EFFICIENT PRIVATE GROUP COMMUNICATION OVER PUBLIC NETWORKS

by

Lakshminath Reddy Dondeti

A DISSERTATION

Presented to the Faculty of

The Graduate College at the University of Nebraska

In Partial Fulfillment of Requirements

For the Degree of Doctor of Philosophy

Major: Computer Science

Under the Supervision of Professors Sarit Mukherjee and Ashok Samal

Lincoln, Nebraska

November, 1999

UMI Number: 9953893

Bell & Howell Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

EFFICIENT PRIVATE GROUP COMMUNICATION OVER PUBLIC NETWORKS

Lakshminath Reddy Dondeti, Ph.D

University of Nebraska, 1999

Advisers: Sarit Mukherjee and Ashok Samal

Private group communication is an essential requirement of many applications such as real-time stock quote distribution, secure multimedia conferencing, panel discussions and virtual private networks. We use encryption for privacy and multicasting for efficient group communication. Distribution of encryption keys to authorized members of a multicast group is the crux of the problem. Key distribution schemes must scale to groups of large sizes. When group membership changes, we need to change encryption keys and send them to current authorized members only. Efficient key distribution to a large and dynamic group is a formidable challenge. This dissertation provides a framework for private group communication on public networks.

Based on the number of senders, secure group communication can be classified as *one-to-many*, *many-to-many* and *few-to-many* communication. We propose three protocols for secure group communication, addressing each of these categories separately. We use distributed group management for efficient management of large and dynamic groups. Our protocols scale well to groups of large sizes. They distribute group management overhead evenly among all entities of a multicast group. We do not expose secret keys to third party entities in the public network. We prove the correctness of our protocols. We show that they are immune to collusions.

Simulation results further reinforce that our one-to-many and many-to-many protocols evenly distribute key management overhead among the entities of a multicast group. We select group membership traces of sizes varying from hundreds of members to thousands of members from past MBONE sessions for our simulations. The results show that our

protocols scale well to large groups. As group size increases our protocols perform better than other solutions in the literature. Our simulation of existing many-to-many protocols demonstrate that our protocol evenly distributes overhead among all senders, unlike other contemporary solutions which tend to overload a small subset of senders.

Acknowledgements

I take this opportunity to thank all those who motivated me to pursue a doctoral degree and those who helped me get through this endeavor.

Professor Ashok Samal reinforced my desire to continue my education and convinced me to pursue a PhD at the University of Nebraska. I thank him for that and for his help and support in meeting all the University requirements for the degree during these past few years. I am grateful to him for introducing me to Professor Sarit Mukherjee. Without Sarit's help professionally, and financially during the past few months I could not have completed my dissertation successfully. I am indebted to Sarit for his support during the past two years. Professor Deogun has been a constant source of advise and encouragement throughout my graduate studies and I am thankful to him. Thanks to my committee members Professors Sayood, Ramamurthy and Seth for their comments and constructive criticism.

I am grateful to Nancy Khawand and Donna McCarthy who taught me a thing or two about successful professional life. Thanks to the Computer Science department staff especially Marilyn and Deb for their help in administrative matters throughout my stay at UNL.

My friend Sekhar motivated me to treat college as a place where I should further myself intellectually and he indirectly influenced me to pursue graduate study. Phani, Suresh, Vijji, Tharun, Anil and Ramu have been very supportive of me throughout my adult life and I take this opportunity to thank them. They helped me get through college in one piece (mentally). Phil and Charles have helped make my stay in Nebraska a pleasant one. I learned a lot of systems administration quirks from them. Thanks guys.

Thanks to Padma and my parents for understanding and being patient while I was working on my doctoral degree. My parents did not like many of my choices in life, but they supported me all the same and I thank them for their love. Jaya has been a wonderful source

of encouragement and I thank him for his confidence in me in personal and professional life. During the last few months, Vagdevi and Prasad hosted me kindly and that helped a great deal in finishing up my dissertation.

Thanks to Sridevi for knowing when to push me and when to encourage me. I started my pursuit before I met her, but I could not have continued without her love, help and patience.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

We consider the problem of using a public network, for example the Internet, for private communication between a group of members. For confidentiality of data, we use encryption. All authorized members and only they should have access to the encryption key. The problem of distributing the secret encryption key to a potentially dynamic set of authorized members is non trivial. If the key distribution protocol fails, even the best encryption algorithm becomes useless. We address key distribution to a group whose membership is very large (comprising of thousands to tens of thousands of members) as well as dynamic. In other words, our problem translates to maintaining and distributing an encryption key for secure communication between the authorized members of a large and dynamic group. The entity or entities responsible for key management need to monitor group membership changes.

Only the authorized members of the group at a given instance in time should have access to the current group key. More formally, we must maintain perfect forward secrecy [31, 45] in the group. In groups with frequent membership changes, joining members must not get access to past data and departing members must not get access to future data, to guarantee perfect forward secrecy. We need to change the key and send the updated version to the new members only, each time there is a membership change. This “rekeying” process must

be scalable i.e., efficient.

Naive approaches to rekeying require the group manager to securely send encryption key(s) to each member, separately. Rekeying overhead in such approaches is proportional to group size. We incur this cost of rekeying each time there is a membership change. Thus, in large and dynamic groups we will frequently incur overhead proportional to group size. For efficient secure group communication, key distribution overhead must be independent of the size of the group.

We classify secure group communication based on the ownership of data. It is reasonable to assume that senders would like to control who receives the data. In *one-to-many* secure group communication, one sender has data to send to a very large and dynamic group. Real-time stock quote distribution is an application. Private conferencing on the Internet, where all members are senders requires a *many-to-many* secure group communication protocol. Finally, *few-to-many* secure group communication is useful for private panel discussions on the Internet, where a few senders speak and a large number of members listen in on the conversation.

1.1 Motivation

The Internet has been an extremely popular resource for one-to-one communication in the past few years. More recently, several group communication applications have sprung up. For example, data replication or mirroring requires one-to-many communication between the data distribution site and the mirrors. Similarly “push” applications such as webcasting [33], which push data to users rather than each user separately “pulling” the information are becoming popular. The Internet may also be used for many-to-many communication, where more than one member sends data to the group. Distributed interactive simulation, multimedia conferencing are examples of such applications.

To illustrate the benefits of a “push” approach over a “pull” approach for data transmission, consider real-time stock quote distribution. Currently stock quote distribution via the world wide web employs a “pull” approach as opposed to a “push” approach. For example, a couple of traders from neighboring buildings in San Francisco may be downloading the same information at the same time from a source in New York City. In the pull mechanism two separate copies of data travels on the network, which is a waste of resources. A push method on the other hand avoids such duplications.

Multicasting is an efficient solution for group communication on the Internet. Senders need not make copies for each destination. It ensures that data travels exactly once on each network link in reaching multiple destinations. Thus multicasting utilizes network resources efficiently and reduces the load at the senders. We can efficiently implement push applications using multicasting.

Group communication applications would be financially more attractive to the industry, if the communications can be private. The challenge is in instilling a mechanism to ensure that only the users who paid for the service get access to the information. Mechanisms for efficient management of large and dynamic groups would make it easy to implement pay-per-use applications, which are attractive for both service providers and their users. Real-time stock quote distribution, private multimedia conferencing on the Internet, panel discussions on the Internet are some popular applications of secure group communication.

Multicasting is an efficient solution for data transmission. For efficient private group communication, key transmission must also be efficient. This motivates the need for scalable key distribution mechanisms. Rekeying overhead must be scalable for effective use of multicasting for private group communication.

Several protocols exist to address security in data networks with respect to unicasting [19, 31]. Unfortunately, these protocols cannot be easily extended to protect multicast data. Along with its advantages, IP multicasting poses some challenges in adding new

functionality such as reliability [13], congestion control [46] and security [33]. In particular, secure multicasting poses several formidable problems which do not arise in securing unicast data transfers [6]. First, IP multicast addresses are not private which enables any interested host to join the multicast session without any hindrance. Next, multicast data are transmitted over many channels of the network which presents multiple opportunities for attacks such as eavesdropping. Furthermore, any host in the Internet can send irrelevant data to the multicast group, which may cause congestion and thus constitute a denial of service attack. The universal knowledge of multicast addresses also allows any host to pose as a member of the group, thereby allowing it to gain access to the multicast data. Finally, adversaries can possibly disrupt the multicast protocol itself by posing as a legitimate member of the group. Secure multicast protocols should address these issues.

Several protocols have been proposed to support secure multicasting [5, 11, 12, 21, 25, 34, 35, 36, 44, 48, 49], in the recent past. Some of these protocols are not scalable [5, 21, 25], hence they are not efficient for large groups. Others use a centralized entity for group management [11, 12, 48, 49]. The centralized entity is a performance bottleneck. Few others use "trusted" third party entities for secret key distribution [5, 34]. It is not desirable to expose secret keys to any entities other than the authorized members. Another class of protocols [11, 12] known as the flat schemes [47], choose to compromise security in favor of performance. They cannot avoid or eliminate colluding hosts efficiently. In these flat schemes, members evicted from a group can combine the secret information they know to deduce the current group key. This dissertation addresses the above issues and proposes scalable protocols for secure group communication.

1.2 Issues and challenges in key distribution

In a typical secure unicast protocol, the sender and receiver verify each other's authentication and communicate using a secret-key [45] or public-key [45] encryption mechanism. When either party leaves, the secure communication session is terminated. Such a scheme cannot directly be extended to handle secure data transmission to multiple receivers. Simple solutions such as establishing separate secure channels between each receiver and the sender are highly inefficient. Moreover, such solutions are not compatible with the scalable nature of multicasting, since the protocol processing overhead at the sender increases with the addition of each receiver to the group. Other traditional solutions suggest the computation of a shared secret using information distributed off-line to individual receivers. These protocols require that group membership be known in advance and are computationally inefficient.

1.2.1 Multicast security threats

The common knowledge of multicast addresses allows any receiver interested in receiving the multicast data to subscribe to the multicast group. It is up to the secure multicast protocol to regulate group membership. Typical multicast data travel through many network channels before reaching all the corresponding group members. This increases eavesdropping opportunities to possible adversaries. These kinds of security attacks where adversaries try to gain access to the data without really disrupting the secure multicast session are called passive attacks [45]. Potential adversaries may use the data collected by means of eavesdropping for cryptanalysis purposes.

Uncontrolled group access allows any host in the global network to send multicast data to a multicast group, which may cause congestion. This presents an opportunity to mount a denial of service attack against the group. Any host in the Internet may pose as another host

that is a member of the group. It can send data, receive data or acquire access to the secret keys posing as a legitimate member of the group. Such an attack is called masquerading. Finally, an adversary can intercept data possibly by eavesdropping or other means and replay it at a later time. Masquerading and replay attacks call for the receivers to be able to determine the source of multicast data. Replay attacks in particular require the receivers to be able to determine the time at which the data were sent. All these attacks are termed as active attacks since they disrupt the multicast session [45].

1.2.2 Components of a secure multicast protocol

Group membership control and secret key distribution are the two major components of a secure multicast protocol. If the multicast group membership is dynamic, i.e., if the group members join or leave frequently during the course of a multicast session, the encryption keys need to be updated accordingly. We change secret keys so that members do not get access to multicast data sent before they join the group or the data sent after they leave the group. Rekeying is necessary whenever there is a membership change in the group. This repeated process must be scalable for the secure multicast protocol to be efficient.

Group membership control is the most basic component of a secure multicast protocol. It allows only the authorized hosts to join the multicast group, guarding against otherwise unilateral subscriptions by arbitrary hosts. Conventional solutions suggest the use of access control lists. The sender or an authorized third party can maintain either an inclusion or an exclusion list of hosts in the Internet corresponding to a multicast group. Each time a host requests to join the multicast group, the sender or the third party checks with the access control list to determine whether the host is authorized to join the group. Alternatively, capability certificates can be issued by a designated third party to the receivers. Capability certificates authenticate the receivers and authorize them to participate in a multicast group. The receivers present the capability certificate to the sender or an authorized third

party to gain access to the group. The third party servers may be replicated for reliability, availability, and efficiency.

Key distribution is the heart of a secure multicast protocol. An effective key distribution scheme must not yield to active security attacks such as masquerading. Many provably secure cryptographic mechanisms such as secret key and public-key mechanisms exist to support secure communication. A secure multicast protocol's role is to ensure that none of the keys are compromised during the key distribution phase. Ideally, the key distribution protocol must be such that hosts can join and leave the multicast group without affecting the other members of the group. In practice, addition and removal of hosts must take place affecting as few members of the group as possible.

Some of the other challenges to multicast security explained earlier, viz., integrity and replay attacks can be solved using digital signatures, time-stamps, message digests [45]. Efficient flow signing and verification procedures for authenticity and non-repudiation also exist in the literature [20, 50].

1.2.3 Classification based on group control

Secure group communication has applications with a variety of characteristics and requirements. Applications involving secure data distribution typically have a single sender and a very large and dynamic group membership. For example, a brokerage service is the sender in real-time stock quote distribution. Their customers are the members. We refer to such communication as *one-to-many* communication. Since the sender owns the data, it should have control over who receives the data. Only the sender and the authorized members are the trusted entities. No third party entity may be trusted.

In private conferencing all members are senders and all have valuable data to send, possibly over a public network. To keep the conversation private, senders may use encryption. In such *many-to-many* communication, it is desirable that all authorized members have

equal control over the group. All authorized members must be trusted equally.

Panel discussions and corporate conferencing where only a few people speak while most others listen in are examples of *few-to-many* communication. The few have the data to send and therefore should have equal control over the group. The senders and authorized members should be treated separately. In particular, the protocol must keep members from sending data to the group. On the other hand, it may make provisions for limited communications by the members to the group.

1.2.4 Group management and scalability

Group management consists of access control and key distribution and rekeying during membership changes. Two popular ways to achieve scalability are hierarchical subgrouping [34] or a logical hierarchy of keys [11, 48, 49]. Hierarchical subgrouping approaches divide members of the multicast group into a tree of subgroups. By limiting rekeying to a subset of members we reduce protocol processing overhead. More precisely, the number of messages and encryptions or decryptions per membership change decrease. Hierarchical subgrouping distributes subgroup management responsibilities to subgroup managers (SGM). SGMs may be third party entities such as ISP routers. In addition to subgroup management they are also responsible for forwarding data encryption keys. Finally note that a centralized group controller retains control of the group and is the root of the key distribution tree.

We use a virtual key distribution tree to best explain the approaches using a logical key hierarchy. Figure 1.1 is an example of a virtual key distribution tree. The square nodes represent members and the circles represent keys. Each member receives all the keys in its path to the root. All members have the root node key and the sender uses that key as the data encryption key. In Figure 1.1, Alice receives all the keys of the grey nodes. To illustrate how logical key hierarchy may be efficient for rekeying, consider Alice leaving

the group. The group manager needs to encrypt the new group key only once for all the nodes in black, i.e., using the root node key of the right subtree.

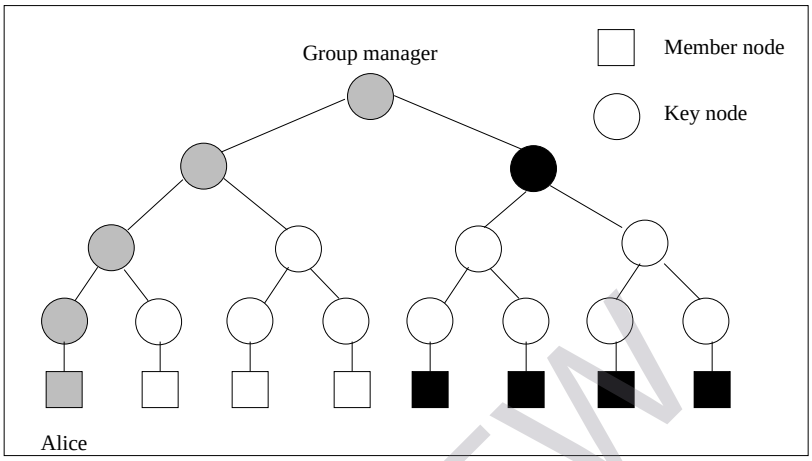

Figure 1.1: Logical key hierarchy

In schemes that use a logical key hierarchy all members have the root node key. The sender uses it for data encryption. Clearly the root key needs to be changed each time there is a membership change. In other words, each join or leave affects all members of the group. This is known as as "1 affects n" scalability problem in the literature [34]. Hierarchical subgrouping is offered as a solution to avoid "1 affects n" scalability problem. A close look at the problem, the solution and the security requirements reveal that perfect forward secrecy and "1 affects n" scalability are complementary. To maintain perfect forward secrecy, we must change the group key every time there is a membership change. In other words maintaining perfect forward secrecy results in "1 affects n" scalability problem.

Scalable key distribution overhead, i.e., number of encryptions and messages during rekeying should be the focus. We note that distributed group management is important to design protocols that are efficient. It is important to eliminate performance bottlenecks. For example notice that in logical key hierarchy, the sender or the group manager is solely responsible for rekeying. Secure multicast protocols must avoid such concentrations of overhead at one or a few entities. It is desirable that group management and rekeying

overhead is distributed among the various entities in a secure multicast group.

In a secure multicast group with several senders, all senders who desire to protect their data must have equal control of the group. Along with group control, they should share the overhead involved in dynamic group management and key distribution evenly among themselves. Distribution of dynamic group management tasks to all senders allows localized processing of joins and leaves, thus avoiding global flooding of control traffic. Distribution of group management tasks also avoids performance bottlenecks and eliminates single points of attack in a multicast group.

In the presence of multiple senders, all of them must be trusted equally and the group must be operational as long as at least one sender is active. Notice also that each sender intends to protect the data it owns and send them to authorized members of the group only.

In secure few-to-many communication, only the senders are allowed to send/receive data while the members (receivers) are allowed to receive only. It is desirable that only senders have control over who receives the data. Members may assist in group management to reduce overhead on the senders. Some applications may have members that need to send limited amounts of data infrequently. Such data transmissions from the members should be moderated by a designated controller(s). The senders may act as the moderators.

1.2.5 Collusions

Key distribution mechanisms change secret keys known to a member after it leaves. This is to ensure that the departed member can no longer access the data being sent to the group. However, this is not enough. It is possible that several members that have left may conspire to break the security of the group. While we know that we are safe from their individual knowledge, we are not aware of the capacity of their combined knowledge.

In general, we must ensure that hosts cannot use their combined knowledge to extract secret information that none of the individual members of the subset already knows [28].

These hosts may be active members of the multicast group or members that were once part of the multicast group. A secure multicast protocol must avoid collusions. At least it should be able to detect colluding entities and eliminate them efficiently.

1.3 Research contributions

A scalable protocol for secure one-to-many communication: We propose a scalable key distribution scheme for secure one-to-many communication. For efficient and secure group communication, the sender sends encrypted data via multicast (example: MBONE). For efficient group management, we divide the authorized members into hierarchical subgroups. Subgroup managers (SGM) alleviate group management load at the sender or the group manager. We forward data encryption keys without exposing them to third party entities. We prove the correctness of the protocol. We analyze possible collusions and prove that our protocol is immune to collusions.

Comparison of one-to-many communication protocols: We compare existing scalable one-to-many communication protocols with DEP, using simulation. For group membership behavior we use real world group membership traces of past MBONE sessions.

A protocol for scalable secure many-to-many communication: We propose DISEC, which is an efficient solution for private conferencing over public networks. It divides group management tasks evenly among all the members. Each member has equal control of the group. DISEC is scalable. Rekeying requires only $O(\log n)$ messages, each carrying a single encrypted key.

We prove the correctness of the protocol and prove that it is not vulnerable to collusion. We also entertain the possibility of multiple simultaneous membership changes