

$$M \xRightarrow{a} C': [[\mathcal{R}, B_{cprv}], C'_{ipub}], [CC([\mathcal{R}, B_{cprv}]), M_{prv}]$$

The merchant performs the refund by sending anonymously the refund token from its bank to the customer. The customer later deposits the refund token with its bank to get credit for the unused portion of the subscription.

4. Conclusions and Future Work

In this paper we describe an online subscription facility that allows a customer to subscribe to online objects in an anonymous manner and then receive these objects, also anonymously. The protocol guarantees that despite collusion between all players involved in the protocol other than the customer, the true identity of the customer can never be linked to a particular transaction. Moreover, the protocol does not require key distribution for every multicast session or group changes. If the group of customers subscribing to an object changes, the existing group members are not aware of this change. This further protects the customer's anonymity.

Much work remains to be done. In this work we assumed that the customer and the merchant behave fairly; that is none of these players behave in a manner that can cause a loss to the other. We have not addressed issues related to unfair behavior. Neither have we described how the protocol handles failures. These will be addressed in a separate work as will be addressed performance issues.

References

- [1] A. Celik and A. Datta, A scalable approach for subscription-based information commerce, *Proceedings of the Second International Workshop on Advanced Issues in E-Commerce and Web-Based Information Systems*, 2000.

- [2] D. Chaum, Untraceable electronic mail: Return addresses and digital pseudonyms, *Communications of the ACM*, vol. 24(2), pp. 84-90, 1981.
- [3] D. Chaum, Blind signatures for untraceable payments, in A. Sherman, R. Rivest and D. Chaum (eds.), *CRYPTO'82 Proceedings*, Plenum, pp. 199-203, 1983.
- [4] D. Chaum, Blind signature system, in D. Chaum (ed.), *CRYPTO'83 Proceedings*, Plenum, pp. 153-164, 1984.
- [5] D. Chaum, Security without identification: Transaction systems to make Big Brother obsolete, *Communications of the ACM*, vol. 28(10), pp. 1030-1044, 1985.
- [6] D. Chaum and E. van Heijst, Group signatures, *Advances in Cryptology – EUROCRYPT'91 (LNCS 547)*, Springer Verlag, pp. 257-265, 1991.
- [7] L. Chen and T. Pederson, Group signatures, *Advances in Cryptology – EUROCRYPT'94 (LNCS 950)*, Springer Verlag, pp. 171-181, 1995.
- [8] G. Chiou and W. Chen, Secure broadcasting using the secure lock, *IEEE Transactions on Software Engineering*, vol. 15(8), 1989.
- [9] A. Fiat, D. Chaum and M. Naor, Untraceable electronic cash, *Advances in Cryptology – CRYPTO'88 Proceedings (LNCS 403)*, Springer Verlag, pp. 319-327, 1988.
- [10] Y. Frankel, A. Chan and Y. Tsiounis, Easy come, easy go – Divisible cash, *Advances in Cryptology – EUROCRYPT'98 Proceedings (LNCS 1403)*, Springer-Verlag, 1998.
- [11] I. Gopal and J. Jaffe, Point-to-multipoint communication over broadcast links, *IEEE Transactions on Communications*, vol. 32(9), pp. 1034-1044, 1984.
- [12] I. Ingemarsson, D. Tang and C.K. Wong, A conference key distribution system, *IEEE Transactions on Information Theory*, vol. 28(5), pp. 714-720, 1982.
- [13] S. Low, N. Maxemchuk and S. Paul, Anonymous credit cards, *Proceedings of the Second ACM Conference on Computer and Communications Security*, pp. 108-117, 1994.
- [14] S. Low, N. Maxemchuk and S. Paul, Anonymous credit cards and their collusion analysis, *IEEE/ACM Transactions on Networking*, vol. 4(6), 1996.
- [15] G. Medvinsky and B. Neuman, Netcash: A design for practical electronic currency on the Internet, *Proceedings of the First ACM Conference on Computer and Communications Security*, pp. 102-106, 1993.
- [16] D. Pointcheval and J. Stern, Provably secure blind signature scheme, in M. Rhee and K. Kim (eds.), *Advances in Cryptology – ASIACRYPT'96 Proceedings (LNCS 1163)*, Springer Verlag, pp. 252-265, 1996.

- [17] I. Ray, I. Ray and N. Narasimhamurthi, A fair-exchange protocol with automated dispute resolution, Technical Report, University of Michigan at Dearborn, 2000.
- [18] S. Stubblebine, P. Syverson and D. Goldschlag, Unlinkable serial transaction, in R. Hirschfield (ed.), *Proceedings of the International Conference on Financial Cryptography (LNCS 1318)*, Springer Verlag, pp. 39-55, 1997.