Self-Healing Group Key Distribution*

Muhammad Junaid Bohio and Ali Miri
(Corresponding author: Ali Miri)

School of Information Technology and Engineering, University of Ottawa
Ottawa, ON, K1N 6N5, Canada (Email: {mbohio, samiri}@site.uottawa.ca)

(Received June 12, 2005; revised and accepted July 4, 2005)

Abstract

In this paper we propose the self-healing feature for group key distribution through Subset Difference (SD) method proposed by D. Naor et al. The subset difference method is one of the efficient proposals for group key distribution, however, recently a polynomial based solution for key distribution was proposed by D. Liu et al., which has the similar message size but also provides self-healing feature. We compare the two schemes and show that, SD has better performance in key recovery operation and is secure against the collusion of any number of revoked users. By incorporating the feature for self-healing to SD, it will be a more practical solution for the networks where packet loss is common. In addition to the self-healing feature, we also present some optimization techniques to reduce the overhead caused by the self-healing capability. Finally, we discuss the idea of mutual healing and mention certain requirements for that method for key recovery.

Keywords: Group key distribution, key recovery, self-healing property, subset difference method, recursive hash chains

1 Introduction

In network communication, messages that are intended for more than one user should be delivered through multicasting to save network resources. There is a need to control access to the content of such messages, and to restrict them to authorized users only. However, in practice, the group of authorized users can vary periodically. Conventional solutions use group keys to implement such control, but face problems in dealing with changes in the group of authorized users. As the group size grows larger, scalability becomes an issue and more efficient protocols are required to provide a desired level of security without trading away performance.

In wireless mobile networks and networks with heavy traffic such as the Internet, packet loss, which can result in loss of crucial information by users, is common. For example, in a key revocation process in such networks, it is very likely that users might not receive their updated session keys. Thus, it is very important to ensure reliable transmission of updated session keys to the authorized users. One naive approach to solve this problem is to expect, those users who did not receive the message to contact the server to get the required key. However, in the case of very large networks, if a significant number of users approaches the server to get the lost session keys, the bandwidth overhead will increase and it will be difficult for the server to manage such requests. A more practical approach is to ensure self-healing in the broadcast messages of the updated session keys. This can reduce the communication overhead and can simplify key management for the server. One major problem in using a self-healing technique is that the broadcast message size will increase in proportion to the number of session keys which are made recoverable through self-healing. Also, the addition of new members or rejoining of old members requires the protection of previous session keys from unauthorized users. This increases computational requirements and the message size. Thus, optimization techniques are required to control the message size and computational requirements of any self-healing scheme.

In this paper, we propose a self-healing technique for the SD key distribution scheme and will also give optimization methods that can be used to minimize message size and computational requirements.

1.1 Related Work

The first significant solution for broadcast encryption and revocation can be traced back to the broadcast encryption scheme by Fiat and Naor [3]. Their solution allows the key distribution center to broadcast messages to authorized users while removing a subgroup of users, and such revocation is secure against the collusion of a group of, say k, users. Another key distribution and revocation scheme proposed in [11], is based on the polynomial interpola-

*The material in this paper was presented in part at the 3rd IEEE International Symposium on Network Computing and Applications, Boston, MA, July 2004

tion technique, which was also used in [7] and [1] for the revocation of users. The scheme in [11] has self-healing capability and keys are distributed to users in the group using a polynomial covered with a masking polynomial. In order to ensure self-healing, partial shares of other session keys are included in the message of each session key. It is sufficient for a user who has missed the key for some session j to have at least one message from the previous update messages before session j, and any one message after session j. Information from the two messages will be used to compute the key for session j. For this recovery, a user does not have to contact the group manager, rather he only has to listen to the traffic which should enable him to get such a key. However, the size of the broadcast message in this scheme increases exponentially as the number of revoked users increases, and its security against the collusion of revoked users is associated to the degree of polynomial.

In [5], a self-healing technique similar to that in [11] is used; however, the key distribution technique is different and more efficient. It can be shown that, by excluding the self-healing feature, its broadcast message size is a linear function of the number of revoked users and is very close to the message size of the SD scheme proposed in [6], which will be discussed in detail later in this section. The key recovery operation in [5] is more expensive than in [6]. The SD scheme is secure against the collusion of any number of revoked users, whereas [5] is limited to the degree of the polynomial being used. This will imply that as higher degree polynomials are used, the computation becomes more expensive. However, the scheme in [5] has the self-healing capability in addition to the revocation process.

In [4], a Layered Subset Difference (LSD) scheme is proposed that reduces the initial key distribution requirement by almost a square root factor over the basic SD scheme. It uses unions of small subsets and reduces such collections of subsets, and thereby reduces the number of initial keys given to users. The authors in [4] also discuss more complicated types of privileged sets defined by nested inclusion and exclusion conditions. The storage requirement in both SD and LSD is further reduced in [2] with an increase in computational overhead.

The SD protocol is based on the idea of a logical key hierarchy (LKH) proposed in [12] and [13]. The LKH protocol uses key hierarchy instead of the hierarchy of group security agents. It is also a re-keying protocol like SD, but revokes a single user at a time and updates the keys of all remaining users. In [15], the authors use periodic batch rekeying (instead of rekeying after each join or leave as in LKH) to improve scalability. For reliable key transport, forward error correction (FEC) coding is used in [15]. C. Wong *et al.* in [14], use UDP over IP multicast for efficient rekey message delivery for large groups. In order to avoid the server being a bottleneck in the registration process, multiple registrars are used to offload client registration from the server. As the UDP protocol is unreliable, packet loss is possible. If the message is larger than the message transmission unit (MTU), IP fragmentation occurs and the message is broken into several IP packets; this increases the message loss probability. To reduce the message loss probability, the authors use the FEC technique. In the case of the message loss, re-synchronization of the lost keys to the requester is done by using pairwise encryption. In [8], the re-synchronization process is non-interactive due to the use of hints for updated keys attached to subsequent data packets. Such hints can be as small as half of the actual key size and users are required to use it to compute the key. The solutions in [8, 12, 13, 14, 15] are stateful i.e. if the keys in the tree are updated, an offline member will remain separated from the group and cannot rejoin without assistance from the group manager.

In [9], Pinkas proposes two solutions to reduce broadcast message size. In his first scheme, a pseudorandom function is used and keys are generated using two seeds. A user remaining offline for a certain period will be given appropriate seeds to derive keys for that period. Since keys are not independent of each other, this scheme is vulnerable to the collusion attack by two users who can generate keys for the unauthorized sessions that are in between their authorized sessions. In the second solution, the session keys are derived from a pseudorandom function on the pattern of a binary tree. The leaves of the tree are considered to be the session keys. Any user remaining offline for a period of t consecutive sessions will be given the key of the root of the subtree containing all such sessions. The collusion of any number of users will not allow them to compute keys for the sessions from which they were excluded. This solution requires requests from the users (who have lost their updates) to the server to resend the lost keys. Our focus is to avoid such requests and thereby reduce the possibility of the server being a bottleneck. We are also presenting a self-healing solution, which ensures that no user can use this capability to compute keys for the sessions for which the user was not authorized.

Next, we will discuss some of the details of the Subset Difference method proposed in [6].

2 Preliminary Information

In conventional tree based schemes, a user is only considered to be the member of subtrees rooted at its ancestors, whereas in the SD scheme it can also be covered in subtrees not rooted at its ancestors. It was shown that this approach can reduce the overall number of constructed subsets needed to deliver the new group key while revoking certain users. The reduced number of such subsets eventually reduces the message size. A subset in SD is described below.

2.1 The Subset Description

A subset $S_{i,j}$ is represented by two nodes $(v_i,\ v_j)$, such that v_i is the ancestor of v_j and the subset $S_{i,j}$ consists of the node v_i and its descendants excluding the node v_j and its descendants. (Please refer to Figure 1.)

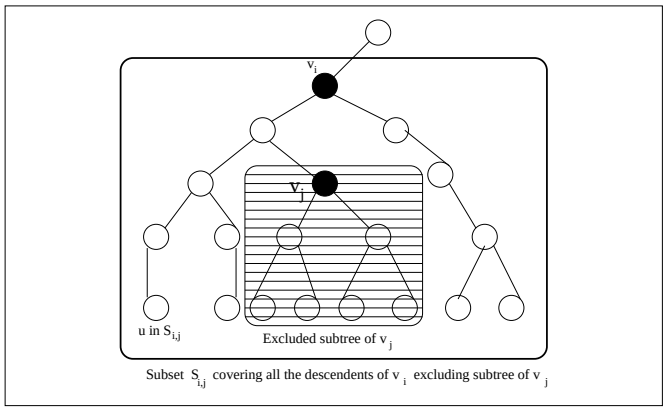

Figure 1: The subset difference method: subset $S_{i,j}$

In this scheme, at an initialization step, a user is given the labels of those nodes that are adjacent to its ancestors but not the labels of its ancestors. A user can derive keys for other descendants from such labels. When a new group key is to be delivered, it is encrypted in the subset key in order to protect it from revoked users. A subset key is the key of the first common ancestor of revoked users that is derived from the label of the root of that subset. Thus, when some users are to be revoked from the group, subsets are constructed through the SD algorithm and a new group key is delivered by encrypting with each subset key for non-revoked users.

2.2 The Subset Difference (SD) Method

For a given set $\mathcal{R}$ of revoked users, the non-revoked users of $\mathcal{N}\backslash\mathcal{R}$ are partitioned into subsets $S_{i_1,j_1}, S_{i_2,j_2}, \cdots, S_{i_m,j_m}$ as follows:
- A Steiner tree $ST(\mathcal{R})$ is constructed of revoked users and the root.
- The subset collection is obtained iteratively, maintaining a tree T which is a subtree of $ST(\mathcal{R})$. T is initially equal to $ST(\mathcal{R})$ and then nodes are removed iteratively from T through the following steps until it is reduced to a single node.

Step 1 : This step will find nodes with the following properties:

(a) IF: there is more than one leaf in the tree T
then: find two leaves v_i and v_j (that are to be revoked) in the tree T, such that the least-common-ancestor v of v_i and v_j does not contain any other leaf of T (i.e. no more than two members of $\mathcal{R}$). The node v would be of outdegree 2, i.e. it has two members of $\mathcal{R}$ on both its branches. Let v_l and v_k be the two children of v such that v_i is the descendant of v_l, and v_j is a descendant of v_k.
(b) else IF: there is only one leaf to revoke or after all iterations only one leaf is left in the tree T
then: the main root will be considered as $v = v_l = v_k$ and the node to be revoked can be considered either v_i or v_j.
Step 2 : IF: $v_l \not\equiv v_i$ (i.e. the two nodes are different)
then: add subset $S_{l,i}$ to the collection of subsets,
similarly, IF: $v_k \not\equiv v_j$
then: add $S_{k,j}$ to the collection of subsets.

Step 3 : Remove from T all the descendants of v and make v a (revoked) leaf $\in T$.
An example of making subsets based on Subset Difference Method is given in Figure 2.

The maximum number of subsets that can occur through this method is $2r-1$, where r is the number of revoked users. We observed that such maximum subsets occur when revoked users are equally scattered throughout the tree. The example in Figure 2 illustrates the case where $N = 16$ and $r = 4$, and for which we get the maximum number of subsets (worst case) i. e. $2r - 1 = 2 * 4 - 1 = 7$.

Later in this paper, we will propose a self-healing feature (that allows nodes to recover lost keys) for the SD method to make it more suitable for networks such as mobile ad hoc networks in which packet loss is common. In the next subsection, we will show that (with the addition of self-healing feature) the performance of the SD method is superior to the polynomial based solution.

2.3 Comparing the Schemes in [5] and [6]

The broadcast message size for both the SD scheme in [6] and the self-healing revocation scheme in [5] is $O(r)$, where r is the number of revoked users. More precisely, in SD the maximum number of encryptions of new session keys can be $2r-1$, and it will also contain a similar number of indices of subsets, so the total elements of $\mathbb{F}_q$ in the message are $4r-2$, where $\mathbb{F}_q$ is the base finite field. For the scheme in [5], by excluding the parameters for self-healing and considering only those required for key distribution and revocation, there are $4r + 2$ elements of $\mathbb{F}_q$. Hence, the message size in both cases is $O(r)$.

In the SD method, key recovery requires $\log(\log N)$ search operations by a user for his/her index, where N is the number of users in the group. The computation of the subset key requires invoking a pseudo random function G at most $\log N$ times. Finally, a single decryption will recover the new session key. This is in comparison to the second scheme in [5], where each user will have to first compute the polynomial $g_j(x)$, which will require the multiplication of r terms of the form $(x - u_i)$. The evaluation of the resulting polynomial $g_j(x)$ at the user index will require r exponentiations of his/her index ranging from 1 to r, and r multiplications with its coefficients. All operations will take place under modulo q, so mod-

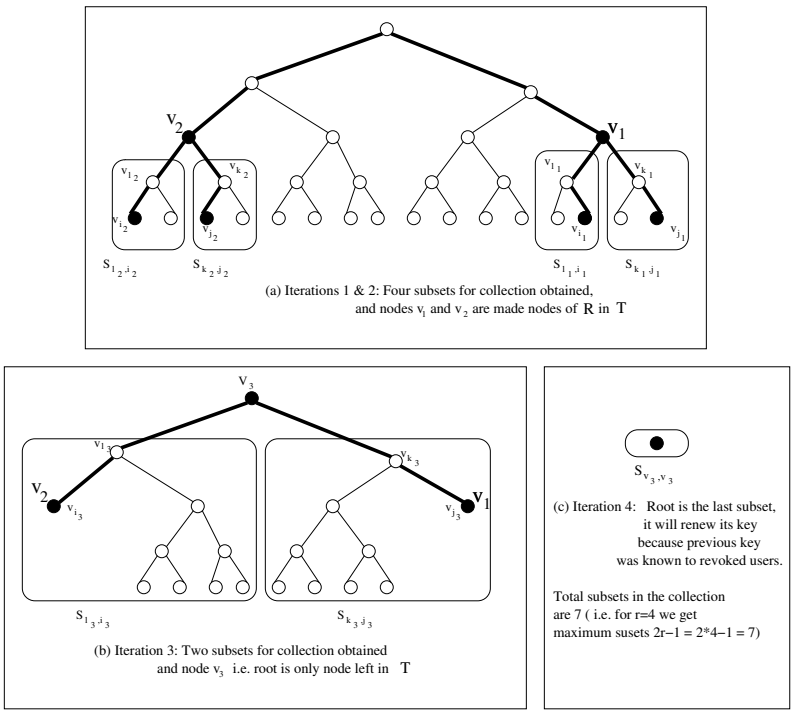

Figure 2: Subset partitions using subset difference method

ular reduction is also required. Next, the user will have to evaluate the broadcast message at his/her index. If r is a large exponentiation degree then this scheme can be proven to be very expensive. Thus, we claim that key recovery in the scheme in [5] is more expensive than the one in the SD scheme [6].

Keys to store in the SD scheme are on the order of $O(\log^2(N))$, whereas in the other scheme they are proportional to the number of sessions, m. If the network is very large the first approach will require more keys to store, whereas this would be the case with the second scheme if there are more frequent revocations. Thus, the decision of which method provides more savings in terms of storage in a large network may depend on a particular application or environment. It is important to point out that the scheme in [5] will require the group manager to make a prior estimation of the possible sessions in order to pre-compute the masking polynomial for each session. This is not the case in the SD scheme, where key distribution is independent of the number of sessions. In addition and because of this property, the SD scheme does not require a redistribution of any secret information after the number of sessions exceeds the estimation, as is the case in [5].

As a final comparison of the two schemes, we look at their resistance against the collusion of revoked users. The scheme in [5] is secure against the collusion of r revoked users, whereas the SD scheme can revoke any number of users and remain secure against their collusion. Given the discussion in this section, we claim that the SD method proposed in [6] shows better performance than the one in [5]; however it lacks the self-healing feature of the polynomial based scheme.

Next, we describe a method to include a self-healing feature in the SD scheme and discuss optimization techniques that can be used to minimize message size and the required computation in different cases.

3 Our Approach

To add a self-healing feature for the SD method, we consider three cases.

Case 1: In this case, we assume that there are no changes to the group membership (no member is added to or removed from the group) between some session s and the following group of t sessions. We can ensure self-healing in the broadcast message for each session by taking the straightforward approach of allowing each broadcast message to contain the respective session key encrypted along with the keys of previous sessions up to session $s+1$. For some subset $S_{i,j}$, the session key K_{s+t} for the session $s+t$ is encrypted with the subset key $L_{i,j}$ along with other keys as:

$$E_{L_{i,j}}(K_{s+t}||K_{s+t-1}||\cdots||K_{s+1})$$

Since all the users in the group are authorized for these sessions, those who did not receive any of the previous session keys can recover that key and the others will discard unwanted keys. The number of encryptions in this case is the same as in the basic SD proposed in [6], however the message size increases by as many the session

keys as are made recoverable. In [5, 11], such session keys are sent under masking polynomials and are concatenated with the current session key in the similar way.

In [6], session keys are randomly chosen and are independent of each other. For optimized self-healing we suggest using a hash chain of session keys for case 1. The server would select a random secret r and compute the hash chain as:

$$h_1 = H(r), h_2 = H(h_1), \cdots, h_t = H(h_{t-1})$$

Keys would be released in reverse order, that is, h_t would be the key for session $s+1$, h_{t-1} for session $s+2$ and so on. Given the current session key, users can compute previous session keys using the one-way hash function H recursively. It is to be noted that the broadcast message will contain only the one session key of the current session and self-healing is possible by computing previous keys using the hash function.

One possible reason for changing session keys without revoking any members could be the use of smaller keys by users with resource-constrained devices in order to improve performance in group communications. Such smaller keys should not be used for a long time and should be refreshed periodically. This can result in different sessions not necessarily revoking any members as in Case 1.

Case 2: In this case, we consider that at any stage after some session s, a group of t sessions occurs in which the members are revoked in some sessions but no new member is added or rejoins. For such cases, we use the same recursive hash chain of the session keys, as in Case 1, with some modification. Starting from session s, if $s+i$, for $1 \leq i \leq t$, is the first session in which the first member is revoked; new subsets are constructed by including all previously unauthorized users and those revoked in session $s+i$ in order to protect the new session key h_{t-i+1}. However, the key h_{t-i+2}, for the session $s+i-1$ and all the sessions before it up to $s+1$, should be appended with the current session key in order to ensure self-healing for revoked users who were authorized for the sessions from $s+1$ to $s+i-1$. (Please refer to Figure 3.)

For every subsequent removal of members, subsets are made including all previously unauthorized users and the newly revoked members. In general, for all those sessions in which any member is revoked, the key of its preceding session is included in the new session key message. This will ensure self-healing for revoked members to recover those keys for which they were eligible. The optimization we get from recursive hash chains is that those members who remain throughout the period of t sessions, only require one key of current session for self-healing, whereas with the basic approach it requires all the previous keys to be appended to the current session key, as in [5, 11]. And those members who are revoked during this period only require the key for the last session they were eligible to be able to recover previous keys. In other words, the optimization of the case 1 is used on the subgroup level in t sessions. We do not allow addition of new members, or rejoining of the old but revoked members in this case in any session, as such members can extract all session keys of previous sessions for which they were unauthorized.

Case 3: In this case we consider that after some session s, some members in subsequent sessions can be added to and/or removed from the group. In order to protect previous session keys from unauthorized users, a straightforward approach (or brute force method) is to make subsets for each session and encrypt its session key with respective subset keys, and append all such encrypted keys with the current session key. This would increase the message size and number of encryptions to be very large. But there are possibilities to optimize it through different methods, some of which are mentioned below.

If same users are revoked in multiple sessions, such session keys can be encrypted once and the SD algorithm will be invoked once. For example, if users u_1, u_2, and u_3 were revoked for some sessions s_1, s_2, and s_3. The subsets based on these revoked users will be made once and three session keys can be encrypted once for each subset. The authorized users also need to decrypt the session keys once. However, we may not find any group of sessions with same revoked members; there can be sessions in which different users were revoked.

In order to reduce number of encryptions and decryptions needed in this case, we require that the root maintains a record or a set of revoked users for each session. The root will then find out a subgroup of such revoked sets among which one set is the superset i.e. it contains all the revoked users of other sets in that subgroup. For example, among $s_1 : \{u_1, u_3, u_4\}$, $s_2 : \{u_2, u_5, u_8\}$, $s_3 : \{u_1, u_3\}$ and $s_4 : \{u_1, u_2, u_3, u_4\}$, we find a subgroup of sessions s_1, s_3, and s_4, where s_4 is the superset. The root will then make subsets based on the members in the superset and encrypt all the session keys (of s_1, s_3, and s_4) for those subsets. There can be some users in such superset that might be authorized for any of the sessions in that subgroup, such as in above example user u_2 is authorized for sessions s_1 and s_3, and user u_4 is authorized for session s_3. Such users should be provided those session keys encrypted in their pairwise keys with the root.

In this way, we are providing previous session keys to only authorized users and reduce encryption and decryption operations. Recall that in the SD method, the maximum number of subsets can occur is $2r-1$; considering that, our above optimization will require 10 encryption operations for three session keys, whereas through straightforward approach of repeated application of the SD method to these sessions will require 15 encryptions. Here, as it is expected, the message size increases in our optimization method due to encryption of multiple keys for different subsets. However, the users authorized for all the sessions in that subgroup will still require one decryption operation instead of multiple decryptions through un-

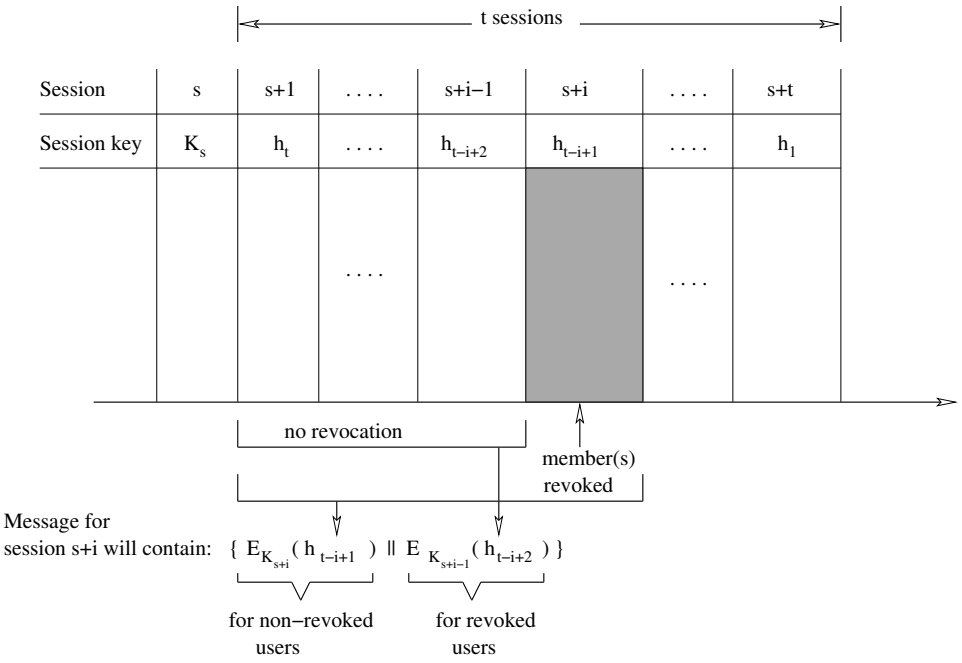

Figure 3: Self-healing for revoked and non-revoked users

optimized method. We require the root to decide either of the method (repeated application of the SD method or optimization method) based on the affordable message size and the number of encryption and decryption operations required for different messages.

Another method which can provide performance improvements is to limit the number of recoverable session keys to fewer sessions, i.e. using sliding-window technique instead of including all previous sessions. This will be helpful to reduce message size and encryption operations. The minimum size of the window should be greater than or equal to the average number of messages or updates a node can miss consecutively, and the maximum sessions allowed for any user to remain offline i.e. *window size = max* [*average packet loss, max. allowed sessions to remain offline*]. In this way upon successful reception of any message a user should be able to recover missing updates.

The periodic relabelling of the nodes can also reduce message size since the administrator does not have to protect keys from the members that are revoked for long time. The decision for relabelling of nodes depends on the complexity of relabelling and the cost associated to keeping track of revoked users. If the cost is high, it may be beneficial to relabel all the nodes which in turn will result reduction in size of subsequent messages.

3.1 Mutual Healing

A main purpose of the self-healing property is to minimize computational and communication overheads for both the central authority and nodes. But, it is virtually impossible to make nodes completely self-relying without affecting network performance. We thus propose the idea of mutual healing among nodes. It requires nodes to be cooperative with each other and would work in a similar way as the packet relaying is carried out in mobile ad hoc networks. The basic principle behind such mechanism is that a node would cooperate with other node(s) as it might need such cooperation for itself in future. One motivation behind mutual healing is that, if a node has missed an update message, it does not have to wait until the next update message to recover previous session key, instead it can get assistance from its neighboring nodes to recover that key instantly. Such service is important, for example, in the case of live or real-time transmissions. There are two requirements for mutual healing, first the authentication of requesting node, and second its authorization for the requested session key.

If a node does not receive the broadcast update from the central authority, it will then contact its neighboring nodes to get the missing session key. The neighboring node first needs to authenticate the requesting node. This requires nodes either share a common secret or use public key cryptography to verify authentication. We suggest the use of an identity-based pairwise shared secret proposed in [10], as it requires less communication and is non-interactive if identities are known to the nodes. The challenge-response protocol can be incorporated in the verification process after computing shared secrets. The authentication process will be helpful to identify cooperating or misbehaving nodes, and will allow nodes to provide such service to only authorized users thereby avoiding becoming the target outsider attacks on for their resource consumption.

In order to determine whether to authorize the requesting node for a given session key, we do not require the node

that is assisting with the self-healing has to get authorization information from the central authority. Instead, it only needs to forward broadcast message it received from the root. If the requesting node is among the authorized nodes, it would be able to recover the session key(s), otherwise not. This requires nodes to store broadcast messages sent by the root to help other nodes. In order to increase the availability of this service, specifically for mobile environments, nodes should backup complete broadcast messages so that mobile nodes of different subsets should also be able to recover their keys.

4 Conclusion

In this paper, we have proposed the self-healing feature for the Subset Difference (SD) method proposed by D. Naor *et al.*. We have also suggested some optimization techniques that can be used to control the message size and encryption operations needed to ensure a self-healing feature. More work in needed to investigate further optimization by efficient use of session keys of different type (independent or associated to other keys), and by proper grouping of sessions into subgroups. We have given a comparison of this method to others in the literature, and have shown its possible advantages for large lossy networks, such as ad hoc networks.

We have also discussed the notion of mutual healing and some of its requirements; moreover, our future work on mutual healing is focused on its further technical details. One drawback of our optimization approach using recursive hash chains is that if any of such session keys or node itself is compromised, its preceding keys can also be revealed i.e. it trades off backward secrecy at the cost of reducing message size. However, this is inherent in a system with revocation and self-healing capabilities, as a revoked user can continue recording all ciphertexts and is able to recover them at a later point by obtaining a new valid key and using the self-healing property. One possible way to overcome this drawback is to allow for additional revocation sets, and refining the algorithm proposed in the paper. It should be noted that under the condition of Case 2, the collusion of any number of users will not allow them to compute keys for the sessions they were not allowed for, and forward secrecy is ensured due to one-way hash operation.

Acknowledgements

This work was partially supported by a grant from Natural Sciences and Engineering Research Council of Canada (NSERC).

References

[1] J. Anzai, N. Matsuzaki, and T. Matsumoto, "A quick group key distribution scheme with entity revocation," in *The Proceedings of Advances in Cryptology - ASIACRYPT'99*, vol. 1716 of *Lecture Notes in Computer Science*, pp. 333–347, Singapore, November 1999. Springer-Verlog.

[2] T. Asano, "Reducing storage at receivers in SD and LSD broadcast encryption schemes," in *The Proceedings of the 4th International Workshop on Information Security Applications - WISE'03*, vol. 2908 of *Lecture Notes in Computer Science*, pp. 317–332. Springer-Verlog, 2004.

[3] A. Fiat and M. Naor, "Broadcast encryption," in *The Proceedings of Advances in Cryptology - CRYPTO'93*, vol. 773 of *Lecture Notes in Computer Science*, pp. 480–491. Springer-Verlog, 1994.

[4] D. Halvey and A. Shamir, "The LSD broadcast encryption scheme," in *The Proceedings of Advances in Cryptology - Crypto'02*, vol. 2442 of *Lecture Notes in Computer Science*, pp. 47–60. Springer-Verlog, 2002.

[5] D. Liu, P. Ning and K. Sun, "Efficient self-healing group key distribution with revocation capability," in *The Proceedings of 10th ACM Conference on Computer and Communications Security - CCS'03*, pp. 231–240, Washington D.C., USA, October 2003. ACM Press.

[6] D. Naor, M. Naor and J. Lotspiech, "Revocation and tracing schemes for stateless receivers," in *The Proceedings of Advances in Cryptology 2001 - Crypto'01*, Lecture Notes in Computer Science, pp. 41–62. Springer-Verlog, 2001.

[7] M. Naor and B. Pinkas, "Efficient trace and revoke schemes," in *The Proceedings of Financial Cryptography 2000*, vol. 1962 of *Lecture Notes in Computer Science*, pp. 1–20, Anguilla, British West Indies, February 2000. Springer-Verlog.

[8] A. Perrig, D. Song and J. D. Tygar, "ELK, a new protocol for efficient large-group key distribution," in *The Proceedings of IEEE Symposium on Security and Privacy*, pp. 247–262, 2001.

[9] B. Pinkas, "Efficient state updates for key management," *Proceedings of the IEEE, Special Issue on Enabling Technologies for Digital Rights Management*, vol. 92(6), pp. 910–917, June 2004.

[10] R. Sakai, K. Ohgishi, and M. Kasahara, "Cryptosystems based on pairing," in *The Proceedings of the 2000 Symposium on Cryptography and Information Security*, Okinawa, Japan, January 2000.

[11] J. Staddon, S. Miner, M. Franklin, D. Balfanz, M. Malkin, and D. Dean, "Self-healing key distribution with revocation," in *The Proceedings of the 2002 IEEE Symposium on Security and Privacy*, pp. 224–240, 2002.

[12] D. Wallner, E. Harder and R. Agee. "Key management for multicast: Issues and architectures,". tech. rep., The Internet Engineering Task Force (IETF), June 1999.

[13] C. K. Wong, M. Gouda and S. Lam, "Secure group communications using key graphs," in *The Proceedings of ACM SIGCOMM'98*, vol. 28(4), pp. 68–79, Vancouver, BC, Canada, 1998.

[14] C. K. Wong and S. Lam, "Keystone: A group key management service," in *The Proceedings of International Conference on Telecommunications - ICT'00*, pp. 1–6, Acapulco, Mexico, May 2000.

[15] Y. Yang, X. Li, X. Zhang, and S. Lam, "Reliable group rekeying: A performance analysis," in *The Proceedings of ACM SIGCOMM'01*, pp. 27–38, San Diego, CA, USA, August 2001.

Muhammad Junaid Bohio completed his M.A.Sc at the University of Ottawa in 2004. He has worked with NTG Clarity Networks Inc. as a Security Analyst. His research interests include cryptography, network security and software security.

Ali Miri received his BSc and MSc in Mathematics from the University of Toronto in 1991 and 1993 respectively, and his PhD in Electrical and Computer Engineering from the University of Waterloo in 1997. Having worked as an NSERC Postdoctoral Fellow at the University of Waterloo and the University of Toronto, he joined the School of Information Technology and Engineering (SITE) at the University of Ottawa in July of 2001, where he is currently working as an assistant professor. His research interests include security and privacy technologies and their applications in e-business and e-commerce, such as network security and the role of Public Key Cryptography.