

Dynamics of Key Management in Secure Satellite Multicast

Michael P. Howarth, Sunil Iyengar, Zhili Sun, *Member, IEEE*, and Haitham Cruickshank, *Member, IEEE*

Abstract—Security is an important concern in today's information age and particularly so in satellite systems, where eavesdropping can be easily performed. This paper addresses efficient key management for encrypted multicast traffic transmitted via satellite.

We consider the topic of encrypting traffic in large multicast groups, where the group size and dynamics have a significant impact on the network load. We consider life cycle key management costs of a multicast connection, and show for a logical key hierarchy (LKH) how member preregistration and periodic admission reduces the initialization cost, and how the optimum outdegree of a hierarchical tree varies with the expected member volatility and rekey factor. This improves network utilization, but encryption at the network layer can pose problems on satellite links. We, therefore, propose and analyze an interworking solution between multilayer Internet protocol security (IPSEC) and LKH that also reduces key management traffic while enabling interworking with performance enhancing modules used on satellite links.

Index Terms—Logical key hierarchy (LKH), multicast, multilayer Internet protocol security (IPSEC), performance-enhancing modules (PEMs), performance-enhancing proxies (PEPs).

I. INTRODUCTION

SATELLITE-BASED broadband Internet protocol (IP) networks have the potential to deliver multicast services cost effectively. However, satellites present some significant security challenges.

- Eavesdropping and active intrusion are much easier than in terrestrial fixed or mobile networks because of the broadcast nature of satellites.
- Satellite systems are resource-constrained, particularly in the areas of limited transmission power (and, thus, channel capacity), and limited processing and switching capability for satellites with on-board processing.
- Satellite channels experience high bit-error rates, which can result in packet loss and the loss of security synchronization.

Security systems for satellite data, thus, have to be optimized to take account of these limitations, in particular the need for confidentiality and the requirement to use satellite resources efficiently. Geostationary satellites also suffer from a long propagation delay, and security systems must, therefore, add only minimal delays to traffic.

Manuscript received December 15, 2002; revised July 1, 2003 and September 20, 2003. This work was supported in part by the EU Information Society Technologies Program under the GEOCAST Project IST-1999-11754.

The authors are with the Centre for Communication Systems Research, University of Surrey, Guildford, Surrey GU2 7XH, U.K. (e-mail: m.howarth@surrey.ac.uk; s.iyengar@surrey.ac.uk; z.sun@surrey.ac.uk; h.cruickshank@surrey.ac.uk)

Digital Object Identifier 10.1109/JSAC.2003.819978

In this paper, we consider scalability of multicast security services, focusing on issues surrounding confidentiality on satellite links, while minimizing satellite resource usage. A particular issue in securing multicast connections is that of key management; this needs to scale effectively, particularly given the large number of multicast recipients (potentially of the order of millions) that are expected in satellite networks. A number of scalable approaches have been proposed, and one in particular, logical key hierarchy (LKH) is analyzed in detail in this paper.

Performance-enhancing modules (PEMs) are used to enhance performance on satellite links. One type of PEM is a performance-enhancing proxy (PEP) [6], which can be used to improve the performance of protocols on network paths such as satellites or wireless links, where native performance suffers due to the characteristics of the link or subnetwork. PEMs are often considered in the context of unicast transmission control protocol (TCP) traffic streams but are also applicable to multicast traffic. For example, an audio multicast stream may be transmitted using real-time transport protocol (RTP) over user datagram protocol (UDP). Since the data content of each packet is small, the transmission efficiency can be increased by using PEMs to compress the headers on the satellite link. Another example is implementing forward error correction (FEC) mechanisms in the PEMs to reduce the satellite link's effective error rate for a multicast channel. PEMs generally read and change the content of the transport layer header and, thus, they can usually only be used when the transport header is not encrypted. Thus, network layer security mechanisms, such as Internet protocol security (IPSEC), that encrypt the entire transport layer packet are incompatible with the use of these PEMs in satellite-based communications: IPSEC hides all details of higher layer protocols and makes it impossible for any intermediate node to process this information. Consequently, any service that requires knowledge of the transport layer header content anywhere other than in the end host cannot function if IP packets are encrypted.

A. Research Contributions

We consider key management, and in particular the life cycle key distribution costs. We take account not only of the cost of rekeying when users join or leave an established group, but we also consider the costs of building a tree during initialization of a multicast group. We show for LKH how user preregistration and periodic admission reduces this initialization cost, and how the optimum outdegree of a hierarchical tree varies with the expected user volatility and rekey factor. Other authors have shown [22] that when a rekey occurs after each join or depart, a lower

bound on the worst-case key update costs is a logarithmic variation with group membership N . Our work, on the other hand, whilst conforming to this logarithmic function, shows how the optimum value of the tree outdegree k varies with the expected rekey frequency, taking into account both the cost of initially building the tree and the cost of rekeying over the lifetime of the multicast group. In [19], the optimum number of keys assigned to each group member is related to the probability of deletion of the individual member. Here, we assume each member is equally likely to join or leave the group, giving a tree with equal numbers of keys for each member.

As described above, PEMs can be incompatible with end-to-end security systems and pose a particular problem for satellite systems. Multilayer IPSEC has been proposed as an approach that allows authorized network devices to access packet headers, while retaining end-to-end privacy of the data carried in the packets. In this paper, we propose and analyze an interworking solution between multilayer IPSEC (ML-IPSEC) and LKH and show how the key management traffic on satellite links can be further reduced using this approach. Finally, the impact of ML-IPSEC and LKH on the life cycle costs is considered.

B. Outline of Paper

This paper is organized as follows. Section II reviews related work in end-to-end satellite security and in key management for multicast group security. In Section III, we briefly review the operation of LKH in key management and present an analysis of life cycle key distribution costs. In Section IV, we consider multilayer IPSEC and how it supports multicast security in satellite systems. In Section V, we demonstrate how it can be combined with LKH to provide efficient and secure key management.

II. RELATED WORK: MULTICAST KEY MANAGEMENT

The process of securing and performing key management for unicast connections is well understood [8], [13], [18], but multicast security is more complex. In principle, a multicast connection could be regarded from the security perspective as a set of unicast connections, but this approach does not scale well for large groups, especially at the scales expected in satellite systems. Protocols that manage the process of distributing keys in a multicast environment are under development [2], [5], [9].

The principal actors in multicast key management are the group controller (GC) and group members (GMs). The former is responsible for creating and distributing keys and rekeying (to maintain security) as appropriate; the latter are entities with access to the group keys. The GC need not be colocated with the multicast data source. Each GM has an initial one-to-one *secure association* with the GC (using techniques such as Diffie–Hellman to create a shared secret known only to the two parties; or a preshared secret; or secret exchange using a public key system [20]). These secure associations are then used to create and share information about a *group secure association* between the GC and all GMs. The ultimate aim of the group secure association is to ensure that a single key, usually called the group traffic encryption key (GTEK), is known to the GC

and all GMs and to no entity outside the group: This key can then be used to encrypt the data multicast within the group.

The multicast group may need to be rekeyed for any of a number of reasons.

- 1) The group key is usually updated regularly (typically every few seconds or minutes) to reduce the probability of successful cryptanalysis of the encrypted traffic.
- 2) The group key may also need to be changed on demand if it is determined that the key has been compromised.
- 3) Rekeying may be required when a new member joins the multicast group. This ensures that the member cannot decrypt encoded traffic sent prior to their joining (backward secrecy).
- 4) Rekeying may be required when an existing member departs from the multicast group. This ensures that the member cannot decrypt encoded traffic sent after they leave (forward secrecy).

For large multicast groups that have frequent membership changes, the cost of rekeying can be significant, since satellite resources are expensive. Scalable rekeying is, therefore, an important problem that needs to be considered in order to support secure communications for large dynamic groups. We now consider rekey techniques for each of the four functions listed above.

Several techniques exist for rekeying 1) and 3): Two options are for the new group key to be encrypted with either a) the old group key or b) a separate “control” key negotiated during session establishment. For 2) and 4), a different rekeying approach is required since the old key is known by at least one user who is no longer to be a recipient of the multicast transmission. We now consider options for this rekeying.

A number of multicast key management approaches have been developed with the objective of improving the scalability of group secure associations by ensuring that parameters grow more slowly than the group size N . Parameters considered include GC encryption effort and memory requirements, network traffic, and GMs’ decryption effort and memory requirements. Key management techniques include a flat system, Iolus [14], LKH [23], [24], and Kronos [21].

In the simplest approach, a flat system, the GC shares a unique key with each individual GM. The GTEK can then be sent to the members by encrypting it N times with each of the N unique keys. Thus, both the GC key encryption load and the rekey traffic increase linearly with N . It should be noted that for this flat system the broadcast nature of a satellite does not provide any benefit compared with a terrestrial network, since the encrypted messages sent to the N GMs are all different.

In Iolus [14], a multicast group is partitioned into several subgroups. The GC manages a tree of group subcontrollers, each of which manages a subset of the group membership. The advantage of this mechanism is that the rekey effort is shared between the subcontrollers, but the drawbacks of this approach are the large number of subcontrollers required in large groups, the need to trust the subcontrollers, and the delay caused by the need to rekey traffic as it passes through each subcontroller.

LKH, described in more detail below, uses a set of keys arranged in a tree structure to reduce the cost of rekeying. For a fully populated tree of outdegree k and depth d , the number of

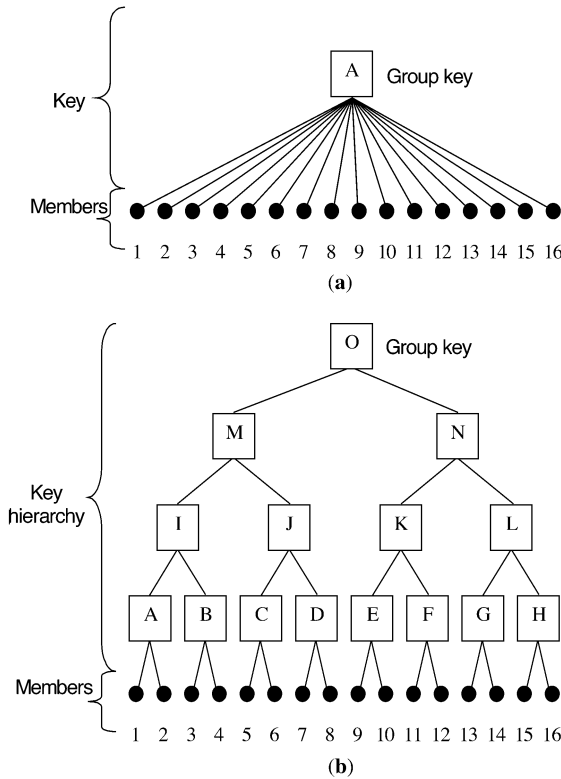


Fig. 1. Key hierarchies. (a) N pairwise keys. (b) Hierarchical tree.

rekeys transmitted on a member compromise is $k \log_k N - 1$; this compares favorably with the cost of $N = k^d$ for a flat system. The system is also robust against collusion, in that no set of users together can read any message unless one of them could have read it individually. Improvements to LKH for the specific case of binary trees ($k = 2$) have also been proposed in one-way function trees [4], [15], and by [7]: Both these approaches reduce the number of rekeys required in the event of compromise of a user from $2 \log_2 N - 1$ to $\log_2 N$.

Kronos is a further approach to reducing rekey traffic in large dynamic multicast groups. This approach recognizes that if two users depart and cause two rekey events to occur, some of the keys that change will be common to the two rekey events. Rekey traffic can, therefore, be saved by bundling changes together and rekeying, perhaps every few seconds.

III. LKH LIFE CYCLE COSTS

A. Description

Before analyzing LKH in detail, we initially briefly review the simple flat key management system. Consider N pairwise keys each shared between the GC and one of the N GMs [Fig. 1(a)]: This represents the flat system described in Section II. The pairwise keys are represented by the circles, and the group key is represented by the box labeled “A.” If the group key is changed, the new group key has to be encrypted with each member’s unique pairwise key and then sent to that member; each of these encrypted keys is represented by one of the lines drawn in Fig. 1(a). Thus, for N members, a total of N encrypted keys are generated and transmitted across the satellite network. In this case, the key management network

traffic volume is independent of whether the keys are unicast, multicast, or broadcast (although for good security the keys should be transmitted to as few entities as feasible).

We contrast this with LKH, where a tree of keys is used. The tree is logical and does not have any mapping onto a physical network layout: The only physical entities remain the GC and the GMs, and LKH is highly suited to satellite systems. In Fig. 1(b), the keys are labeled A through O, each circle again represents a member’s pairwise key, and the lines each represent encrypted keys sent across the network, as we shall now see. Each member holds the keys on the tree path from the member’s pairwise key (at a tree leaf) back to the root. Suppose now that member 11 needs to be deleted from the multicast group. Then, all of the keys held by member 11 (keys F, K, N, O) must be changed and distributed to the members who need them, without permitting member 11 or other unauthorized entities to obtain them. To do this, we must replace the keys held by member 11, proceeding from the bottom up.

The GC chooses a new key for the lowest node (not the leaf, for which a unicast secure association exists between the GC and the GM) and then transmits it encrypted with the appropriate child keys. Thus, for this example, the first key replaced is key F and this new key will be sent encrypted with member 12’s unique pairwise key. The second key replaced is key K, which is sent encrypted with the newly replaced key F (for member 12) and also sent encrypted with key E (for members 9 and 10). Key N is then sent encrypted with the newly replaced key K (for members 9, 10, and 12) and also encrypted with key L (shared by members 13–16). Finally, key O is replaced, and this new key is sent encrypted with the newly replaced key N (for members 9, 10, and 12–16) and also separately is encrypted with key M (shared by members 1–8). Since we are proceeding from the bottom up, each of the replacement keys will have been replaced before it is used to encrypt another key. The encrypted keys need to be multicast or broadcast (noting that as above, for good security the keys should only be transmitted within the group), and those keys that are encrypted with pairwise keys may be unicast.

The seven keys sent represent a significant saving on the 16 keys that would need to be transmitted using the flat key system of Fig. 1(a). In general, the number of transmissions required is the sum of the degrees of the replaced nodes. In a fully populated k -ary tree in which a GM sits at depth d , this is a total of $kd - 1 = k \log_k N - 1$ transmissions.

The GTEK, used to encrypt data traffic, may, depending on the group security policy, either be key O [Fig. 1(b)], or it may be separately encrypted using key O and transmitted to all GMs. In the analysis that follows, we adopt the former approach and choose not to include the GTEK in the rekey count.

B. Life Cycle Costs

There is an implicit assumption in the LKH literature that user join/leave rekeying is the principal cost of key management, with tree initialization costs being negligible. However, applications exist (such as file transfer), where the user population has a low volatility and, therefore, the tree initialization costs become significant. Also, and distinctly, there may be some multicast groups (e.g., pay-per-view) where subscribers join the group at

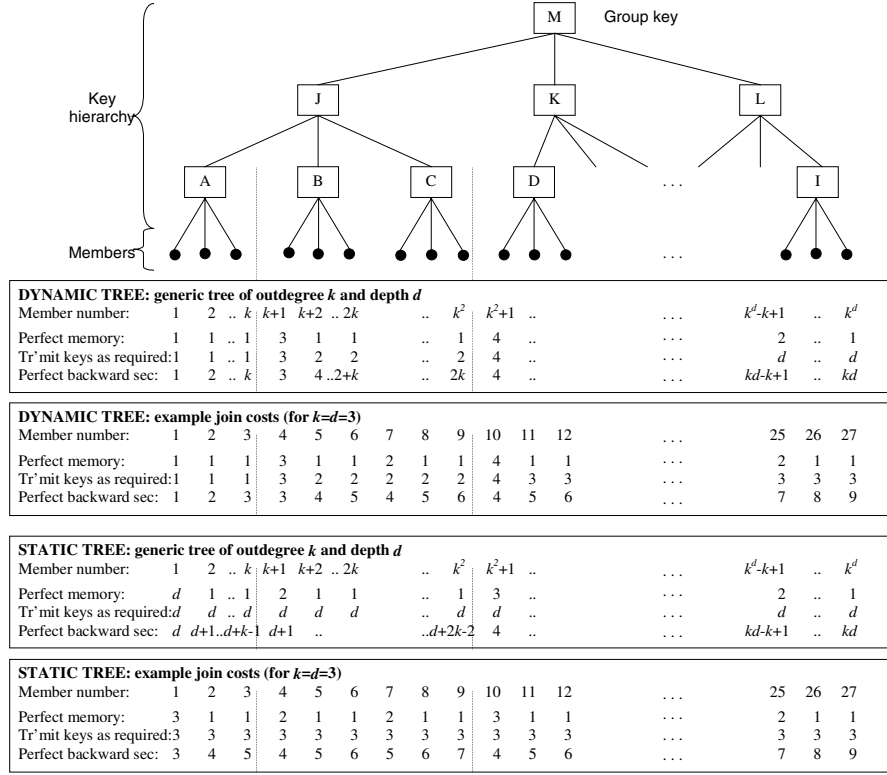


Fig. 2. Tree initialization approaches.

around the same time, with a high demand on the establishment of the group secure association.

For our analysis, we divide the life cycle of a secure group connection into the following phases.

- Initialization: Members are authenticated to the GC and are issued with keys, including the group key used to encrypt data.
- Data transfer: This comprises two rekey activities:
 - The GTEK is updated regularly to reduce the probability of successful cryptanalysis of ciphertext;
 - Rekeying occurs when members join or depart to ensure perfect backward or forward secrecy, respectively, or when a key or member is compromised.
- Termination: Once the data transfer is complete the secure connection ends. We assume that the keys are simply discarded, since they are of no further use.

We assume in this analysis that the entire LKH tree is populated during the initialization phase, that once the initialization phase is complete the LKH tree remains balanced [16], and that it remains nearly full as members join and depart.

C. Analysis

As each GM joins a group the member is authenticated and a pairwise key for secure communication with the GC is established using a mechanism that in line with current approaches [9] is out of scope of this work. Our metric is the number of encrypted keys transmitted by the source: Ignoring protocol overheads, this is proportional to the key network traffic. For secure multicast over satellite, where resources are limited, it is particularly important to minimize the key management traffic costs.

We now consider the initialization cost. Three approaches are considered, in increasing order of cost, with generic cases and numerical examples being shown in Fig. 2. For each of the three approaches we may either consider a *static* tree, in which the tree depth d is assumed fixed in advance, or we may consider a *dynamic* tree, in which the tree depth grows as the number of members increases, the depth being one when there are not more than k members, two when the number of members lies in the range $k+1$ to k^2 inclusive, and so forth. The initialization costs for each of the three approaches are as follows.

Case 1) Assume each receiver has perfect memory: Once a given key has been transmitted, any receiver that is given the encoding key (even at a later time) can decrypt the encoded key. Although this assumption is not expected to be valid in practice, it provides a useful comparison in this analysis since it has the lowest initialization cost. As will be seen later, it also corresponds to a useful category of tree building. It is shown in Appendix A that for both static and dynamic trees, the total number of keys transmitted to admit N users to the group is

$$B_1 = \frac{k}{k-1}(N-1) \sim O(N). \quad (1)$$

Case 2) Transmit all the keys needed by a receiver at the time the receiver joins the group. Users do not require memory of previous transmissions and may even be switched off prior to joining the group. Keys are, therefore, in general, transmitted multiple times, when different users require them. In this case, the total number of keys transmitted to admit

N members to the group is $O(N \log_k N)$ with the exact expressions for static and dynamic trees, respectively, being (Appendix A)

$$B_{2\text{static}} = N \log_k N \quad (2)$$

$$B_{2\text{dynamic}} = (N+1) \log_k N - \frac{N}{k-1} + \frac{1}{k-1}. \quad (3)$$

Case 3) Rekey on each join: This ensures perfect backward secrecy is maintained as the tree is initialized. To ensure perfect backward secrecy as each member joins the group, not only must the group key be changed, but also, the keys that enable the joining receiver to decrypt ancestor keys must also be changed: If this is not done, then a newly joined receiver with memory could obtain the previous group key and, therefore, decrypt transmissions that occurred prior to the user joining the group. The total number of keys transmitted to admit N users is also $O(N \log_k N)$, with the expressions for static and dynamic trees being (Appendix A)

$$B_{3\text{static}} = \left(\frac{k+1}{2}\right) N \log_k N \quad (4)$$

$$B_{3\text{dynamic}} = \left(\frac{k+1}{2}\right) N \log_k N - \frac{N}{k-1} + \frac{k}{k-1}. \quad (5)$$

D. Implications

The perfect memory approach has the lowest tree initialization cost. It also corresponds to a useful way of building a multicast tree: consider a multicast session that is advertised [perhaps using the session announcement protocol (SAP)] and for which potential members are invited (or required) to register in advance. Then, at the time the session is due to start the source knows the number N of members and builds the tree in the following order: All lowest level keys ($A, \dots, I$, Fig. 2) are sent encrypted with each recipient's pairwise key, then the second level keys ($J, \dots, L$, Fig. 2), and so on up the tree. Each receiver can decrypt each key as it is received and the tree initialization cost (1) is $O(N)$.

The second approach corresponds to members arriving intermittently. In this case, the source cannot assume that a newly joining receiver has any key information, and all the keys required by each receiver have to be transmitted with the initialization cost being $O(N \log_k N)$.

The cost [(2), (3)] can, however, be reduced if joining receivers are grouped together (perhaps every few seconds) so that keys can be transmitted together. For example, if (Fig. 2) members 25 and 26 join separately, six keys are sent. However, if they join simultaneously only four keys are sent. This is similar to periodic rekeying [21], and since here we are predominantly initializing new users, the required set of keys will be highly correlated, and so significant savings can be realized. The savings for periodic rekeying have been quantified by [25].

E. Volatility

We now consider the life cycle cost of the group connection, which we define in this section as the sum of the cost of initializing the LKH tree and the cost of rekeying during the data

transfer phase when GMs join or depart. We further define the volatility α as the mean number of rekeys per GM. Thus, a value of $\alpha = 1$ means that on average there is one rekey operation per GM during the lifetime of the secure transmission. The cost of each such rekey is $R = k \log_k N - 1$ when a GM departs or $R = k \log_k N$ when a GM joins: these functions have a minimum value at $k = 3$ (noting that k can only take integer values). However, when we include both the initialization and rekey costs, the optimum value of k that gives a minimum life cycle cost is different. For each initialization approach, the life cycle cost for static trees is

$$C_1 = B_1 + RN\alpha = \frac{k}{k-1}(N-1) + \alpha N(k \log_k N - 1) \quad (6)$$

$$C_{2\text{static}} = B_{2\text{static}} + RN\alpha = N \log_k N + \alpha N(k \log_k N - 1). \quad (7)$$

To ensure perfect backward secrecy, we include the rekey cost on a user join $k \log_k N$. Assuming equal number of joins and departs

$$\begin{aligned} C_{3\text{static}} &= B_{3\text{static}} + R'N\alpha \\ &= \left(\frac{k+1}{2}\right) N \log_k N + \alpha N(k \log_k N - 0.5). \end{aligned} \quad (8)$$

We differentiate these expressions (Appendix A) and use Newton-Raphson iteration to find the minimum cost as a function of k for a given α . Fig. 3 shows each cost as a function of α for dynamic trees and static trees. To illustrate the effect of optimizing k , curves are drawn both for a tree of fixed outdegree $k = 3$ and for the case when the tree outdegree is optimized for the given volatility. Fig. 4 shows what this optimum value of k is for each value of volatility α . Fig. 5 shows how the total cost varies as a function of k for two example values of α .

We observe the following.

- Preregistration has the lowest cost.
- The intermittent arrival cost can be reduced significantly at low volatility by optimizing k (the optimum values are $k = 37$ at $\alpha = 10^{-2}$ and $k = 8$ at $\alpha = 10^{-1}$ for $N \sim 10^3$ to 10^6 , Fig. 4).
- Backward secrecy has the highest cost.
- The life cycle costs are almost identical for static and dynamic trees (static trees are, therefore, used for Figs. 4 and 5).
- At low volatility, there are significant cost differences between the optimum value of k and the conventional value $k = 2$ or $k = 3$, for case 1 (perfect memory receivers) and case 2 (transmit keys at the time the receiver joins the group) (Fig. 3).
- For frequent rekeying ($\alpha \gg 1$), the curves converge and the cost is independent of the initialization approach.

An example, low volatility population is a corporate video-conference transmitted to employees, who might be expected not to leave the conference frequently. A significant example of zero volatility is a file transfer: It is not meaningful to leave and rejoin since this will result in data loss. In this case, the volatility $\alpha = 0$ and the optimum key hierarchy is flat. Indeed, at low volatility the Case 1 preregistration (= perfect memory receivers) and Case 2 intermittent

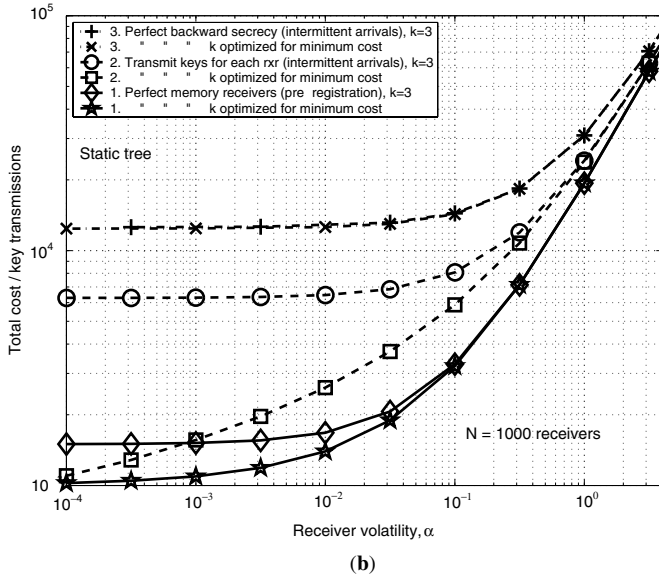
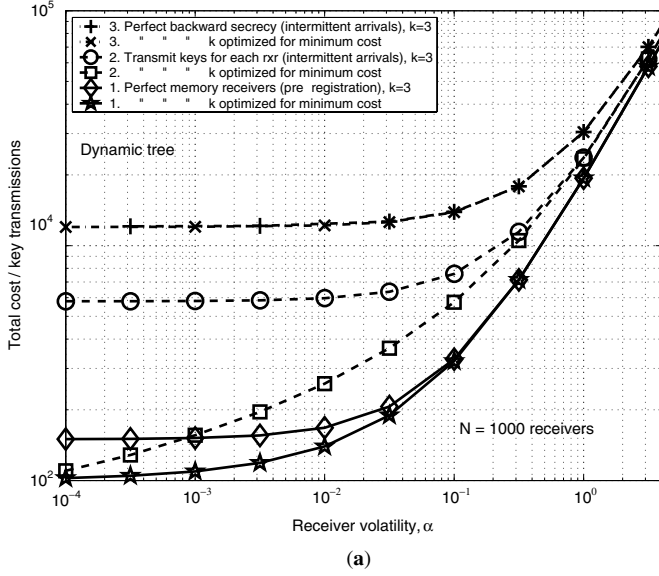


Fig. 3. Life cycle cost as a function of volatility α . (a) Dynamic tree. (b) Static tree.

arrival curves with k optimized converge to the same cost $C_1 = C_2 = N$ (Fig. 3).

F. Group Key Update

The life cycle cost can also be considered taking into account rekeying to prevent cryptanalysis. In this section, we therefore redefine the life cycle cost as the sum of the cost of initializing the LKH tree and the cost of regular GTEK updates during the data transfer phase. Let the lifetime of the group be T , and let the group update period be τ . Then, the total number of changes of the group key over the life of the group is T/τ . If each new group key is encrypted with each of its children and then multicast (e.g., $\{M\}_J, \{M\}_K, \{M\}_L$, Fig. 2), the total number of keys transmitted due to group key updates is kT/τ . We define a normalized rekey factor $\beta = T/N\tau$; thus, a group of 10^5 users

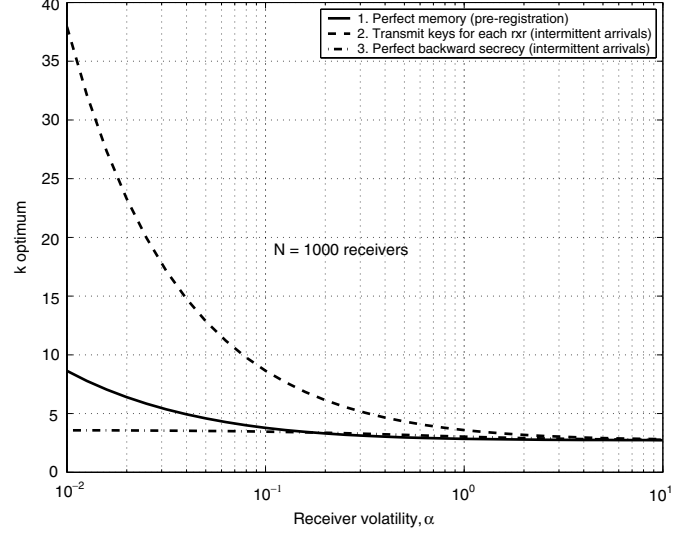


Fig. 4. Optimum value of tree outdegree k as function of volatility α .

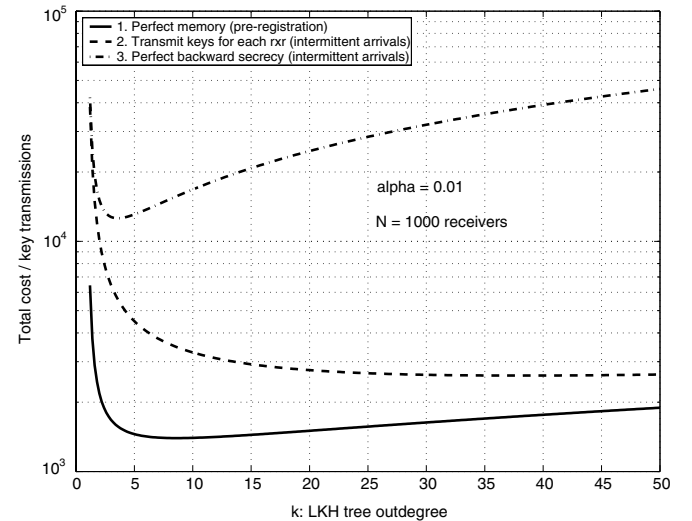
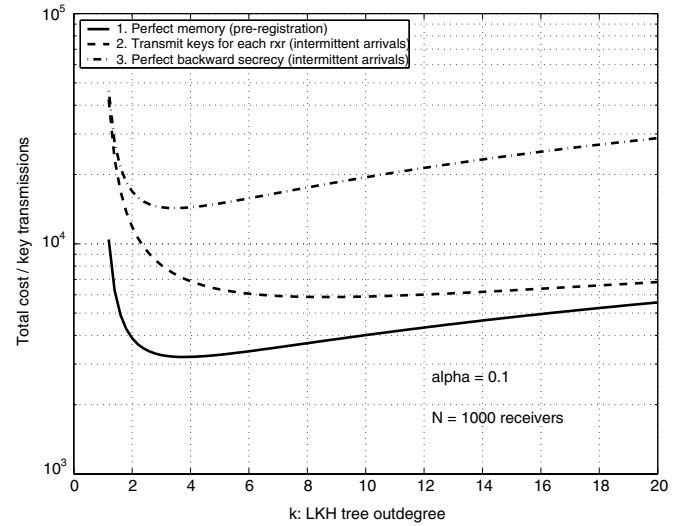


Fig. 5. Cost sensitivity for various values of volatility α .

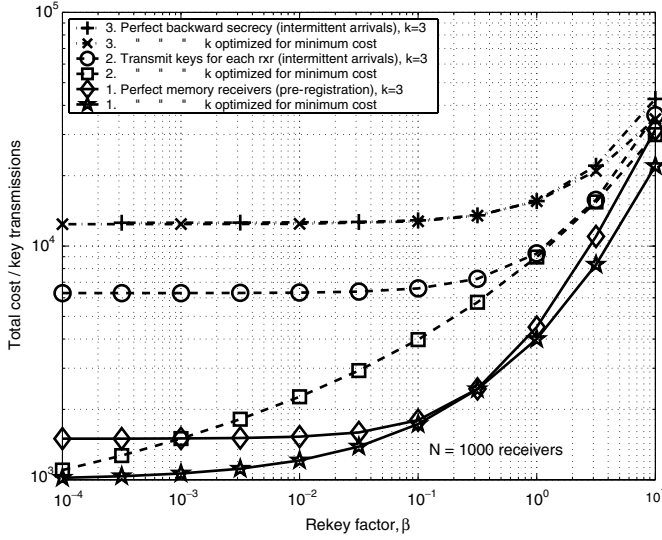


Fig. 6. Life cycle cost as a function of rekey factor β .

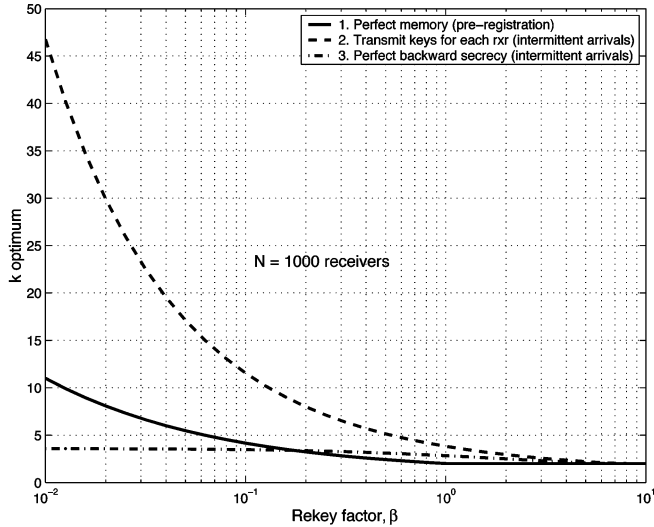


Fig. 7. Optimum value of tree outdegree k as function of rekey factor β .

with a lifetime of 3 h and group key update period of 10 s has a $\beta = 10^{-2}$. The life cycle costs are then, for example, for perfect memory/preregistration

$$C'_1 = B_1 + kN\beta = \frac{k}{k-1}(N-1) + kN\beta \quad (9)$$

which has a minimum with respect to k when $k = 1 + (1/\sqrt{\beta})$. The life cycle costs are shown as a function of β for static trees in Fig. 6, and the optimum value of k as a function of β is shown in Fig. 7. We note the following.

- At low β , when the number of rekeys per GM is low, the optimum value of k is $\gg 3$.
- At high β , rekeying to prevent cryptanalysis predominates over tree build costs, and so the optimum value of k is $k = 2$ (noting that it is not meaningful for k to be less than 2).

IV. ML-IPSEC

The analysis of Section III shows how the network rekey traffic can be optimized, but when we consider multicast on satellite links, the use of traffic encryption at the network layer can be incompatible with PEMs. We, therefore, now proceed to consider a mechanism that permits network layer traffic encryption with satellite PEMs. In this section, we describe ML-IPSEC and show how it supports multicast security over satellites while providing enhanced utilization of the satellite resources. In Section V, we go on to illustrate a solution that provides interworking between ML-IPSEC and LKH, while further reducing the key management traffic.

A. IPSEC

The IP security architecture IPSEC provides standardized Internet security at the IP layer [10] and supports interoperable cryptographically based security services (i.e., confidentiality, authentication, integrity, and nonrepudiation). It consists of an authentication protocol, authentication header (AH) [11]; a confidentiality protocol encapsulated security payload (ESP) [12]; it also includes an Internet security association establishment and key management protocol (ISAKMP) [13]. These security protocols are designed for both IP version 4 (IPv4) and IP version 6 (IPv6) environments.

IPSEC supports two modes: transport and tunnel. In transport mode, the IP header is transmitted in clear, and the IP data is encrypted. Network devices can, therefore, process the IP datagram. In tunnel mode, the entire datagram (header and payload) are encrypted and a new IP header is prefixed to the datagram. In either mode, the transport layer header is encrypted. This, therefore, includes transport headers that contain information needed in the case of satellite gateways to perform performance enhancing or other intelligent routing functions.

The transport layer may be either unicast (TCP or UDP) or multicast or reliable multicast. Some basic rules for TCP optimization techniques used in satellite communications and the implications they might have for IPSEC have been pointed out in [17]. However, if the optimization techniques involve intermediate routers or gateways and these require read or write access to the transport layer header or encapsulated data, IPSEC cannot be used without some kind of adaptation.

B. ML-IPSEC

Work on ML-IPSEC has been carried out for example by Hughes Network Systems [26]. It has also been considered for mobile networks [3], and optimization of multicast over satellites is still a research issue. ML-IPSEC defines a security relationship that involves not only the sender and the receiver of a security service but selected intermediate nodes along the traffic stream as well. The IP datagram is divided into several zones and different protection schemes are applied to each zone (illustrated in Fig. 8 for IPSEC transport mode). Individual security relationships can be used to cover each zone of the IP datagram and then build a new type of secure association (SA) called a composite SA (CSA). Thus, in Fig. 8, the transport data is encrypted using key K2, while the transport header is encrypted using key K1.

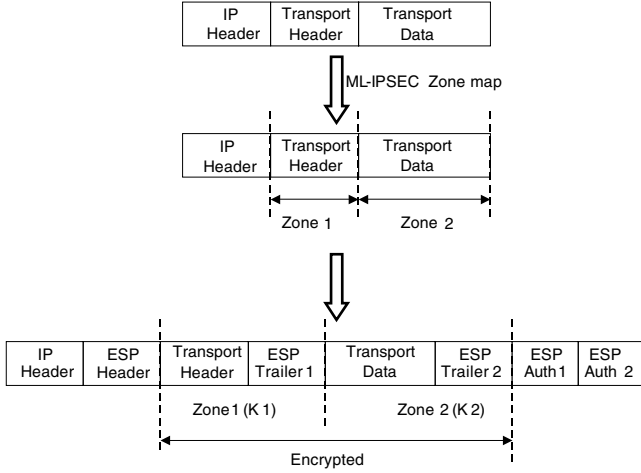


Fig. 8. ML-IPSEC datagram structure (transport mode).

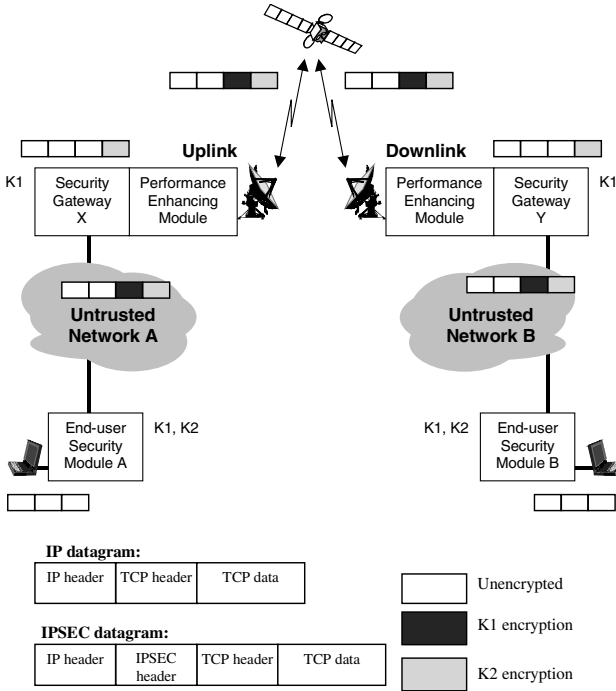


Fig. 9. ML-IPSEC in a satellite context.

C. ML-IPSEC for Satellites

Fig. 9 illustrates the use of ML-IPSEC in a satellite environment [1]. For simplicity, the analysis is initially illustrated here in terms of a secure unicast connection. For multicast, the analysis is identical, although there are potentially many downlink gateways. A user in network A wishes to establish a secure link with a node in network B. End-user security module A establishes secure associations with end-user security module B and also with the satellite security gateways X and Y. When user A sends an IP datagram, it is encrypted using key K1 to encrypt the transport header and key K2 to encrypt the transport content. On receipt by the uplink terminal the transport header only is decrypted by security gateway X, any performance enhancing functions such as header compression or spoofing are performed, and the transport header is re-encrypted. The datagram is then transmitted over the satellite link, where security

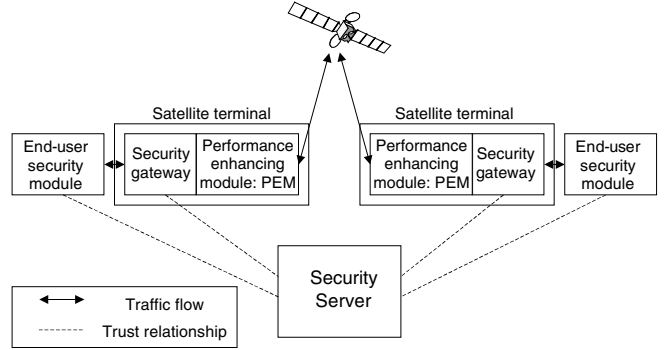


Fig. 10. ML-IPSEC trust relationships.

is assured by the encryption. At the satellite downlink (security gateway Y), the transport header can be decrypted using K1, and any performance enhancing functions can be performed. The header is re-encrypted by security gateway Y using K1 and forwarded. The datagram is fully secured in transit over the untrusted network B. At end user B, the datagram is decrypted using both K1 and K2.

In summary, it can be seen that the security entities at the ends of a connection (i.e., at the source A and at each destination B of a multicast transmission) need both group keys K1 and K2. However, intermediate security gateways that are responsible for performance-enhancing functions only need access to group key K1 to enable them to read and if necessary change the transport header.

This approach raises issues of trust, since there are two different entities which have access to common information. The security gateway and PEM are both part of the satellite terminal and, thus, under the control of the satellite terminal operator (Fig. 10); the end-user security module, on the other hand, is part of the end user terminal or application and is controlled by the end user. The security gateway needs access to key K1 and the end user(s) need access to keys K1 and K2. In order to achieve this, both the satellite terminal operator and the end user need to trust a secure third party (the security server shown in Fig. 10). The trusted third party is responsible for generating and distributing the keys and this forms the basis for the end-to-end security between users. The model also implies a limited trust between the user and the satellite terminal operator, where the latter is trusted to have access to the transport headers.

V. ML-IPSEC AND LKH INTERWORKING

A. Analysis

We now present an extension of LKH as discussed in Section III that provides an efficient and scalable key management system for multicast ML-IPSEC. The two group keys K1 and K2 could be managed using two separate LKH trees, but a saving can be made by integrating them into a single hierarchy as follows. Fig. 11 shows the proposed key hierarchy for a set of users U1 to U9 and a set of intermediate gateways G1 to G4. All users and intermediate gateways are members of the multicast group. In this illustration, the nine users are in a subtree of outdegree $k_U = 3$, and the four gateways are in a subtree of outdegree $k_G = 2$. The root (key H) has two children,

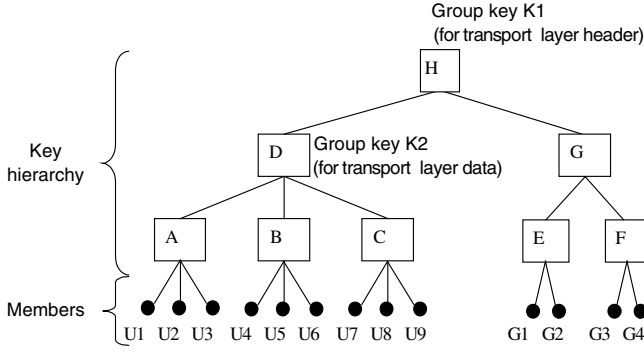


Fig. 11. Integrated LKH tree for ML-IPSEC.

irrespective of the values of k_U and k_G . As in Section III, Keys K1 and K2 could either be the LKH keys H and D shown in Fig. 11, or they could be separate keys encrypted using H and D, respectively, and transmitted to members. The group key K1 used to encrypt the transport layer header is at the root of the tree and the group key K2 used to encrypt the data is one of the root's two child keys. Recalling that in LKH each member only knows the keys that lie on the path from the member's leaf node to the root, it can be seen that users have access to both K1 and K2, while gateways only have access to key K1. In the event of a gateway compromise, as part of the normal LKH rekeying we transmit $\{K1\}_{K2}$, and so, users can still decrypt the transport header. For the users located behind the compromised gateway, it is assumed that a protection path exists via one of the other gateways so that they can still receive the multicast traffic.

If there are N_U users and N_G intermediate gateways, then the cost of rekeying for this single integrated tree is as follows. For a user depart, assuming a fully populated tree, the rekey cost in keys is

$$R_{U\text{-integrated}} = k_U \log_{k_U} N_U + 1. \quad (10)$$

For a gateway depart, the rekey cost is

$$R_{G\text{-integrated}} = k_G \log_{k_G} N_G + 1. \quad (11)$$

This compares with rekey costs for two separate trees (Fig. 12) as follows. The combined LKH tree for key K1 has $N_U + N_G$ members; let its outdegree be k_{comb} . The LKH tree for K2 has N_U members, and we assume its outdegree to remain at k_U . Then, for a user depart, the rekey cost to rekey both keys K1 and K2 is

$$R_{U\text{-separate}} = \{k_{\text{comb}} \log_{k_{\text{comb}}} (N_U + N_G) - 1\} + \{k_U \log_{k_U} N_U - 1\} \quad (12)$$

and for a gateway depart, the rekey cost is (to rekey K1 only)

$$R_{G\text{-separate}} = k_{\text{comb}} \log_{k_{\text{comb}}} (N_U + N_G) - 1. \quad (13)$$

B. Implications

The rekey costs on a user departure and a gateway departure are shown, respectively, in Figs. 13 and 14. These show that for a user rekey, the rekey cost is almost halved for values of $N_U \geq N_G$ and is reduced still further in the unlikely event that the number of gateways exceeds the number of users. For a gateway departure, the integrated LKH saving compared with

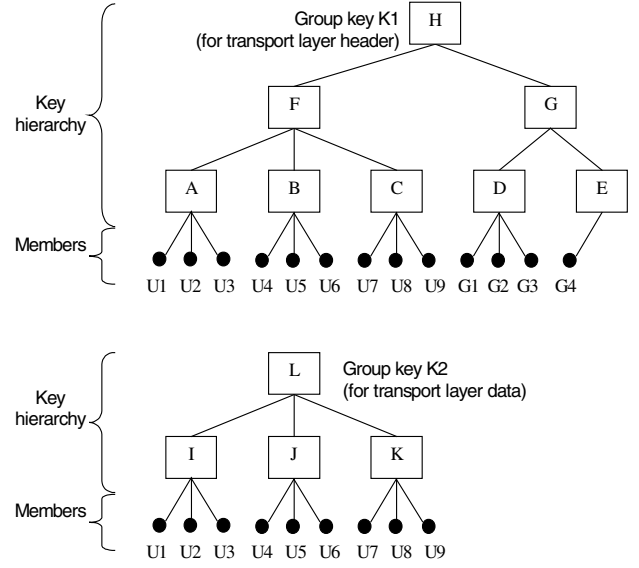


Fig. 12. Separate LKH trees for transport header and data.

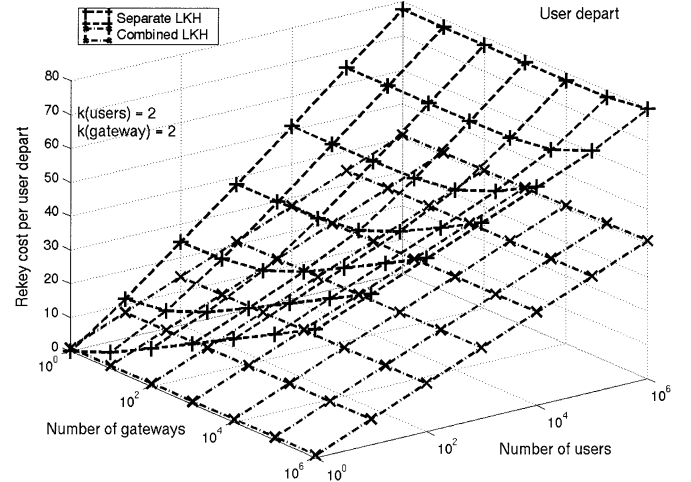


Fig. 13. ML-IPSEC and LKH interworking: Rekey costs on user depart.

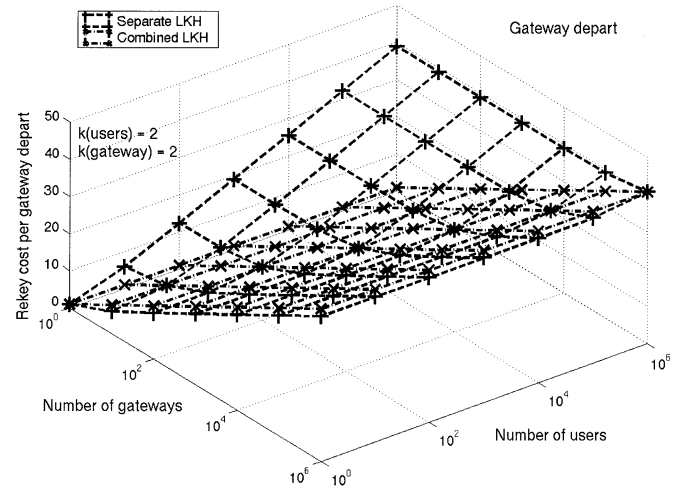


Fig. 14. ML-IPSEC and LKH interworking: Rekey costs on gateway depart.

the separate LKH rekey cost is particularly high for $N_U \gg N_G$, that is, when there are a relatively small number of gateways.

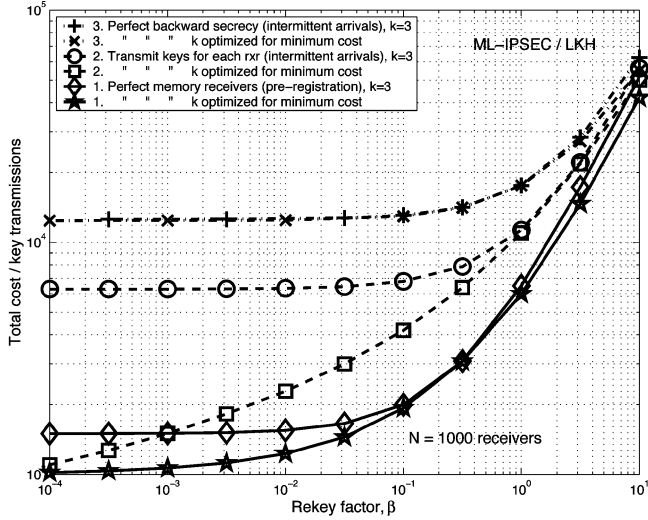


Fig. 15. ML-IPSEC and LKH: Life cycle cost as function of rekey factor β .

The gateways would normally be expected to be less volatile than the users. In this case, the optimum outdegree of the user subtree may be selected using the analysis described in Section III using the outdegree of the user subtree, k_U .

Rekeying to update the group key can either change just the transport data key K_2 , or it can involve changing both K_1 and K_2 . In the former case, the analysis of Section III and the life cycle rekey costs given in Fig. 6 are applicable for a tree of N_U users and outdegree k_U . In the latter case, the number of keys transmitted for each rekey is $2 + k_U$, and the life cycle cost is illustrated in Fig. 15.

VI. CONCLUSION

Satellite-based network services present significant security issues, in particular the requirement for privacy and the need to use satellite resources efficiently. This paper makes two distinct contributions to this area for multicast services.

In the first contribution, we have considered life cycle key costs in multicast groups that use LKH for key management. We have shown how preregistration can reduce the hierarchical tree initialization cost. When preregistration is not feasible, periodic admission can reduce this cost. For applications with low volatility α , where the fraction of users joining and leaving the group over the connection lifetime is low, there is an optimum tree outdegree that varies with the volatility and gives the minimum life cycle key cost. Similarly, the optimum tree outdegree varies with the rekey factor β that reflects the number of group key updates. This analysis, therefore, minimizes the key management traffic demand on the satellite; it is also of general application to terrestrial networks. We have motivated our analysis with low-volatility application examples such as file transfer and videoconferences.

In the second contribution, we have presented an interworking solution between ML-IPSEC and LKH that supports the implementation of PEMs for satellites. In the proposed approach, the end users are placed in one branch of the LKH tree, and the satellite terminals or gateways are located in another branch. The root key in the LKH tree can be used

for securing the transport header and a branch key acts as GTEK and secures the data content for the end users. The proposed scheme is scalable, in that the rekey effort varies with $\log N$, and efficient, in that for user departures the number of rekeys required is in the region of half that of two separate tree hierarchies.

APPENDIX A LKH TREE BUILDING COSTS

A. Perfect Memory Receivers

For perfect memory receivers, each edge on the tree is transmitted once only. The number of keys is, therefore, independent of whether the tree has been built statically or dynamically, and for a fully populated tree, the total number of keys transmitted to admit $N = k^d$ users to the group is, therefore

$$B_1 = k + k^2 + k^3 + \dots + k^d = \frac{k}{k-1}(N-1).$$

If the life cycle cost is given by $C_1 = B_1 + RN\alpha$, where R is the rekey cost $R = k \log_k N - 1$, and α is the volatility, then the minimum cost occurs when $(\partial C_1 / \partial k) = 0$; assuming $N \gg 1$, this occurs at

$$f_1 = 0 = \frac{-1}{(k-1)^2} + \alpha \left(\frac{\ln k - 1}{\ln^2 k} \right) \ln N.$$

B. Transmit Keys Required for Each Receiver

For a static tree, each joining receiver requires d keys to be transmitted, and the total number of keys transmitted is simply

$$B_{2\text{static}} = Nd = N \log_k N.$$

The life cycle cost $C_{2\text{static}} = B_{2\text{static}} + RN\alpha$ has a minimum when $(\partial C_{2\text{static}} / \partial k) = 0$, which occurs when

$$f_{2\text{static}} = 0 = \alpha \ln k - \frac{1}{k} - \alpha.$$

We next consider a dynamic tree. The first k users are assigned to a tree of depth 1, with group key A (Fig. 2), and so they join at a cost of 1 key each. User $k+1$, however, requires the tree depth to increase to 2 with group key J (Fig. 2) and joins at a cost of three keys: $\{J\}_A$, $\{J\}_B$, and $\{B\}_{k+1}$. When user $k+2$ joins, we assume it does not have $\{J\}_B$ (because it did not save this message when it was transmitted, or because it was switched off) and, therefore, $\{J\}_B$ is now transmitted a second time, giving a cost of two keys. Users $k+3$ through k^2 are also added at cost of two keys each. In general, the cost of adding users $k^i + 1$ through k^{i+1} is $(i+2) + (i+1)(k^{i+1} - k^i - 1)$, and so the total number of keys transmitted to admit N users is

$$B_2 = k + \sum_{i=1}^{d-1} [(i+2) + (i+1)(k^{i+1} - k^i - 1)]$$

which simplifies to

$$B_2 = (N+1) \log_k N - \frac{N}{k-1} + \frac{1}{k-1}.$$

Assuming $N \gg 1$, the minimum life cycle cost $(\partial C_{2\text{dynamic}} / \partial k) = 0$ occurs at

$$f_{2\text{dynamic}} = 0 = \frac{\ln N}{k \ln^2 k} (\alpha k \ln k - \alpha k - 1) + \frac{1}{(k-1)^2}.$$

C. Rekey on Each Join (Perfect Backward Secrecy)

In rekeying, we assume that each changed key is transmitted k times encrypted with each of its k child keys.

For a static tree the first k receivers join at a total cost (Fig. 2) of $x = kd + (1/2)(k-1)k$ keys, and the first k^2 receivers (1 through k^2) join at a total cost of $kx + k(1 + 2 + \dots + (k-1)) = k^2d + k^2(k-1)$. By induction, the total cost of tree initialization is

$$B_{3\text{static}} = k^2d + \frac{d}{2}k^2(k-1) = \frac{k+1}{2}N \log_k N.$$

The minimum life cycle cost $C_{3\text{static}} = B_{3\text{static}} + R'N\alpha$, where $R' = k \log_k N - 0.5$ occurs at $(\partial C_{3\text{static}}/\partial k) = 0$, i.e.,

$$f_{3\text{static}} = 0 = \left(\frac{1}{2} + \alpha\right) \ln k - \left(\frac{1}{2} + \frac{1}{2k} + \alpha\right).$$

The initialization cost for a dynamic tree is as follows. The first user joins at a cost of one key; when the second user joins, the group key A is changed and is transmitted to both users at a cost of two keys. This continues until user k joins at a cost of k keys. User $k+1$ requires a new group key J to be created and joins at a cost of three keys: $\{B\}_{k+1}$, $\{J\}_A$, and $\{J\}_B$. User $k+2$ requires both B and J to be changed, and four keys are sent: $\{B\}_{k+1}$, $\{B\}_{k+2}$, $\{J\}_A$, and $\{J\}_B$. The number of keys transmitted to initialize N users sums to

$$B_{3\text{dynamic}} = \frac{1}{2}k(k+1) + \sum_{i=1}^{d-1} \left[2k^i \left(\frac{1}{2}k(k+1) - 1 \right) + k^{i-1} \frac{i}{2}k(k-1)(k+1) \right]$$

which simplifies to

$$B_{3\text{dynamic}} = \frac{k+1}{2}N \log_k N - \frac{N}{k-1} + \frac{k}{k-1}.$$

Assuming $N \gg 1$, the minimum life cycle cost $(\partial C_{3\text{dynamic}}/\partial k) = 0$ occurs at

$$f_{3\text{dynamic}} = 0 = \frac{\ln N}{\ln k} \left(\frac{\ln k}{2} + \alpha \ln k - \frac{k+1}{2k} - \alpha \right) + \frac{1}{(k-1)^2}.$$

REFERENCES

- [1] M. Annoni *et al.*, "Interworking between multi-layer IPSEC and secure multicast services over GEO satellites," presented at the COST-272 Symp., Thessaloniki, Greece, June, 20–21 2002. Doc. TD-02-016-P.
- [2] J. Arrko *et al.*, "MIKEY: Multimedia Internet Keying," IETF Internet Draft, work-in-progress, draft-ietf-msec-mikey-06.txt, Feb. 2003, expires Aug. 2003.
- [3] N. Assaf *et al.*, "Interworking between IP security and performance enhancing proxies for mobile networks," *IEEE Commun. Mag.*, vol. 40, pp. 138–144, May 2000.
- [4] D. Balenson *et al.*, "Key management for large dynamic groups: One-way function trees and amortized initialization," IETF Draft, work-in-progress, draft-balenson-groupkeymgmt-01.txt, Feb. 1999.
- [5] M. Baugher *et al.*, "The group domain of interpretation," IETF Internet Draft, work-in-progress, draft-ietf-msec-gdoi-08.txt, May 2003, expires Nov. 2003.
- [6] J. Border *et al.*, "Performance enhancing proxies intended to mitigate link-related degradations," IETF, RFC3135, June 2001.
- [7] R. Canetti *et al.*, "Multicast security: A taxonomy and some efficient constructions," in *Proc. IEEE INFOCOM*, 1999, pp. 708–716.
- [8] D. Harkins and D. Carrel, "The Internet key exchange," IETF, RFC2409, Nov. 1998.
- [9] H. Harney, A. Schuett, and A. Colegrove, "GSAKMP Light," IETF Internet Draft, work-in-progress, draft-ietf-msec-gsakmp-light-sec-01.txt, July 2002, expires Dec. 2002.
- [10] S. Kent and R. Atkinson, "Security architecture for the Internet protocol," IETF, RFC2401, Nov. 1998.
- [11] —, "IP authentication header," IETF, RFC2402, Nov. 1998.
- [12] —, "IP encapsulating security payload (ESP)," IETF, RFC2406, Nov. 1998.
- [13] D. Maughan *et al.*, "Internet security association and key management protocol (ISAKMP)," IETF, RFC2408, Nov. 1998.
- [14] S. Mittra, "Tolus: A framework for scalable secure multicasting," in *Proc. SIGCOMM*, 1997, pp. 277–288.
- [15] M. J. Moyer *et al.*, "A survey of security issues in multicast communications," *IEEE Network*, vol. 13, pp. 12–23, Nov. 1999.
- [16] M. J. Moyer *et al.*, "Maintaining Balanced Key Trees for Secure Multicast," IETF Internet-Draft, work-in-progress, draft-irtf-smug-key-tree-balance-00.txt, June 25, 1999.
- [17] G. Noubir and L. von Allmen, "Security issues in Internet protocols over satellite links," in *Proc. 50th Vehicular Tech. Conf.*, 1999, pp. 2726–2730.
- [18] H. Orman, "The OAKLEY key determination protocol," IETF, RFC2412, Nov. 1998.
- [19] R. Poovendran and J. S. Baras, "An information-theoretic approach for design and analysis of rooted-tree-based multicast key management schemes," *IEEE Trans. Inform. Theory*, vol. 47, pp. 2824–2834, Nov. 2001.
- [20] B. Schneier, *Applied Cryptography*. New York: Wiley, 1996.
- [21] S. Setia *et al.*, "Kronos: A scalable group re-keying approach for secure multicast," in *Proc. IEEE Symp. Research Security Privacy*, May 2000, pp. 215–228.
- [22] J. Snoeyink, S. Suri, and G. Varghese, "A lower bound for multicast key distribution," in *Proc. IEEE INFOCOM*, Apr. 2001, pp. 422–431.
- [23] D. Wallner, E. Harder, and R. Agee, "Key management for multicast: Issues and architectures," IETF, RFC2627, June 1999.
- [24] C. K. Wong, M. Gouda, and S. S. Lam, "Secure group communications using key graphs," *IEEE/ACM Trans. Networking*, vol. 8, pp. 16–30, Feb. 2000.
- [25] Y. R. Yang *et al.*, "Reliable group rekeying: A performance analysis," *Comp. Commun. Rev.*, vol. 31, pp. 27–38, 2001.
- [26] Y. Zhang, "Multi-layer Internet security for satellite & wireless networks," Hughes Res. Lab., HRL Tech. Rep. 99-611, Dec. 1, 1999.



Michael P. Howarth received the B.A. degree in engineering science, the D.Phil. degree in electrical engineering, both from the University of Oxford, Oxford, U.K., in 1981 in 1984, respectively, and the M.Sc. degree in telecommunications from the University of Surrey, Surrey, U.K.

Prior to joining the University of Surrey, he worked for several networking and IT consultancies. His research interests include IP multicast, traffic engineering, security systems, protocol design and optimization, and satellite communications.

Dr. Howarth is a Chartered Electrical Engineer and Member of the Institution of Electrical Engineers (IEE), U.K.



Sunil Iyengar received the B.Sc. degree in electronic engineering from the University of Pune, Pune, India, in 1997 and the M.Sc. degree in telecommunications and software from the University of Surrey, Surrey, U.K., in September 1999, where he is currently working toward the Ph.D. degree in the field of IP network security.

He has been a Research Fellow at the Centre for Communication Systems Research (CCSR), University of Surrey, since January 2000, and he has worked on several European research projects

including GEOCAST and VIP-TEN.



Zhili Sun (M'99) received the B.Sc. degree in mathematics from Nanjing University, Nanjing, China, in 1982 and the M.Phil. and Ph.D. degrees from the Department of Computing, Lancaster University, Lancaster, U.K.

He is a Reader in the Centre for Communication Systems Research (CCSR), University of Surrey, Surrey, U.K. From 1989 to 1993, he did Postdoctoral Research in the Telecommunications Group, Queen Mary, University of London, London, U.K. He has been principal investigator and technical coordinator

in a number of European R&D projects including the ESPRIT BISANTE project on evaluation of broadband traffic over satellite using a simulation approach, the TEN-telecom VIP-TEN project on quality-of-service (QoS) of IP telephony over satellites, the GEOCAST project on IP multicast over satellites, and the ICEBERGS project on IP-based multimedia conferencing over satellites. He is Leader of the Satellite Networking Group of research fellows and Ph.D. students. He also teaches M.Sc., undergraduate, and industrial courses on satellite networking, computer and data networks, IP networking, and Internet traffic engineering.



Haitham Cruickshank (M'99) received the B.Sc. degree in electrical engineering from the University of Baghdad, Baghdad, Iraq, in 1980, the M.Sc. degree in telecommunications from the University of Surrey, Surrey, U.K., and the Ph.D. degree in control systems from Cranfield Institute of Technology, Cranfield, U.K., in 1995.

He is a Senior Research Fellow at the University of Surrey and has worked there since January 1996 on several European research projects in the ACTS, ESPRIT, TEN-TELECOM, and IST programs. His main

research interests are network security, satellite network architectures, and VoIP and IP conferencing over satellites.

Dr. Cruickshank is a Member of the Satellite and Space Communications Committee of the IEEE Communications Society and is also a Chartered Electrical Engineer and Corporate Member of the Institution of Electrical Engineers (IEE), U.K.