

# Mykil: A Highly Scalable and Efficient Key Distribution Protocol for Large Group Multicast

Jyh-How Huang and Shivakant Mishra  
Department of Computer Science  
University of Colorado, Campus Box 0430  
Boulder, CO 80309-0430, USA.  
Email: {huangjh|mishras}@cs.colorado.edu

## Abstract

*Mykil is a new key distribution protocol for secure group multicast applications. It has been designed to be efficient and scalable to large group sizes. It is based on a combination of group-based hierarchy and key-based hierarchy systems for group key management. Important advantages of Mykil include a fast and efficient rekeying operation for large group sizes, continuous availability of the key management service in a disconnected network environment, an ability to map the group structure to the underlying network infrastructure, robustness, and support for user mobility and smaller hand-held devices.*

IP multicast is increasingly being used to construct important applications that provide important services over the Internet today. Examples of such applications include pay-per-view programs, video-on-demand services, frequent stock quote updates, video conferencing, discussion forums, and advertising. IP multicast does not provide security guarantees by itself. Instead, support for *secure multicast* is built on top of IP multicast. A secure multicast protocol consists of an admission control mechanism to determine who can join a secure multicast group, and a secure dissemination mechanism for disseminating multicast data while preserving its authenticity, integrity, and confidentiality. The admission control mechanism typically consists of member authentication and registration, and the secure dissemination mechanism typically consists of using an appropriate encryption mechanism. Thus, a *secure multicast group* is a multicast group in which members register and authenticate themselves with a designated registration authority, receive a set of *cryptographic key(s)*, and use these keys to encrypt the multicast data that they send and decrypt the multicast data that they receive. The registration and authentication process, along with the use of encryption for multicast data ensures that only legitimate entities

know the cryptographic keys, and hence can send or receive multicast data.

After registration, users contact a key management server using the key(s) and other materials obtained from the registration server. A key management server manages a set of cryptographic keys used for various purposes in a secure multicast group, e.g., one or more *group key(s)* that is (are) used to encrypt and decrypt multicast data. It stores these keys, updates them when certain events occur, and distributes them to the group members using a key distribution protocol. The process of updating and distributing cryptographic keys is called a *rekeying* operation. Rekeying is required in secure multicast to ensure that only the *current* group members can send encrypted multicast data, and decrypt the received multicast data.

In this paper, we focus on large multicast groups with frequent membership changes, i.e., groups consisting of a significantly large number of members (100,000 members or more) with members joining or leaving quite frequently. There are a large number of multicast applications that exhibit these characteristics. For example, a popular pay-per-view program can have a very large number of subscribers, or a popular discussion forum can have a very large number of participants at certain times. When a group is large, the cost of key management can become prohibitively expensive. This is because a rekeying operation requires distributing various keys, including group key(s), to all group members. If this is done naively, it may require  $O(n)$  messages, where  $n$  is the number of members in the group. Furthermore, if a rekeying operation is performed after every membership change, and if the membership changes are frequent, key management will require exchanging a very large number of messages per unit time.

We propose a new protocol called Mykil (**M**ulti-**H**ierarchy **B**ased **K**ey **D**istribution Protocol) for managing cryptographic keys in large multicast groups that exhibit frequent membership changes. Mykil cleverly combines

two different types of hierarchy schemes—group-based hierarchy and key-based hierarchy, to provide an efficient and scalable solution for key management in large multicast groups. Mykil borrows several interesting ideas from the earlier work done in the area of key management for large group multicast, and provides a solution that is better than the previous solutions.

Mykil provides several useful features. First, it is highly scalable in terms of group size. A member join or leave event affects only a very small subset of group members. Second, it provides a very efficient and fast rekeying operation by ensuring that key updates take place at only a small number of group members during a member join or leave event. The cost of rekeying operation is further reduced by batching member join and leave events. Third, the group organization maps quite well to the underlying network infrastructure in Mykil. Fourth, it is designed to support group members that access a multicast service using small devices such as PDAs or cell phones that have limited resources. This is done by minimizing the memory, bandwidth, and CPU requirements for key management functions at different group members. Fifth, Mykil is designed to support both static and mobile group members. Finally, the design of Mykil ensures that the key management functionality is robust and remains available to all group members even when the underlying communication network partitions.

Mykil is based on Iolus and LKH. It uses the idea of group-based hierarchy of Iolus to divide a multicast group into several smaller subgroups called *areas* with a designated area controller (AC) for each area. There is a separate *area key* for each area. Different areas are linked with one another to form a tree structure, with ACs providing the links—an AC of an area *A* is also a member of another area *B* (area *B* is *A*'s parent in the tree-structure organization). A group member belongs to exactly one area. Like LKH, Mykil builds a tree-structured hierarchy of cryptographic keys called *auxiliary-key tree* in each area to facilitate key distribution to the area members. The area controller of an area serves as the root of the auxiliary-key tree of that area, and each member of this area is associated with a different leaf of this auxiliary-key tree. Multicast data propagation in Mykil is identical to the multicast data propagation in Iolus. A group member multicasts data by encrypting it using a random key. This random key is encrypted using the member's area key and appended to the multicast message. To forward multicast data (that has been encrypted using a random key) to another area, an AC decrypts the random key, and reencrypts it using the other area's area key. Responsibilities of an area controller include: (1) managing cryptographic keys of its area, (2) forwarding multicast data, (3) managing authorization information to determine who can join the group, (4) maintaining the auxiliary key tree of its area, and (5) managing member join and leave events.

Mykil employs batching to reduce the overhead of rekeying operations in three ways: (1) aggregation of join events, (2) aggregation of leave events, and (3) aggregation of join and leave events. The main idea is that all join and leave events are aggregated at an area controller until a new multicast data is received. The keys are updated just before the multicast data is forwarded. An analysis shows that batching can save up to 40% of the bandwidth.

An important aspect of Mykil is that it is designed to be robust, support mobile group members, and support operation in a disconnected environment. Common failures such as communication partitions and node crashes can result in a loss of communication between a group member and the area controller of its area. Mobility of a group member can result in either a loss of communication, or degraded communication between the member and its area controller. The decentralized nature of Mykil allows operation in a disconnected environment. As long as a member can contact its area controller, it can continue to multicast data and receive data multicast by another member within the same partition of the network. However, if a member loses contact with its area controller, it can neither receive, nor send any multicast data. In a nutshell, fault-tolerance support in Mykil consists of three parts:

1. When a group member detects that it can no longer communicate with its area controller, it attempts to join another area by contacting that area's area controller.
2. When an area controller detects that it can no longer communicate with one of its area member, it terminates the membership of that member from its area.
3. Finally, when an area controller detects that it can no longer communicate with the area controller of its parent area, it attempts to change its parent area by contacting that area's area controller.

To simplify the authentication process and ensure a seamless service for a mobile client, Mykil employs a mechanism similar to Kerberos. An area controller provides a user a *membership ticket* when the user first joins the group. This ticket is valid for a fixed time period, and encrypted using a secret key that is known only to area controllers. When a user attempts to join another area, it presents its membership ticket to the area controller of the new area. This area controller verifies this membership ticket before granting membership to the rejoining user.

We have simulated Mykil in ns2 network simulator. Measurements show that Mykil scales significantly better than LKH or Iolus under different computing environment. We have also implemented Mykil on a network of Unix/Linux workstations. A performance measurement over a wide-area network shows that Mykil provides high scalability, efficiency, and fault-tolerance properties.