Cryptanalysis of a Polynomial-based Key Management Scheme for Secure Group Communication

Abdel Alim Kamal
Electrical and Computer Engineering Department, Concordia University, Montreal, Qc, Canada
1455 Boulevard de Maisonneuve Ouest Montreal, QC H3G 1M8
(Email:a_kamala@ece.concordia.ca)

(Received May 19, 2012; revised and accepted June 12, 2012)

Abstract

Piao *et al.* proposed a polynomial-based key management scheme for secure intra-group and inter-group communication. In this note, we present a simple attack on this scheme and show that it does not satisfy group forward and backward secrecy. In other words, we show that when a node leaves a group, it can easily compute the new intra-group key based on its old key and the publicly broadcasted data. Similarly, we also show that when a node joins a group, it can discover the old keys.

Keywords: Cryptanalysis, Polynomial-based key management, Group communication.

1 Introduction

Secure group communication is an important component in many applications (e.g., see [3, 7, 2]). Designing efficient key distribution and key update protocols for secure intra-group and inter-group communication is a challenging task. Wang *et al.* [5, 4, 6] proposed a polynomial-based scheme to ensure the security of inter-group communication where they utilized polynomials to support the distribution of personal key shares and protect the inter-group multicast traffic. Recently, Piao *et al.* [1] adopted similar polynomial-based mechanisms to achieve efficient intra-group key refreshment and to create an inter-group key. In this scheme, the group members and the group controller can share the intra-group key without any heavy encryption/decryption operations. The proposed mechanism has a small number of rekeying messages when members of the group get changed. It also lessens the storage overhead of group members and the group controller.

Despite the above appealing features, in this note, we show that the Piao *et al.* polynomial-based key management scheme is not secure; it does not satisfy the intra-group forward and backward secrecy requirements. In other words, we show that when a node leaves a group, it can easily compute the new intra-group key based on its key and the publicly broadcasted data. Similarly, we also show that when a node joins a group, it can discover the previously used key.

The rest of this note is organized as follows. In the next section, we briefly review the relevant details of the Piao *et al.* group key management scheme. The proposed attack is described in Section 3 and our conclusion is given in Section 4.

2 Description of the Piao *et al.* Group Communication Scheme

In this section, we briefly review the relevant details of the Piao *et al.* intra-group key management scheme. Further details regarding the generation of inter-group keys can be found in [1]. For our purpose, it suffices to note that compromising the intra-group keys naturally leads to the compromise of the inter-group keys.

Let n denote the number of members in a group. Each member (also refereed to as node) is identified by a unique ID. Nodes in the networks are divided into d different groups, where $k \in \{1 \cdots d\}$ denotes the group index. KEK_i is the secret key which is shared between the group controller and member i within the group.

In Piao *et al.* group key management scheme, two kinds of polynomials are applied. The first polynomial (denoted by P) is used to derive the intra-group key, and the second polynomial is used to create the inter-group key. In what follows we focus on the intra-group key management scheme which aims to allow members in group G_k to share the intra-group key GK_k securely and efficiently. The intra-group key agreement protocol can be summarized as follows:

1) The group controller gives every member $i, i = 1 \cdots n$, a Key Encryption Key, KEK_i, using a secure channel.

2) The group controller generates a polynomial

$$P = \begin{array}{l}(x - KEK_1)(x - KEK_2) \cdots \\ (x - KEK_n) + GK_k\end{array} \quad (1)$$

which uses all secret keys $KEK_i, i = 1 \cdots n$, and GK_k is the group key of G_k generated by the group controller. The group controller broadcasts the coefficinets of the expanded P to the members.

3) When the i^{th} group member receives P, this member computes the group key GK_k as:

$$GK_k = P(KEK_i), i = 1, \cdots, n.$$

When a group membership change happens, the corresponding intra-group and inter-group keys must be renewed to enforce forward and backward secrecy.

In rekeying for member join, suppose that, a member w wants to join the group G_k. Also assume that the current members of G_k have been using GK_k to encrypt the multicast traffic within the group. To prevent member w from getting access to the previous messages, the group key GK_k must be replaced by a new random key, $GK_k^{'}$. The steps of the rekeying protocol can be described as follows:

1) The member w shares secret key KEK_w with the group controller.

2) In order to maintain backward secrecy the group controller generates a new polynomial

$$P_{new} = (x - KEK_1)(x - KEK_2) \cdots \\ (x - KEK_w) \cdots (x - KEK_n) + GK_k^{'}$$

where $GK_k^{'}$ is the new intra-group key. The group controller broadcasts, in the clear without any encryption, the new polynomial to the members.

3) After obtaining the new polynomial P, all of the group members including w can derive the new group key $GK_k^{'}$ using their KEK_i (i.e., by substituting into P_{new} with $x = KEK_i, i = 1, \cdots, n$).

Similarly, when a member i is expelled from G_k, the group key GK_k must be replaced by the new secret $GK_k^{'}$. The group controller regenerates a new polynomial

$$P_{new} = (x - KEK_1) \cdots (x - KEK_{i-1}) \\ (x - KEK_{i+1}) \cdots (x - KEK_n) + GK_k^{'}$$

where $GK_k^{'}$ is the new group key generated by the group controller. Other members in G_k can derive $GK_k^{'}$ but node i is not supposed to be able to derive $GK_k^{'}$.

In all the above protocols, the group controller sends the expanded polynomials (P, P_{new}) without any encryption. The authors in [1] argue that it is not easy to guess the intra-group key from this polynomial because of the difficulty to factor these polynomial in the form of Equation (1) given the fact that GK is not known (see the security argument and example in Section 4.1.2 in [1].) In the next section, by analyzing the rekeying operations, we show that this is not the case. In fact, as will be shown below, both the forward and backward security requirements can be easily violated without the need to perform any polynomial factorization.

3 The Proposed Attacks

In this section, we show that the rekeying operations described above are not secure. In particular, we show that forward secrecy is not assured; when a node leaves a group, it can easily access the traffic after leaving using the old keys. Also, backward secrecy is not assured; when a node joins a group, it can discover the old keys based on its current knowledge.

In rekeying for member join, suppose that a member w wants to join the group G_k, then after sharing its secret key KEK_w with the group controller, this new member receives the new polynomial

$$P_{new} = \begin{array}{l}(x - KEK_1)(x - KEK_2) \cdots \\ (x - KEK_w)(x - KEK_n) + GK_k^{'}\end{array} \quad (2)$$

via the group controller. So, the member w can calculate the new intra-group key $GK_k^{'}$ by substituting $x = KEK_w$ and in this case $GK_k^{'} = P_{new}(KEK_w)$.

Since all polynomials are transferred in the clear, then the member w can easily access

$$P_{old} = \begin{array}{l}(x - KEK_1)(x - KEK_2) \cdots \\ (x - KEK_n) + GK_k\end{array} \quad (3)$$

From Equation 2, w can calculate

$$\frac{P_{new} - GK_k^{'}}{(x - KEK_w)} = \\ (x - KEK_1)(x - KEK_2) \cdots (x - KEK_n)$$

Thus node w can calculate the old intra-group key $GK_k = P_{old} - \frac{P_{new} - GK_k^{'}}{(x - KEK_w)}$ and consequently becomes able to access previously exchanged messages which were encrypted using GK_k.

Similarly, in rekeying for member leave, when member i is expelled from G_k, the group key GK_k must be replaced by the new secret $GK_k^{'}$. Thus we have

$$P_{new} = \\ \begin{array}{l}(x - KEK_1)(x - KEK_2) \cdots \\ (x - KEK_{i-1})(x - KEK_{i+1}) \cdots \\ (x - KEK_n) + GK_k^{'}\end{array} \quad (4)$$

Note that member i knows the old public polynomial

$$\begin{array}{l} P_{old} = \\ (x - KEK_1)(x - KEK_2) \cdots (x - KEK_{i-1}) \\ (x - KEK_i)(x - KEK_{i+1}) \cdots \\ (x - KEK_n) + GK_k \end{array} \quad (5)$$

since it was transmitted in the clear. Thus this evicted member can calculate

$$\begin{array}{l} \frac{P_{old} - GK_k}{(x - KEK_i)} = \\ (x - KEK_1)(x - KEK_2) \cdots \\ (x - KEK_{i-1})(x - KEK_{i+1}) \cdots (x - KEK_n) \end{array}$$

to derive the new intra-group key

$$GK_k^{'} = P_{new} - \frac{P_{old} - GK_k}{(x - KEK_i)}.$$

Remark 1 *Using the same idea of the above attacks, it is easy to show that if $n-1$ members collude, then they can easily recover the secret key of the n^{th} member. For example, members 1 to $n-1$ can recover the secret key of the victim member n, i.e., KEK_n by calculating*

$$(x - KEK_n) = \frac{P - GK_k}{(x - KEK_1)(x - KEK_2) \cdots (x - KEK_{n-1})}$$

Recovering KEK_n allows these colluding members to eavesdrop private communications between the group controller and this member. Similar attacks, that requires factoring low order degree polynomials, can be launched by a smaller number of colluding nodes.

4 Conclusions

The group communication scheme proposed by Piao *et al.* is not secure. In particular, it does not satisfy the forward and backward secrecy requirements. Furthermore, colluding members within a group can recover the key encryption keys of other group members.

References

[1] Y. Piao, J. Kim, U. Tariq, and M. Hong, "Polynomial-based key management for secure intra-group and inter-group communication," *Computers and Mathematics with Applications*, 2012.

[2] R. Srinivasan, V. Vaidehi, Rajavelu Rajaraman, S. Kanagaraj, R. Chidambaram Kalimuthu, and R. Dharmaraj, "Secure group key management scheme for multicast net-works," *International Journal Network Security*, vol. 11, no. 1, pp. 33–38, 2010.

[3] H. Y. Um and Edward J. Delp, "A secure group key management scheme for wireless cellular systems," *International Journal Network Security*, vol. 6, no. 1, pp. 40–52, 2008.

[4] W. Wang and B. K. Bhargava, "Key distribution and update for secure inter-group multicast communication," in *Proceedings of the Workshop on Security of ad hoc and Sensor Networks (SASN'05)*, pp. 43–52, Alexandria, VA, USA, 2005. ACM.

[5] W. Wang and T. Stransky, "Stateless key distribution for secure intra and inter-group multicast in mobile wireless network," *Computer Networks*, vol. 51, no. 15, pp. 4303–4321, 2007.

[6] W. Wang and Y. Wang, "Secure groupbased information sharing in mobile ad hoc networks," in *Proceedings of the International Conference on Communications (ICC'08)*, pp. 1695–1699, Charlotte, NC, USA, 2008. IEEE.

[7] Q. Zhang, Y. Wang, and J. P. Jue, "A key management scheme for hierarchical access control in group communication," *International Journal Network Security*, vol. 7, no. 3, pp. 323–334, 2008.

Abdel Alim Kamal received his B.Sc. degree in Pure Mathematics and Computer Science, and M.Sc. degree in Computer Science from Menoufia University, Egypt, in 1999 and 2006, respectively. Presently, he is pursuing his Ph.D. degree in Electrical and Computer Engineering at Concordia University, Montreal, QC, Canada. His current research interests are in the area of cryptography and data security.