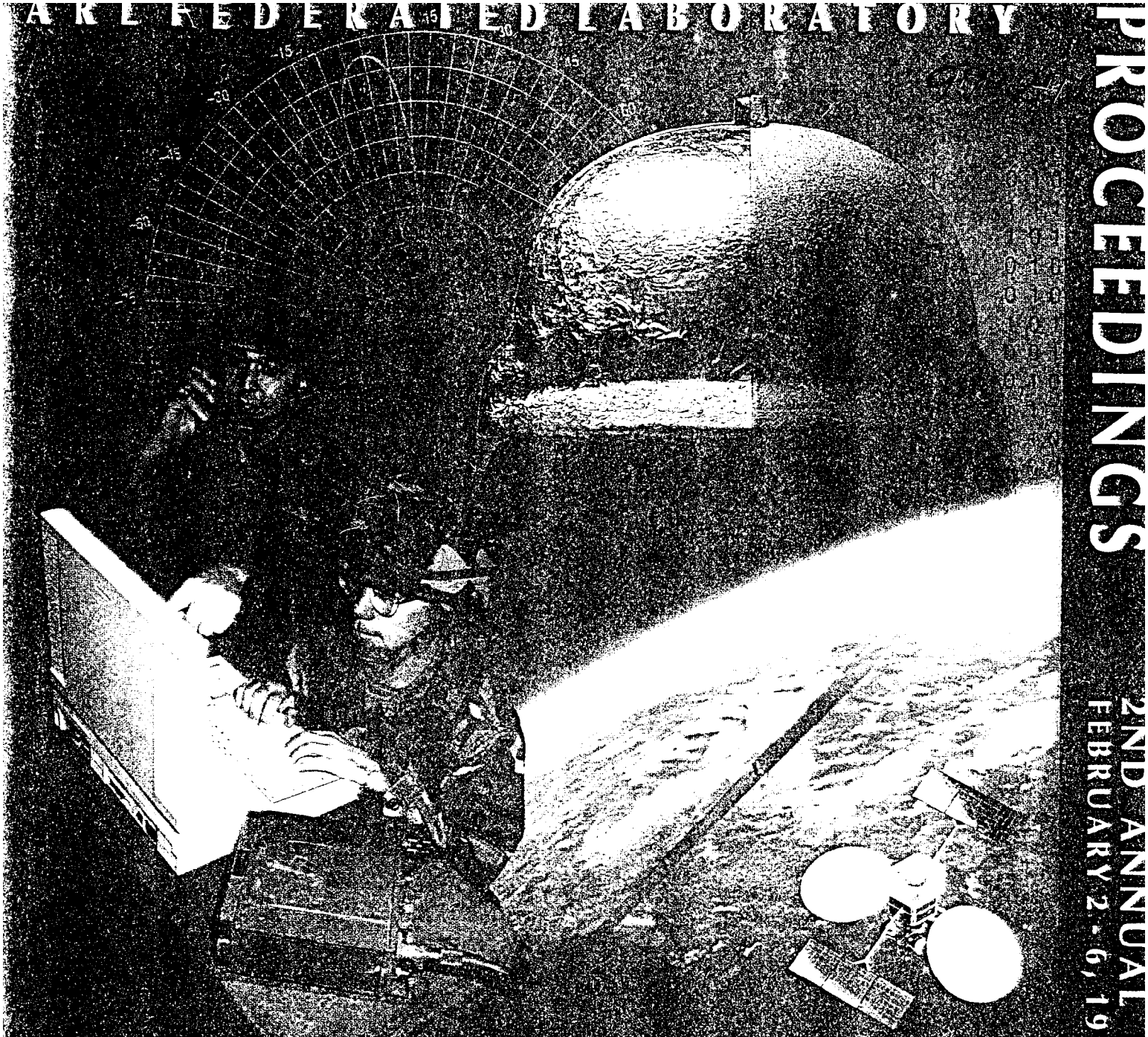
ARL FEDERATED LABORATORY



Advanced Telecommunications & Information Distribution

CONSORTIUM

PROCEEDINGS

2ND ANNUAL FEDLAB SYMPOSIUM

FEBRUARY 2 - 6, 1998 • COLLEGE PARK, MD

A SCALABLE EXTENSION OF GROUP KEY MANAGEMENT PROTOCOL *

R. Poovendran, S. Ahmed, S. Corson, J. Baras
Institute for System Research
University of Maryland
College Park, MD 20472

ABSTRACT

This paper presents a robust, scalable extension to the recently proposed multicast Group Key Management Protocol (GKMP) ([1], [2]), in terms of security administration. The GKMP has two major security related problems, (a) lack of any mechanism to remove a compromised group administrator, (b) lack of scalability. We are able to remove a compromised single panel member from generating the group keys by setting the panel members with shared authority to generate the group keys. We then introduce the sub-controllers who have all the functionalities of the group control panel except the authority to generate the group keys. The sub-control panel helps scalability of the network in terms of the security operations. The sub-controllers are chosen using a threshold based clustering algorithm.

INTRODUCTION

Many military and Internet applications are based on multicast communication. These applications often need privacy and protection from intruders. In order to enforce the privacy and protection, these applications require cryptographic key usage for secure admission of new members and secure communication among existing members. The issue of exchanging the relevant keys in a multicast environment is an active area of research.

Apart from the secure joining and information exchange, there are other administrative duties related to security and communication that a group should provide. For example, mobility of the members is an issue in some multicast groups. In a highly dynamic group, members may decide to leave the group and rejoin at a later time. It is also possible that late arriving members get access to the previous data of the session in progress to get the context of the current communication.

*"Prepared through collaborative participation in the Advanced Telecommunications/Information Distribution Research Program (ATIRP) Consortium sponsored by the U.S. Army Research Laboratory under the Federal Research Laboratory Program, Cooperative Agreement DAAL01-96-0002"

Depending on the anticipated size and nature of the group, it may be possible to establish some mechanism to *control* the security and other administrative duties of the group. In a military environment, it is possible that a high command authority provides the necessary *key* to the selected members who will establish the group. This may be done in person in a one-on-one manner. Though this method is possibly secure, it is rather inefficient.

Another way to establish a secure group is to have a dedicated Key Distribution Center (KDC) to generate and issue high quality keys (symmetric). Everyone who wants to join a particular group will have to request an appropriate key from the KDC. After verification of the member's credentials, the KDC will issue the necessary symmetric key encrypted using the session key between the KDC and the member. This mechanism has the following advantage: if the group key is compromised, KDC can individually contact the members and rekey the group. However, this method has the following disadvantages: (a) regardless of the size of the group, every new member has to join via the KDC, (b) every time the key is updated, the KDC should be able to access all the members, (c) if the number of groups is large, which would be the case in the Internet, the KDC would be heavily over loaded, (d) if the KDC is compromised, then the credentials of all the groups will be accessible by the intruder.

The GKMP [1] allows a designated group member called the *group controller* to generate and distribute the keys for privacy and security. In doing so, the KDC is effectively replaced by another security entity (entities) which operates at the group level. Any node of the network can be a group controller as long as it is certified to do so by a Security Manager (SM). This approach essentially creates *virtual KDC* for each group.

The use of an entity other than KDC requires that additional security related concerns are addressed. When the KDC was used for group key generation and distribution, the authentication of the members at the time of joining was implicit since the KDC had to have authenticated the member before giving the group key to

the member. In the case of GKMP, the group controller need not have the information and/or the authority to authenticate a new member. Hence, the group controller needs to have the support of a secure entity that can perform the needed authentication. The GKMP assumes that such an entity exists and is accessible. Without this assumption, the key generation and key distribution to the group members can not guarantee the desired secure joining. The GKMP, however, is not able to deal with a compromised member and scalability. Our approach addresses these two critical issues in this paper. These are critical since the GKMP is supposed to be for secure multicast communication.

The paper is organized as follows: We first present an overview of the GKMP mechanism and identify some critical shortcomings of the GKMP. We then present the necessary modifications to the GKMP, including a clustering scheme.

BACKGROUND OF GKMP

Among the multicast key management protocols we know of ([1], [2], [3]), the GKMP is independent of the architecture of the multicast network being used. The GKMP proposes to create grouped symmetric keys and distribute them amongst communicating peers. The GKMP has several desirable properties that are well suited for a multicast group since, (1) it is virtually invisible to operator, (2) no central key distribution site is needed, (3) only group members have the key, (4) it allows sender or receiver-oriented operation, (5) it can make use of any multicast communications protocols. The GKMP assumes that there is a Group Key Management (GKM) application available and executable by any node in the network.

We note that although the GKMP has been able to remove the constraint of having a centralized KDC, it requires the existence of a similar mechanism and assumes that this role is performed by a SM. It is necessary to have this *central entity* since, at the beginning, everyone has to be authenticated and there has to be a mechanism to validate a new member's identity. For this purpose, there has to be a central location or a replicated central location which has all security parameters and the other credentials. This functionality is independent of the key generation for a particular group. Clearly, more than one group may have to verify the same security parameter information from the SM during a new membership application process.

The GKMP group security administration consists of the following steps:

1. SELECTION OF THE GROUP KEY CONTROLLER

Independent of whether the system is sender-initiated or receiver-initiated, the initial task is the identification of the group originator. The originator of a multicast group first obtains a certification from a trusted entity verifying that the originator is responsible for generating and distributing the group key. The originator then sends the membership list to the GKM.

We note that the specification of GKMP is quite flexible since it is not tied to any particular naming convention or crypto scheme. We also note that before sending the member list to the GKM, the originator has to verify each join request using the *security manager* which is still a central authority. This authority may be duplicated but can not be removed like the KDC due to the very nature of the function it has to perform. Therefore, implicit in the assumption of the member list generation is a *central security* entity for final authentication.

2. GROUP KEY CREATION

The initial member list is used by the GKM application to generate the Group Key Packet (GKP). The GKMP currently allows the GKM to arbitrarily select and contact any one member from the list provided by the originator and generate the GKP. The GKP contains the current Group traffic encrypting key (GTEK) and the future Group Key Encrypting Key (GKEK). The GKMP allows GKM to identify itself as the group key controller, which is validated by the member participating in the GKP generation. The GKP has the following format as specified by the GKMP: $GKP = [GTEK_n, GKEK_{n+1}]$, where the index n denotes the number of times the GKP has been updated.

We note that the GKMP allows the GKM to choose the security parameters. We note that this mechanism of GKMP is neither necessary nor desirable. We propose to have the group originator specify a desired level/range of security for the group.

3. GROUP KEY DISTRIBUTION AND REKEYING

Once the GKP is generated, the group controller distributes it to every member of the group. In distributing the GKP, the group controller first contacts every member, verifies the security parameters of the member and then generates a Session key Packet (SKP) for that member. The SKP contains TEK and KEK and has the form: SKP = [STEK, SKEK]. This SKP

is first given to the member encrypted using the public key of the member. The controller then uses the SKEK of that member to encrypt the GKP and then creates a Group Rekey Package (GRP) for that member. The GRP with the usual notations is of the form: $GRP = \{\{GKP\}^{SKEK}\}^{PK_{controller}}$.

When the group has to be rekeyed, the GKMP allows GKM to contact any member of the group and generate a new GKP and then broadcast it after encrypting using the old GKEK.

This approach has the following problem as pointed out in [1]: if a current member is compromised, the proposed method of rekeying will not be effective since the compromised member still has the access to the old GKEK and can decrypt the new GKP. One expensive way to eliminate the compromised member is to generate the GKP and then contact each member individually and distribute the keys. This may be very costly as pointed out by the GKMP.

MAJOR SECURITY RELATED SHORTCOMINGS GKMP

Although the GKMP allowed "peer-managed" secure keying mechanism and allowed any member to be group controller, only one member is allowed (by some election mechanism) to serve as a group controller at any given time. Allowing one group controller to generate the keys leads to some problems discussed below.

1. SINGLE POINT OF FAILURE

Although the group controller takes the partial role of KDC in generating the desired keys, the role of group controller is being performed by a single node at anytime. Hence the failure of the group controller node will lead to denial of service for new membership and group and session key generations are temporarily terminated. If the controller fails during the key generation period, the whole group has to be rekeyed since the keys have fixed lifespans. This is can be a serious threat if the group is quite large. Apart from physical node failure, if the controller is compromised, currently there is no mechanism in GKMP to immediately prevent the controller from further generating group keys until another member is elected as the group controller by the security manager. The GKMP does not have any mechanism -other than rekeying the group- to remove a compromised controller. We will show that it is possible make sure that no compromised member takes the role of controller once identified as compromised.

2. LACK OF SCALABILITY IN SECURITY RELATED ADMINISTRATION

Since a single controller is responsible for generating the keys and also validating the entries of the new members, as the group size becomes larger, the node having the role of group controller will be heavily loaded. Though having a -node other than the conventional KDC- act as the group controller helps to distribute the load on different nodes for different groups, each group still has a single node functioning in the role of KDC. Although the GKMP proposed to allow any member to have the ability to perform operations other than the key generation, this has a serious problem when members are compromised. There is no way to prevent a compromised member from being able to permit intruders into the group. Therefore, the GKMP in its present form has scalability problem.

PROPOSED SOLUTIONS FOR ROBUSTNESS AND SCALABILITY

Our solution to the shortcomings of GKMP is discussed below.

1. ROBUSTNESS OF THE GROUP CONTROLLER

In providing security, we need to ensure network survivability and continuous service in the presence of any node failure. Failure may be due to physical failure at the node, link failure and/or existence of a compromised group controller. As noted earlier, having a single node as group controller leaves that node as the single point of failure of the whole security network at-least for a short period of time.

Considering the fact that the purpose of the multicast communication is to possibly serve a large number of members, it is of interest to provide a group-control mechanism that can survive a single point of failure. Apart from functional survivability, the new group-control mechanism should also be able to prevent a compromised group-controller from being able to generate any future keys for the group. These features are not present in the current form of GKMP. In providing robustness, a criteria for robustness has to be chosen. We opted to choose the Byzantine robustness model. We are able to achieve the desired properties of the group-control mechanism as described below.

First, the notion of group controller is changed to a *panel* of controllers. This panel consists of three members at any given time. Among these three, one will serve as

the active group controller, with the group keys being generated by two panel members with the constraint that no two panel members may participate in consecutive key generations. This approach allows every panel member to have only shared key generation authority. At least two members of the control panel have to agree in order to rekey the group. The only exception comes when the group is of size less than three in which case the communication reduces to a pairwise event or no communication at all.

The proposed scheme ensures that (a) a single member of the control-panel can leave the group at any time, (b) any physical failure (node, link etc.) of a single controller does not affect the immediate functioning of the group, (c) if any one controller is compromised, the other two may inform the SM to replace the compromised controller of any active duties, (d) ACL, RCL, membership verification and routing need not be performed by the same member of the control panel, (e) since the number of controllers is three, we can randomly send data packets to the panel members with some probability distribution. This prevents an attacker from tapping into a single line to get the information.

Replacing a single group controller with a panel of three members adds more functionality to the panel and reduces the probability of failure of the whole group controller panel. As in the case of the single group-controller, any member can serve as a panel member. However, at any given time only three members will be serving on the panel. The panel membership may be updated due to one of the following reasons, (a) a panel member may leave the group region or the communication range in a dynamic environment, (b) node or a link failure may prevent one or more panel members from playing the role of the controller, (c) one or more members of a panel are compromised.

If a member has to leave the panel, the leaving member has to inform the rest of the panel members and the SM about it. This policy requires a voluntary service that may not be feasible under circumstances like a node failure. So there has to be another mandatory mechanism to check the aliveness of the three members without having to involve the rest of the group.

In order to ensure that the group functionalities are independent of a particular member in the panel, we suggest that all three members have a mechanism for keeping track of the other current panel members. This may be achieved using a periodic "hello" message mechanism. At any given time, two panel members can send the check packet with the time stamp, signed using the private key. The receiver checks the message, encrypts the time stamp in the received data packet, signs the encrypted message using the private key and sends it back to the sender. If all the members are functional, this will happen before the allocated wait time. If there is a problem with one of the members, the member will fail to participate in this process.

2. SCALABILITY OF THE CONTROLLER PANEL

If one member is compromised, then the other two members can inform the SM to effectively remove the compromised panel member. We note that if more than one panel member is compromised, the protection mechanism we have proposed is at the same level as the GKMP or KDC in terms of the ability to deal with a compromised member. However, the probability of more than one panel member failing at the same time is lower than that of a single node failure.

We note that authentication, verification, join-authorization, session key parameter negotiation and distribution have to be scalable for a dynamic multicast group. These security operations are independent of the current GKP generation. Hence, these functions need not be performed by the core-control panel. This observation is the key to scalability. We note that only the GKP generation is tied to the group controllers.

One way to achieve scalability is to allow any member to perform all these operations (GKMP suggests this). However, there is a security threat in doing so. As noted in the case of the controller in GKMP, if any member is allowed to perform many of the security operations such as key exchange, ACL, RCL updates, etc., there is no way to remove or prevent the compromised member from allowing intruders into the group and further compromising the group.

To effectively scale the key exchange, ACL, RCL and other issues, we propose to have the group first segmented into clusters and each cluster managed by a *sub-controller* panel of three members which has all the authorities of the main control panel *except* the GKP generation. Every sub-panel member is allowed to have only shared authority and no two panel members are allowed to perform any operation consecutively. We note that if one sub-control panel member is compromised, the member can be effectively stripped of the control functionalities by SM without affecting the cluster functions.

In our scheme, only one panel is identified as core and is responsible for generating the group key packet

which includes the GTEK and GKEK. The core panel then transmits this information to the cluster panels which will then use the information in generating the session keys, etc. The need for cluster or sub-group being transparent to the existing members forces this kind of structure. Having a global GKP enables a member of a group to reach the other members without having to know in which cluster the members reside.

We note that of the sub-cluster panel members can perform several of the security-related and other administrative tasks. The sub-group controllers or the cluster controllers perform the following tasks: (a) serve as routers, (b) receive the group key packet from the core-control panel for re-keying, (c) authenticate and issue new memberships; negotiate with new members and generate the necessary session key for the new members, (d) keep local ACL, RCL, and other group management activities, (e) serve as loggers.

Another approach is to let each of the clusters be autonomous. If any particular cluster is known to be compromised, that can be separately rekeyed without affecting the rest of the group. But this comes with the cost that each cluster has to use another key to encrypt the key used by the cluster for intra-cluster communication.

3. FORMATION OF CLUSTERS AND SUB-GROUP PANELS

The size of any particular group is the deciding factor in formation of clusters and hence the *sub-group panels*. Independent of the type of the multicast group, the clustering involves the following steps: (a) cluster formation, (b) cluster splitting, (c) cluster merging, (d) cluster joining.

Cluster formation involves setting up the cluster *sub-control panel* id, establishing a cluster id, and setting up the link to the neighboring clusters, if any. One way to pick the initial *sub-group panel* is to pick three members with the highest ids. Since each member has to have a unique id, this is a simple mechanism for starting the panel. Thereafter, the panel may follow a policy to update the panel in a periodic or need-based method.

In a given cluster, a threshold mechanism can be used to find if the number of members exceeds a fixed threshold. Only the local *sub-control panel* is allowed to make the decision to split the cluster. Once the cluster split decision is made, the *sub-control panel* chooses three members from the current cluster and informs that to the SM. The SM then assigns the authority to the new sub-control panel and informs the group by multicasting the new sub-panel information. The cluster id's can be based on the crypto identities of the panel members.

As pointed out earlier, since the key exchange is transparent to the cluster a member is currently in, if the member migrates to another cluster, there is no need for a new key exchange as long as the member has the current GKEK and GTEK. If GTEK is not used, any group member migration will force a new authentication.

CONCLUSION[1]

In this paper we proposed a robust, scalable extension to the Group Key Management protocol for multicast communication. By replacing the single group controller by a panel, we reduced the threat of the node failure. This scheme also helps in removal of a panel member if the member is compromised. We then introduced the *sub-groups* that had all the security related authorities except the generation of the Group Key Packet. The sub-group panels help in scalability of the secure joining, authentication, and revocation. The following issues are addressed in a related material to be presented elsewhere for such a key management protocol, (a) GKP generation, (b) removal of a compromised member.

REFRENCES

[1] H. Harney C. Muckenhirn, "GKMP Architecture", RFC 2093, July 1997.

[2] H. Harney, C. Muckenhirn, "GKMP Specification", RFC 2094, July 1997.

[3] A. Ballardie, "Scalable Multicast Key Distribution", RFC 1949, May 1996.

[3] R. Poovendran, S. Corson, J. Baras, "A Scalable Multicast Key Management Protocol (SMKMP) for Mobile Agents", to be presented at the *Mobile Agents and Security Workshop*, Oct. 1997, UMBC, Maryland.

[1] "The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government.