Securing Dynamic Group Membership Information over Multicast: Attacks and Immunization

Yan Sun and K. J. Ray Liu
Department of Electrical and Computer Engineering
University of Maryland, College Park, MD 20740

Abstract— **In secure multicast communications, key management is employed to prevent unauthorized access to the multicast content. Key management, however, can disclose the information about the dynamics of the group membership to inside attackers, which is a potential threat to many multicast applications. In this paper, we investigated several attack strategies for stealing group dynamic information and demonstrated their effectiveness through simulations. Further, we proposed an anti-attack technique utilizing batch rekeying and phantom users, and derived performance criteria describing the security level of the proposed scheme. The proposed anti-attack scheme was evaluated based on the membership data obtained from real MBone sessions.**

I. Introduction

The rapid progress in the technologies underlying multicast networking has led to the development of many multicast services, such as streaming stock quotes, video conferencing and communal gaming [1]. Before these group-oriented applications can be successfully deployed, *access control* mechanism must be developed such that only authorized users can access the group communication [2] [3]. Access control is usually achieved by encrypting the content using an encryption key, known as the session key (SK) that is shared by all legitimate group members. Since the group membership will most likely be dynamic with users joining and leaving the service, it is necessary to update the SK in order to prevent the leaving user from accessing future communication and prevent the joining user from accessing prior communication [2] [3]. This key updating process is usually referred to as the *Key Management*.

A popular class of multicast key management schemes employs a tree hierarchy for the maintenance of keying material [2]–[6]. These schemes focus mainly on the problem of maintaining access control with dynamic membership and reducing the usage of computation, communication and storage resources for the users and the group controller. These schemes, however, can disclose information about the dynamics of the group membership to both insiders and outsiders. We collectively refer to group dynamics information (GDI) as information describing the dynamic membership of a group application, such as the number of users in the multicast group as a function of time, and the number of users who join or leave the service during a time interval.

In many multicast applications, GDI is confidential and should not be disclosed to either group members or outsiders. For example, in a commercial multicast program, disclosure of the GDI to competitors could enable them to analyze the statistical behavior of the audience and help them to develop effective competition strategies. Another example is the military involved scenario, where GDI may represent the number of soldiers in the battlefield and the number of soldiers moving into or out of certain areas. In this situation, the valid group members, i.e. regular soldiers, are only entitled to obtain general information through the secure multicast communication, but not entitled to acquire the GDI. Leaking GDI to outsiders, most likely the enemies, is even more dangerous.

The traditional key management schemes [4] [6] focus solely on preventing unauthorized access to the multicast content and neglect the issues of the group dynamic information. Therefore, it is important to improve the design of key management schemes such that the GDI are protected from both insiders and outsiders while maintaining the access control to the multicast content.

To address this issue, we have developed several effective strategies for attackers to steal group dynamic information from the key management schemes. These strategies involve exploiting the format of the rekeying messages and estimating GDI directly from the size of the rekeying messages. We also developed an anti-attack method that is fully compatible with existing key management schemes. By utilizing batch rekeying [7] and introducing phantom users, the proposed anti-attack method reduces the mutual information between the rekeying processes observed by the attackers and the true GDI. Various aspects of the proposed anti-attack scheme, such as communication overhead, were evaluated based on the data obtained from real MBone sessions.

The rest of the paper is organized as follows. The attack strategies and the anti-attack method are discussed in Section 2 and Section 3 respectively. In Section 4, the performance criteria of the proposed anti-attack method are derived and the optimization problem is formulated. Simulation results based on the user log data of real MBone sessions are shown in Section 5, followed by the conclusion in Section 6.

II. Attacks on Group Dynamic Information

In this paper, the group dynamic information is particularly referred to as a set of functions as:

- $N(t)$: the number of users in the group at time t.
- $J(t_0, t_1)$: the number of users who join the multicast group between time t_0 and t_1.
- $L(t_0, t_1)$: the number of users who leave the multicast group between time t_0 and t_1.

To acquire GDI information, launching attacks on the key management schemes is a cost-efficient way for the adversaries. Instead of trying to break the encryption or compromise the key distribution center (KDC), the adversaries pay a small

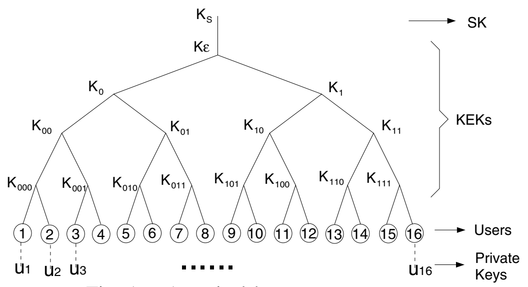

Fig. 1. A typical key management tree

amount of money, subscribe to the service, and become inside attackers. Even if the adversaries cannot become valid group members, they can still seek the opportunity of stealing GDI as outsiders by observing the traffic and trying to distinguish the rekeying messages and other data. In this work, we focus on the attacks on centralized key management schemes, where the trusted third party generates and distributes keys to group members.

A typical key tree used in centralized key management schemes [3]–[6] is illustrated in Figure 1. The multicast data is encrypted using the session key K_s. Each user stores his private key u_i, the session key K_s, and a set of auxiliary keys on the path from himself to the root of the key tree. For example, user 16 possesses $\{u_{16}, K_s, K_\epsilon, K_1, K_{11}, K_{111}\}$. To distinguish the keys, each key is addressed through a unique fixed ID, a version number and a revision number that reflects updates of the keying material. The notation x^{old} represents the old version of key x, x^{new} represents the new version of key x, and $x\{y\}$ represents the key y encrypted by key x.

The key updating process in [6] is briefly described as follows. When a user leaves the service, all his keys need to be updated in order to prevent him from accessing the future communication. For example, when user 16 leaves, new keys are conveyed to the remaining users through a set of rekeying messages as: $u_{15}\{K_{111}^{new}\}$, $K_{111}^{new}\{K_{11}^{new}\}$, $K_{110}^{old}\{K_{11}^{new}\}$, $K_{11}^{new}\{K_1^{new}\}$, $K_{10}^{old}\{K_1^{new}\}$, $K_1^{new}\{K_\varepsilon^{new}\}$, $K_0^{old}\{K_\epsilon^{new}\}$, $K_\epsilon^{new}\{K_s^{new}\}$. Each rekeying message has the same size as the length of the K_s.

When a user joins the service, the KDC chooses a leaf position on the key tree to accommodate the joining user, increases the revision number of all the keys along the path from the new leaf to the root, and generates new keys through a one-way function [6]. The joining user obtains the new keys through the unicast channel, while the existing users in the group will know about the key change when the first data packet indicating the use of the increased revision arrives. No additional rekeying messages are necessary.

Despite some slight differences, most centralized key management schemes [2]–[6] share two common properties. First, group members can distinguish the key updating process due to user join and the process due to user departure. Second, the amount of rekeying messages is closely related with the group size. These properties lead to several attack strategies.

A. Estimation of $J(t_0, t_1)$ and $L(t_0, t_1)$ by inside attackers

When an inside attacker receives the rekeying message containing K_ϵ^{new} encrypted by one of his other keys, he assumes that one user leaves. When he observes the increase of the revision number of K_ϵ, he assumes that one user joins the service. This strategy is effective when most users do not join/leave simultaneously. Otherwise, more complicated techniques involving examining the amount of rekeying messages should be utilized.

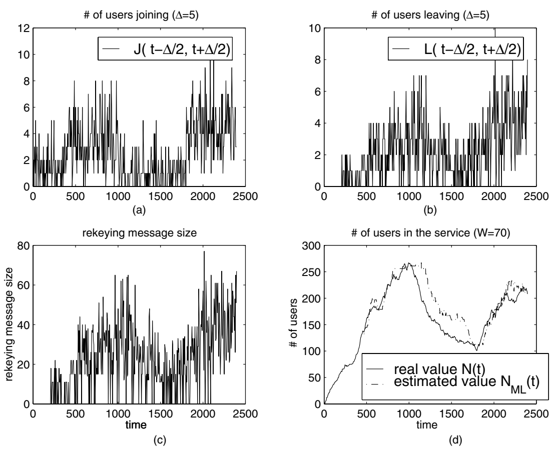

Fig. 2. Group dynamic information and the estimation of $N(t)$

B. Estimation of $N(t)$

When the previous attack is successful, $N(t)$ can be calculated from $J(t_0, t_1)$ and $L(t_0, t_1)$ as: $N(t_1) = N(t_0) + J(t_0, t_1) - L(t_0, t_1)$. Otherwise, the attacker estimates $N(t)$ directly from the *rekeying message size*, defined as the amount of rekeying messages measured in the unit as the same size as the SK. This strategy can be used by insiders, as well as outsiders who can observe the traffic and distinguish the rekeying messages and other multicast data.

We assume that $N(t)$ does not change much within a short period of time. In this time period, there are W departure users who do not leave simultaneously. Thus, the attacker makes W observations of the rekey message size due to single user departure, denoted by $\{m_1, m_2, \cdots, m_W\}$. We also assume that the position of the joining user is chosen such that the key tree is maintained as balanced as possible, and the attacker knows the degree of the key tree, denoted by d. We introduce several variables as: $\{l_i = (m_i + 1)/d\}$, $L_{max} = \max(l_i)$, and $L_{min} = \min(l_i)$. Then, we can show that the Maximum Likelihood(ML) estimator of $N(t)$ is:

$$N(t)_{ML} = \frac{W}{\sum_{k=L_{min}}^{L_{max}} h(k) \cdot d^{-k}} \qquad (1)$$

where $h(k)$ denotes the number of elements in set $\{l_i, l_i = k\}$, and obviously, $\sum_k h(k) = W$.

This ML estimator is applied to a simulated multicast service. As suggested in [8], we assume that the user arrival process is Poisson, and the service duration is exponentially distributed. The entire service period is divided into several sessions. The model parameters, i.e. user arrival rate and average service time, are fixed within each session and vary in different sessions. Figure 2(a) and Figure 2(b) illustrate the number of join and departure users, respectively. Figure 2(c) illustrates the rekeying message size. The true values

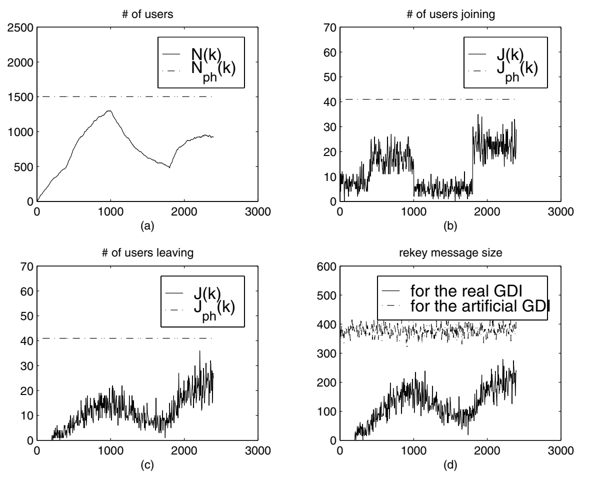

Fig. 3. The anti-attack scheme using phantom users and Batch rekeying

of $N(t)$ and the values obtained by using the ML estimator are compared in Figure 2(d). The difference between $N(t)$ and $N_{ML}(t)$ is smaller than 20, and the changing trend of the group size is well captured by the attacker. Although not perfect, this estimator can help the attackers to achieve many of their goals, such as analyzing audience behavior and monitoring the changes in group size.

III. Anti-attack techniques

The discussion in Section II does not cover all possible attack strategies. New attacks may emerge in the future. Therefore, we propose a one-fit-all anti-attack method that is robust to various types of attacks by utilizing batch rekeying and phantom users.

As proposed in [7], batch rekeying is to postpone the update of keys such that several users can be added to or removed from the key tree together. In this work, batch rekeying is implemented as updating keys periodically, and the key updating period is denoted by B_T. Since key updating only occurs at time kB_T for integer k's, the notations of GDI are simplified as: $J(k) = J((k-1)B_T, kB_T)$, $L(k) = L((k-1)B_T, kB_T)$, and $N(k) = N(kB_T)$.

To protect the GDI from various types of attacks, we propose to insert phantom users into the system. These phantom users, as well as their join and departure behavior, are created by the KDC in such a way that the combined effects of phantom and real users lead to a new rekeying process, which reveals little information about the real GDI.

Let $N_{ph}(k)$ denote the total number of real and phantom users, and $J_{ph}(k)$ and $L_{ph}(k)$ denote the total number of real and phantom users that join/leave the service respectively. $N_{ph}(t)$, $J_{ph}(k)$, and $L_{ph}(k)$ are referred to as the *artificial GDI*. After inserting phantom users, the rekeying process reveals only the artificial GDI to the attackers. Intuitively, the correlation between the artificial GDI and the real GDI should be reduced as much as possible. We choose artificial GDI as a set of constant functions, i.e. $J_{ph}(k) = L_0$, $L_{ph}(k) = L_0$, and $N_{ph}(k) = N_0$. It is important to note that the artificial GDI are not independent of the real GDI because the number of phantom users must be non-negative. Utilizing phantom users and batch rekeying, the key management scheme is modified as:

(1) Determine N_0 and L_0 based on the system requirements and the users' statistical behavior. The criteria for selecting N_0 and L_0 will be described in Section IV.
(2) Before the service starts, create N_0 phantom users and establish a key tree to accommodate them. Let t_r denote the time when the latest key update occurs. Set $t_r = 0$ and index $k = 1$.
(3) While the service is not finished, execute the follows:
   - Record user join and departure requests in the time interval $[t_r,\ t_r + B_T]$, i.e. $J(k)$ and $L(k)$. During this time, the current SK will be sent to the joining users so that they can start receiving the multicast content without delay.
   - At time $t_r + B_T$, we create $L_0 - J(k)$ phantom users that join the service, and then select $L_0 - L(k)$ phantom users in the current system and make them leave. According to the rekeying procedure presented in [6], keys are updated for both phantom and real users. It is clear that the total number of real and phantom users are maintained to be N_0.
   - Set $t_r = t_r + B_T$, and $k = k + 1$.

To demonstrate the effects of the phantom users, the real GDI ($N(k)$, $L(k)$, $J(k)$) and the artificial GDI ($N_{ph}(k)$, $L_{ph}(k)$, $J_{ph}(k)$) are illustrated in Figure 3(a) 3(b) 3(c). The simulation results of communication overhead is shown in Figure 3(d), where the solid line represents the rekeying message size without using phantom users and the dash line represents the rekeying message size when the proposed anti-attack method is applied. It is seen that the observed rekeying process reveals very limited information about the real GDI. Not surprisingly, the communication overhead increases, which is the major disadvantage of the proposed anti-attack method.

IV. Performance Measurement and Optimization

In this section, we derive the performance measurement for the proposed anti-attack scheme and then formulate the optimization problem for choosing the parameter L_0 and N_0.

A. Overflow probability

Since the real GDI are random processes, it is possible that the predetermined L_0 and N_0 are not large enough such that the artificial GDI cannot be maintained as straight lines. For example, when $N(k) > N_0$, the artificial GDI at time $k \cdot B_T$ cannot be the predetermined value N_0 because the number of phantom users must be non-negative. Thus, we define the *overflow probability* as:

$$\begin{aligned} F_N(N_0) &= \max_k prob\{N(k) > N_0\}, \\ F_J(L_0) &= \max_k prob\{J(k) > L_0\}, \\ F_L(L_0) &= \max_k prob\{L(k) > L_0\}, \end{aligned}$$

where $prob\{.\}$ denotes the probability. The overflow probability can be interpreted as the probability that the proposed anti-attack method fails to protect the real GDI.

B. Communication Overhead

Communication overhead, measured by the rekeying message size, is one of the major performance criteria of key management schemes [2] [3]. We introduce the notation $M(L,N,d)$ as the expected value of the rekeying message size when removing L users from the key tree that contains total N users and has degree d. Thus, $M(L(k),N(k),d)$ and $M(L_0,N_0,d)$ represent the rekeying messages size at time kB_T without or with phantom users respectively. We can show that:

$$\begin{aligned} & M(L,N,d) \\ \approx\ & \sum_{l=0}^{D-1} d\cdot B(d^l, L, \frac{N}{d^D}) + (d-1)B(s_1, L\frac{N-d^D+s_1}{N}, d) \end{aligned}$$

where $D=\lfloor \log_d N\rfloor$, $s_1=\lceil \frac{N-d^D}{d-1}\rceil$. $B(b,i,a)$ describes the number of keys that need to be updated, and equals to the expected number of the occupied boxes when putting i items in b boxes with repetition where each box can have at most a items. A box is called occupied when one or more items are put into the box. We also derived the upper bound as:

$$M(L,N,d) \le dL\log_d(N). \tag{2}$$

The extra communication overhead introduced by using the phantom users is:

$$\sum_k (M(L_0,N_0,d) - M(L(k),N(k),d). \tag{3}$$

C. Leakage of the GDI

The exact amount of information about GDI obtained by the attackers largely depends on their attack methods. To avoid handling various attack strategies, we use mutual information to measure the leakage of the GDI, which represents the maximum amount of information that an attack can possibly obtain.

To simplify the notation, a r.v. X is used to denote either $N(k)$, $L(k)$, or $J(k)$. Without observing any rekeying process, the attackers have some prior knowledge of X. This prior knowledge is the presumed pmf of X, denoted by $p'(x)$. It is important to note that $p'(x)$ is most likely not the true distribution of X. When the attackers have no idea of X at all, $p'(x)$ can be a uniform distribution between 1 and the maximum number of subscribers in a geological service area. We introduce a new r.v. Y with the pmf:

$$p(y) = \begin{cases} 1 & y=Y_0 \\ 0 & o.w. \end{cases}, \text{ and } Y_0 = \begin{cases} N_0, & X=N(k) \\ L_0, & X=L(k),J(k) \end{cases}$$

In the worst case scenario, the attacker is able to obtain the perfect estimation of the artificial GDI, i.e. the value of Y_0, and knows that $Y \ge X$. In this case, the information obtained by the attacker can be described by mutual information $I(X;Y)$. We assume that the conditional pmf $p(x|y)$ is obtained by truncating $p(x)$ and multiplying a scaling factor, as:

$$p(x|y) = \begin{cases} \frac{p(x)}{1-\sum_{x=Y_0+1}^{\infty} p(x)} & x \le Y_0 \\ 0 & x > Y_0 \end{cases}$$

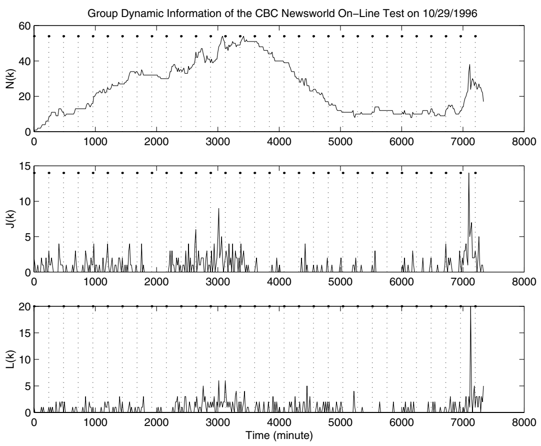

Fig. 4. The GDI of a long audio session in MBone

Then, the mutual information is calculated as:

$$\begin{aligned} & I(X;Y) = H(X) - H(X|Y) \\ = \ & -\sum_x p(x)\log p(x) - \left(-\sum_{x,y} p(x)p(x|y)\log p(x|y) \right) \end{aligned} \tag{4}$$

D. Optimization

From the system design points of view, parameter L_0 and N_0 should be chosen such that the extra communication overhead in (3) is minimized while the overflow probability and the leakage of the GDI do not exceed certain requirements. Thus, the optimization problem is formulated as:

$$\min_{N_0,L_0} M(L_0,N_0,d) \tag{5}$$

$$\begin{aligned} \text{subject to: } & \max(F_N(N_0), F_J(L_0), F_L(L_0)) \le \epsilon_a \\ & I(X;Y)_{X=N(k)} \le \epsilon_n, \\ & I(X;Y)_{X=J(k)} \le \epsilon_J, \\ & I(X;Y)_{X=L(k)} \le \epsilon_L, \end{aligned} \tag{6}$$

where ϵ_a,ϵ_n,ϵ_J and ϵ_L are small positive numbers representing the maximum allowed overflow probability or GDI leakage. It is easy to show that the overflow probability and the mutual information functions in (6) are all monotonous non-increasing with L_0 and/or N_0, and the cost function in (5) is non-decreasing with L_0 and N_0. Therefore, the solution of this optimization problem is the minimum values of all L_0's and N_0's that can achieve the equalities in (6).

V. Simulations based on real MBone sessions

The proposed anti-attack scheme is applied to the data of MBone sessions collected in 1996 [9] . Particularly, we selected one audio session that started on Oct. 29th and lasted for about 5 days and 20 hours. Figure 4 shows the $N(k)$, $L(k)$ and $J(k)$ of this session, where the B_T is chosen to be 15 minutes.

It is suggested that the users statistical behavior, such as inter-arrival and membership duration, can be modelled as experiential distribution or Zipf distribution in a short period of time [8]. In the simulation, the entire service time is divided into non-overlapping sections, as illustrated in Figure 4. The

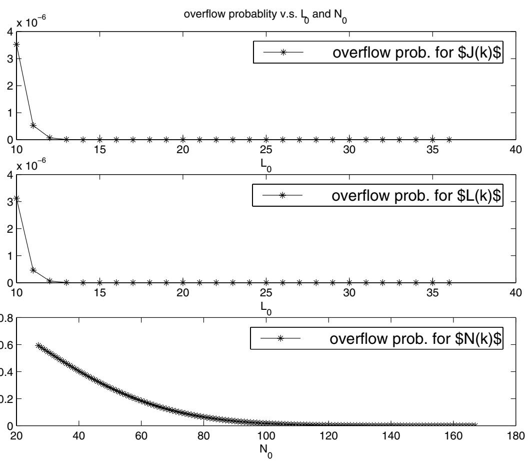

Fig. 5. Overflow probability

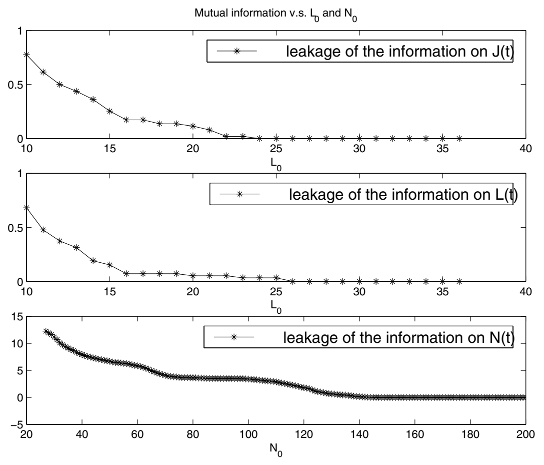

Fig. 6. The maximum possible leakage of the GDI measured by the mutual information

length of these sessions is 4 hours. To simplify the analysis, it is assumed that $N(k)$ is a stationary and ergodic Gaussian process, $L(k)$ and $J(k)$ are stationary and ergodic Poisson processes within each session. Then, we can calculate the overflow probability as a function of L_0 and N_0 for the selected multicast service, which is shown in Figure 5.

In general, it is difficult for the KDC to obtain the attackers' prior knowledge of the GDI. To calculate the mutual information in (4), we assume that the attackers know the distribution of GDI functions averaged over all similar sessions. In particular, let $p(x, s)$ denote the pmf of X for session s, where $s \in S$ and S denotes the set of multicast sessions sharing common properties such as the format of media and the length of service time. Then, the attackers' prior knowledge of X is assume to be the marginal distribution $p(x) = \frac{1}{|S|}\sum_s p(x, s)$. In this work, S is selected from 1996 MBone sessions, and contains all the long audio sessions with more than 10 users. $p(x)$ is approximated by the histogram obtained from these selected sessions. In Figure (6), the mutual information calculated as in (4) is shown as a function of L_0 and N_0.

From Figure 5 and Figure 6, it is straightforward to obtain the minimum values of L_0 and N_0 that satisfied the constrains in (6), which is just the optimal L_0 and N_0 that minimize the communication overhead in (5).

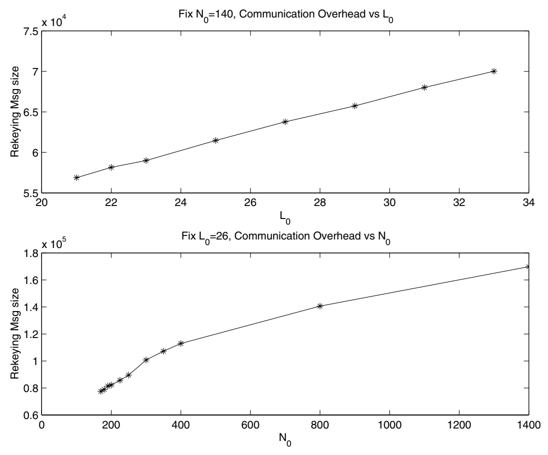

Fig. 7. Communication overhead

Figure 7 illustrates the communication overhead as a function of L_0 and N_0. As predicted in (2), the communication overhead increases linearly with L_0 when N_0 is fixed, and increases linearly with the logarithm of N_0 when L_0 is fixed.

VI. Conclusion

In this paper, we considered the issues of the disclosure of the group dynamic information by the key management schemes in secure multicast communications. By exploiting the properties of rekeying process and the size of rekeying messages, we demonstrated that the inside attackers can successfully obtain good estimates of GDI. Further, an anti-attack scheme was proposed to fight against various attacks by employing batch rekeying and phantom users. We derived a set of performance criteria and provided a framework of selecting the parameters for the proposed anti-attack scheme. The proposed scheme was tested on user log data from MBone sessions.

References

[1] S. Paul, *Multicast on the Internet and its applications*, Kluwer Academic Publishers, 1998.

[2] D.M. Wallner, E.J. Harder, and R.C. Agee, "Key management for multicast: issues and architectures," Internet Draft Report, Sept. 1998, Filename: draft-wallner-key-arch-01.txt.

[3] M.J. Moyer, J.R. Rao, and P. Rohatgi, "A survey of security issues in multicast communications," *IEEE Network*, vol. 13, no. 6, pp. 12–23, Nov.-Dec. 1999.

[4] C. Wong, M. Gouda, and S. Lam, "Secure group communications using key graphs," *IEEE/ACM Trans. on Networking*, vol. 8, pp. 16–30, Feb. 2000.

[5] R. Canetti, J. Garay, G. Itkis, D. Miccianancio, M. Naor, and B. Pinkas, "Multicast security: a taxonomy and some efficient constructions," *Proc. IEEE INFOCOM'99*, vol. 2, pp. 708–716, March 1999.

[6] M. Waldvogel, G. Caronni, D. Sun, N. Weiler, and B. Plattner, "The VersaKey framework: Versatile group key management," *IEEE Journal on selected areas in communications*, vol. 17, no. 9, pp. 1614–1631, Sep. 1999.

[7] Y. R. Yang, X. S. Li, X. B. Zhang, and S. S. Lam, "Reliable group rekeying: a performance analysis," *Proc. of the 2001 conference on applications, technologies, architectures, and protocols for computer communications*, pp. 27 – 38, August 2001.

[8] K. Almeroth and M. Ammar, "Collecting and modeling the join/leave behavior of multicast group members in the mbone," in *Proc. High Performance Distributed Computing (HPDC'96), Syracuse, New York*, 1996, pp. 209–216.

[9] "http://ftp.cc.gatech.edu/people/kevin/release-data," .