The present invention provides software for accomplishing the dynamic management of key information (Ks) for accessing secure (KE) multimedia multicast (k1-kn). The invention additionally concerns computer systems that use such software.
Title of the Invention:

**Dynamic Key Management Architecture for Ensuring Conditional Access to Secure Multimedia Multicast**

Field of the Invention:

The present invention relates to software for accomplishing the dynamic management of key information for accessing secure multimedia multicast, as well as to computer systems that use such software.

Cross-Reference to Related Applications:

This application is a continuation-in-part of United States Patent Applications Serial Nos. 60/289,131 (filed May 8, 2001), and 60/233,841 (filed September 20, 2000), which applications are both herein incorporated by reference in their entirety. This application claims priority from United States Patent Applications Serial Nos. 60/289,131 and 60/233,841.

Interest of the United States Government:

The invention disclosed and/or claimed herein was made in part through the support of the United States Government (National Science Foundation Grant MIP9457397A). The Government may have certain right in this invention.

Background of the Invention:

With the advancement of networking technologies, such as broadband IP and satellite networks, many opportunities have been created for the delivery of bandwidth intensive media such as audio and video. Many of these future multimedia applications will involve group-based scenarios, where users may join and leave at anytime. Multicast communication is the most suitable method for delivering data to groups of users due to its efficient usage of network resources. Over the Internet, for example, the recipients of a group communication are
associated with a Class D IP address, and may receive messages sent to that address. A server that desires to send communication to the group addresses messages with the group address and transmits a single copy of the message. It is the responsibility of the network and the multicast-enabled routers to deliver the message to the users. By sending only a single copy of the message on the network, the usage of server-side resources such as bandwidth and processing is reduced.

Multicasting technologies allow for the simultaneous delivery of information to a group of participants efficiently. Television and radio channels are two classical examples in which information is simultaneously delivered to large groups of users via multicasting technologies. The Internet has also seen a recent development of multicast protocols. The Inter-Domain Routing Protocol (IDRP) and the Distance Vector Multicast Routing Protocol (DVMRP) are two examples of the use of multicasting to allow parties to relay common information.

The convergence of textual, audio, video and other data (multimedia) are altering the needs of the communications and entertainment industries. Already, multimedia content is ubiquitous. Content editing software and hardware, such as digital cameras are allowing users to create content more easily. The availability of the Internet and the Web have encouraged artists, both professional and amateur, to share their creative expressions. In the future, users may desire to personalize their experiences by interacting with the multimedia content, and selecting to composite different media streams to create a unique user experience. In addition, as content becomes easier to create, users will want to share experiences so that they may work together as well as play together.

Technologies are needed that will enable people to access movies, sporting events, news services, weather, etc. efficiently on demand from their homes, offices or from remote or mobile locations. Companies may arise with extensive multimedia databases of movies, television shows, sporting events, graphics data, news clips, etc. that would be broadcast, simultaneously to many users at once.
Regardless of the media in which the transmission occurs, multicast promises to introduce several new commercial areas: for example, the ability to establish remote education domains ("global virtual universities") in which members can attend lectures by specialists from around the globe. Group-based technologies will play a key role in delivering services shared by many users, such as pay-per-view broadcasts of sporting events, as well as allowing for interactive multimedia applications such as interactive television, video conferencing, and communal gaming. Medical and professional teleconferencings are already established applications that promise future growth.

Integral to many of these future ventures will be the ability to broadcast or multicast identical data simultaneously to groups of users. Multicast and broadcast communications are efficient, reducing the demands placed on network and on bandwidth resources. The development of multimedia multicast technologies is thus important not only to governments and organizations, but even to average consumers. The Internet for example, has seen a rapid increase in sites delivering streaming video and audio. The demand for multimedia seems destined only to grow as new technologies such as DVD, HDTV, and wireless communications become more developed and reach the market. The combination of well-developed multimedia standards, such as MPEG-4 and H.324, and advances in both wireless and networking technologies are creating opportunities for new commercial markets such as HDTV, wireless video, and pay-per-view services. In the future, as such standards become adopted and set into practice, an increasing amount of data and multimedia services will be provided to the public through wireless broadcast.

However, the large size of the data conveyed in multimedia information makes it inefficient to use point-to-point communication, particularly in wireless environments where one could devote a time slot, frequency slot, or code slot to a single broadcast and designate to the users which resource slot corresponds to their broadcast. Even across a medium such as the Internet, establishing a point-to-point connection between the server and every user wishing to use the service is inefficient as it can lead to over-burdened network links if the unicast routes overlap.
Multimedia, therefore, is a prime candidate for using multicast and broadcast techniques. In addition, multimedia data is suitable for data embedding, which is the almost invisible hiding of extra information in the data structure.

However, before such commercial ventures can be successfully deployed, the issue of controlling access to multimedia content must be addressed. Service providers must be able to ensure the availability of multimedia data to privileged (i.e., paying) members while preventing unauthorized use of this data by non-privileged users.

The problem of access control is more difficult when the content is being distributed to a group of users since the membership will most likely be dynamic in nature, with users independently joining and leaving the service. Unlike unicast communication, the departure of a group member does not imply the termination of the communication link. In addition, upon departing the service, users must be de-registered so that they can be prevented from obtaining future broadcasts. Similarly, when new members join the service, it is desirable to prevent them from accessing past content. Both of these scenarios might arise in conferences where prior and future information might be confidential, and meant for select subgroups. In addition, any solution to access control should address issues of resource scalability for scenarios involving large groups of privileged users.

In sum, technological advancements have led to the convergence of audio, video, and graphics in current multimedia. Well-developed standards have created many new applications, and are allowing users to share content and express their creativity. Many future multimedia technologies will involve group-based scenarios, where users work and play together. The most relevant enabling network technology for group communication is multicast, which supports the requirements of both the service providers and the end user. However, adaptation of multicast into commercial and selective multimedia applications depends on the ability to secure the communications, and the use of multicasting is however presently limited by problems of security, control and scalability. Several approaches for achieving
such secure multicasting have been described (see, e.g., U.S. Patents Nos. 6,230,205 (Garrity et al.), 6,195,751 (Caronni et al.), 6,170,061 (Beser), 6,154,463 (Aggarwal et al.), 6,119,163 (Monteiro et al.), 6,049,878 (Caronni et al.), 6,002,768 (Albanese et al.), 5,987,518 (Gotwald), 5,949,877 (Traw et al.)).

Despite such efforts, a need remains for technology that would be able to facilitate the controlled-access delivery of information, particularly multimedia information, to subscribers, or to authorized recipients. The present invention addresses such a need, and provides such an advance by specifying new key management architectures that exploit either a residue-based formulation or a polynomial interpolation formulation, as well as a method for distributing key maintenance information using data embedding technology. By using data embedding, key updating messages associated with secure multicast key management schemes may be hidden in the multimedia data and used in conjunction with encryption to protect the content from unauthorized access.

Summary of the Invention:

The problem of controlling access to multicasts in scenarios where the group membership is dynamic is critical to the success of many future applications. Encrypting the bulk data with a session key provides access control to multicast content. In order to prevent non-privileged users from accessing content, it is necessary to update the session key upon changes to the membership profile. Intermediary keys are used to update the session key, as well as themselves. Since many application domains will consist of large group sizes, the key management protocol must be efficient. The present invention provides a key management architecture that is suitable for managing the keys needed to secure multicasts. In its preferred embodiments, the protocols of the present invention employ a tree-structured key hierarchy, and basic primitive operations to construct more advanced protocol operations that allow for users to join and depart the service, to recover membership if rekeying messages are missed, and to transfer access rights temporarily or permanently to other users. By using a scheme that utilizes either
residue-based arithmetic operations or polynomial interpolation in conjunction with a broadcast seed and one-way functions, the amount of communication overhead during member departure is reduced when compared to conventional methods that require that a user index string be sent. A flag-based protocol may be employed that informs users of the messages that are intended for them. Using the performance analysis of the update communications under member addition and deletion, one can obtain the optimal degree of the rooted tree-based scheme. A stochastic population model is provided that allows one to study the mean behavior of a key tree under different degrees of occupancy. Additionally, the invention provides results comparing the computational requirements of the methods of the present invention with a residue-based scheme.

The present invention thus provides a computer software protocol for managing the controlled distribution of multimedia in a multicast scenario. The invention additionally introduces a new paradigm to multicast security by providing a way to conduct secure multicasting through the exploitation of the properties of the data that is being sent. Thus, rather than treating all transmitted data as generic, as is typically done in multicast technologies, in accordance with the protocols of the present invention, the special characteristics of multimedia data are exploited to convey the messages and information required for registration and deregistration in the privileged group.

In detail, the invention provides a computer-facilitated protocol for accomplishing the dynamic management of key information between a group center and one or more group members in order to provide a secure multicast of multimedia data, wherein the protocol permits the Group Center to provide key information to the group members through the use of a parametric one-way function and a broadcast seed (especially a random broadcast seed or a broadcast seed comprising key information previously transmitted to the one or more group members), or through the analysis of data subjected to modulo analysis by a group center computer using previously sent key information.
The invention also provides a computer system comprising a group center computer and at least one group member computer, wherein in the system, the group center computer securely transmits multimedia data to at least one group member computer wherein the security is accomplished through the use of a parametric one-way function and a broadcast seed, or through the analysis of data subjected to modulo analysis by a group center computer using previously sent key information.

The invention also provides a computer specially adapted by software to be able to function as a group center computer and to securely transmit multimedia data to at least one group member computer wherein the software permits the specially adapted computer to accomplish the security through the use of a parametric one-way function and a broadcast seed, or through the analysis of data subjected to modulo analysis by a group center computer using previously sent key information.

The invention also provides a computer specially adapted by software to be able to function as the recipient of securely transmitted multimedia data, wherein the software permits the specially adapted computer to accomplish the receipt through the analysis of a parametric one-way function and a broadcast seed transmitted from a group center computer, or through the analysis of data subjected to modulo analysis by a group center computer using previously sent key information.

The invention further provides the embodiments of such computer-facilitated protocols, computer systems or specially adapted computers, wherein the key information comprises a session key, and one or more personal key encrypting keys, and optionally an embedding key. The invention further provides the embodiment of such computer-facilitated protocols wherein the key information is provided using a media-independent channel or a media-dependent channel.

The invention further provides the embodiment of such computer-facilitated protocols, computer systems or specially adapted computers, wherein the parametric one-way function is a parametric one-way function $h$ that maps a sequence $(x,y)$ of $2B$ bits to $B$ bits, or wherein the parametric one-way function additionally comprises a function $f(x,y) = 1 \| h(x,y)$, wherein $f$ optionally prepends a single 1 bit in front of
the output of $h(x,y)$. The invention further provides the embodiment of such computer-facilitated protocols, computer systems or specially adapted computers, wherein the output of the parametric one-way function is used to mask user-specific security data that is involved in the residue-based method or polynomial interpolation method.

The invention further provides the embodiment of such computer-facilitated protocols, computer systems or specially adapted computers, wherein residue-based method specifically refers to any employment of rekeying messages of the form:

$$\alpha_i(t) = K_s(t) + \prod_{i=1}^{z_i} f(K_{s,i}, \mu(t)).$$

The polynomial interpolation method includes the use of the parametric one-way function in forming polynomials $p(x)$ that interpolate points $(z_j, w_j)$, where $z_j$ is a quantity that identifies user $j$, or a node of a tree-based hierarchy, and the value $w_j$ is given by:

$$w_j = K_s(t) + f(K_{s,j}, \mu(t)) \pmod{p}$$

The invention further provides the embodiment of such computer-facilitated protocols, computer systems or specially adapted computers, wherein the multimedia data is provided to the group member in multi-layered form, and particularly wherein the protocol employs a tree-based algorithm to update the session key and the personal key-encrypting key. The invention further provides the embodiment of such computer-facilitated protocols, computer systems or specially adapted computers, wherein the protocol or software accomplishes the dynamic management of key information in conjunction with the MPEG-4 technology, and specifically the MPEG-4 Intellectual Property Management and Protection framework.

The invention further provides the embodiment of such computer-facilitated protocols, computer systems or specially adapted computers, wherein bits containing key data are embedded in said multimedia data by changing half-pixel motion estimation to an integer-pixel motion vector of said multimedia data, and in particular, wherein two bits of key data $B_n B_{n+1}$ are embedded in one motion vector of
said multimedia data by specifying a set \( S_m \), which a motion vector will belong to
using \( m = 2B_n + B_{n+1} \), wherein the motion vector \((D_x, D_y)\) of one multiblock in frame
\( k \) is determined by:

\[
(D_x, D_y) = \arg\min_{(d_x, d_y) \in S_m} \sum_{i=0}^{15} \sum_{j=0}^{15} \left| f^k(i, j) - f^{k-1}(i + d_x, j + d_y) \right|
\]

where \((d_x, d_y)\) is a motion vector candidate corresponding to the pixels in set
\( S_m \).

**Brief Description of the Figures:**

**Figure 1** illustrates a simple key distribution scheme for \( n \) users.

**Figures 2(a), 2(b), and 2(c)** illustrates the sequences involved in updating
the key information during member additions and deletions using data embedding.
**Figure 2(a)** corresponds to updating the root KEK; **Figure 2(b)** corresponds to
updating the session key, and **Figure 2(c)** corresponds to changing the embedding
rule.

**Figures 3(a) and 3(b)** shows a tree-based key management architecture.
**Figure 3(a)** depicts a binary tree. **Figure 3(b)** depicts a ternary tree-based
distribution.

**Figures 4(a) and (b)** illustrate the two message structures used in the
primitive protocols.

**Figures 5(a) and (b)** illustrate the amount of communication required in
worst case scenarios of member join operations (**Figure 5(a)**) and member departure
(**Figure 5(b)**) for different tree degrees \( a \) and different amounts of users \( n \).

**Figure 6** shows the average \( C_{MD} \) and \( C_{AD} \) for different tree degrees \( a \) and
different amounts of users \( n \).
Figures 7(a) and (b) show the expected amount of communication for a degree 4 tree with 6, 8, and 10 levels as a function of the probability $q$ that a leaf node is occupied. Figure 7(a) shows the result for member join; Figure 7(b) shows the result for member departure.

Figure 8 shows the worst-case member departure communication overhead required in a conventional tree-based rekeying for different tree degrees versus the baseline communication required when using the polynomial interpolation scheme. The baseline communication corresponds to $B\mu = 64$ bits.

Figure 9 illustrates two approaches to distributing key information in multicasting.

Figure 10 illustrates a general information embedding scheme. In the scheme, information $m$ is embedded in the host signal $x$. A noise $n$ corrupts the composite signal $s$. The decoder extracts the estimate $\hat{m}$ and reconstruct signal $x$ from channel output.

Figure 11 illustrates the integer-pixel motion prediction in H.263: the block (with dashed line) at integer pixel $A$ in frame $K$ is the motion prediction of the current block (with solid line) in frame $K+1$.

Figure 12 illustrates the half-pixel motion prediction in H.263. The half-pixel motion vector is found by looking for the minimum SAD (Sum of Absolute Difference) among the half-pixels 1~8 and integer pixel $A$.

Figure 13 illustrates key transmission by data embedding in H.263/MPEG2 half-pel based video coding standards.

Figures 14(a) and 14(b) illustrate the results of two simulations (Foreman (Figure 14(a)) and Miss America (Figure 14(b))) in which the peak signal-to-noise ratio (PSNR) of luminance component with different data embedding rates are compared with the PSNR of luminance without embedding.
Figure 15 illustrates the time needed to refresh an entire set of keys during a member departure using the bottom-up approach with different frame rates $F$, and different amounts of bits embedded per frame. The group size is $n = 2^{20}$ (approximately one-million users).

Figure 16 illustrates block diagram of a generalized hybrid video coding scheme with motion-compensating prediction.

Figure 17 shows the general form of Motion Compressed Prediction (MCP), where $(x,y)$ denotes spatial coordinates.

Figure 18 illustrates a multicast scenario. The service provider distributes content gathered from many content providers and delivers content to multimedia terminals with different profiles.

Figure 19 illustrates a key distribution scheme for multi-layer multimedia multicast.

Figure 20 illustrates a media-independent framework for distributing keys with IPMP bitstreams.

Figure 21 illustrates the IPMP media-dependent framework for distributing keys using data embedding.

Description of the Preferred Embodiments:

The present invention relates to computer-facilitated protocols for accomplishing the dynamic management of key information for accessing secure multimedia multicast, as well as to computer systems that use such protocols for ensuring the security of a multicast transmission. As used herein, the term "key" refers to data related to the authentication of the identity of the data transmitter, and is used to perform encryption of content or maintenance of other keys.

The term "computer-facilitated protocol" refers to a computer-controlled decisional process or processes that employ software and hardware devices, and that
involve data exchange, analysis and/or processing. The term “computer,” as used herein is intended to encompass not only mainframe or base computer systems, but to generally include any device (e.g., personal computers, data processors, switching systems, telephones, fax machines, PDAs, 2-way radios, etc.) that is capable of processing data. Likewise, the term “transmission” is intended to encompass wired, wireless, broadcast, optical fiber, microwave, etc., methods of transmission. As will be appreciated, in accordance with the present invention, signals may be exchanged between a content provider and a data recipient, or between data recipients. Thus, both the content provider and the data recipients can serve as transmitters and receivers of data.

The computer-facilitated protocols of the present invention permit the secure transmission of data. As used herein, the terms “secure” and “security” are intended to denote that a transmission cannot be accessed by an unauthorized user without surmounting a substantial access burden (i.e., a time burden, a computational power threshold, and/or a communication resources burden). The term “time burden” is intended to denote the amount of time that would be required in order to access a signal. The term “computational power burden” is intended to denote the data processing capacity that would be required to attain such access. The term “communication resources burden” is intended to denote the receiver, signal intercept, or signal monitoring capacity that would be required to attain unauthorized access.

More specifically, the security provided by the secure protocols of the present invention will be sufficient to increase the access burden faced by an unauthorized user to such a level that unauthorized access could be attained within the duration of the transmission only at the expense of substantial time, computational power and/or communication resources. More preferably, the security provided by the secure protocols of the present invention will be sufficient to increase the access burden faced by an unauthorized user to such a level that unauthorized access could not be attained within the duration of the transmission regardless of the computational power or communication resources dedicated to the
access attempt. Most preferably, the security provided by the secure protocols of the present invention will be sufficient to increase the access burden faced by an unauthorized user to such a level that unauthorized access could not be attained regardless of any time, computational power or communication resources that could reasonably be dedicated to the access attempt. Thus, for example, the security provided by the protocols of the present invention might merely be sufficient to so encumber efforts to attain unauthorized access that potential users would be effectively required to seek authorization. Alternatively, the security provided by the protocols of the present invention might be sufficient to bar individuals and entities from accessing a transmission during its duration, but insufficient to prevent such individuals and entities from thereafter accessing such transmission (or deducing how they could have attained access). Alternatively, the security provided by the protocols of the present invention might be sufficient to bar individuals and entities from accessing a transmission despite any "real world possible" commitment of time, computational power or communication capacity.

The computer-assisted protocols and computer systems of the present invention may thus be used in a variety of transmission arrangements. For example, the invention may be used to facilitate and/or control the transmission of "pay-per-view" television/radio signals, satellite television/radio programming, etc. Through the use of the present invention, the content administrator (e.g., the provider of the programming content) would be able to block access to unauthorized users. The computer-assisted protocols and computer systems of the present invention may additionally be used in any transmission in which confidential or proprietary information is desired to be exchanged, and in which one, more than one or all of the parties has a need to establish the authorization of one, more than one or all of the other parties to a transmission.

For example, financial data and instructions (stock ownership information and investment instructions, account balance information, account transfer instructions, payment instructions, credit information, etc.), corporate information (sales, marketing, strategic data or information, conference calls (especially video
conference calls), scientific data (e.g., data, such as measurements, etc., in which the integrity of the data transmission is important), governmental communications (in particular diplomatic or military communications (e.g., battlefield, tactical, and/or command and control communications, weapon systems control, etc.), etc.

5 I. Overview of the Invention

The distribution of data, such as multimedia, can be accomplished via a unicast or a multicast approach. A unicast distribution system involves the separate transmission of data to each user, and is inefficient, requires significant communication resources, and is amenable to use only by limited numbers of users.

The distribution of time or weather information over a telephone line is an example of a unicast data distribution approach. In contrast, a multicast approach seeks to provide data to multiple users collectively. Broadcast television, and pay-per-view programming are examples of multicast data distribution systems. In a multicast system, if 10 users subscribe to the system, the multicast server prepares one copy of the transmission, and the multicast (network) routers duplicate the data as needed when distributing to each user. Multicast systems are thus able to distribute data to larger numbers of users than can be served with a unicast system. A need exists for multicast systems that can expand to meet the demands of large numbers of users, and that can in addition provide content only to authorized users. The present invention is directed to such improved multicast technologies.

The present invention thus provides an approach to the distribution of multicast key information that is suitable for commercial multicast use, including uses that employ MPEG-4 Intellectual Property Management and Protection systems. The invention provides a standardized solution to group key management in the MPEG-4 IPMP framework, and thus allows easier deployment of multimedia multicast applications. As such, the present invention permits the building of a new infrastructure for the delivery and consumption of multimedia content.

In the sections below, the problem of controlling access to multimedia multicast using the distribution and maintenance of key information, even for
dynamic groups, is discussed. These sections provide a general framework for multicast/broadcast technologies used in conjunction with multimedia content, and illustrate the invention with several exemplary methods for key distribution for multimedia. The first exemplary method for distributing key information involves a protocol that is independent of the content, while the second approach uses the content itself to convey the keys by means of a data embedding technology. It is observed that using data embedding to convey rekeying messages can provide an additional layer of security against external attacks when compared with the traditional media-independent method. The preferred method of the present invention is an efficient rekeying protocol that employs a tree-based key structure to achieve logarithmic scalability in the communication requirements needed to maintain the keys during membership changes. The preferred method of the present invention has the ability to handle dynamic groups, and is suitable for: multicast, key management, user/content mobility, and multimedia. The preferred method allows efficient access control to multiple layers of multimedia content. Section II discusses key management in multicast communications. Section III discusses the operation of the multicast. Section IV discusses scalability of the key distribution scheme. Section V discusses implementation issues. Section VI discusses key distribution. Section VII discusses data embedding considerations.

In Section VIII, the amenability of the invention to multilayered services is described. Example 1 focuses on the application of multicast key management schemes in conjunction with the MPEG-4 Intellectual Property Management and Protection system as described in ISO/IEC 14496–1.

The present application specifically focuses on the use of the methods of the present invention in securing multicasts of MPEG-4 content to groups of users with different terminal profiles. The member join and member departure operations of the multicast key management of the preferred schemes of the present invention provide a powerful framework for addressing the needs of multicast security.
By using data embedding, key updating messages associated with secure multicast key management schemes may be hidden in the data and used in conjunction with encryption to protect the data from unauthorized access.

II. Key Management in Multicast Communications

A. General Considerations

The distribution of identical data to multiple parties using the conventional point-to-point communication paradigm makes inefficient use of resources. The redundancy in the copies of the data can be exploited in multicast communication by forming a group consisting of users who receive similar data, and sending a single message to all group users.

The efficiency in multicast communication has created many new application areas, and made others more feasible (S. Paul. “Multicasting on the Internet and its Application,” Kluwer Academic, 1998). For the commercial success of most of these applications, it is essential to control access to the data so that only members of the multicast group have access to the data. In order to provide access control to the multicast communication, the data is typically encrypted using a key that is shared by all legitimate group members. The shared key, known as the session key ("SK" or $K_s$), will change with time, depending on the dynamics of group membership as well as the desired level of data protection. Since the key must be changeable, the challenge is in key management – the issues related to the administration and distribution of keying material to multicast group members.

In order to update the session key, the party responsible for distributing the keys (called the group center ("GC")), must be able to securely communicate with the users in order to distribute new key material. The GC shares keys, known as "key encrypting keys" ("KEKs"), that are used solely for the purpose of updating the session key and other KEKs with group members.

During the design of a multicast application, several issues should be kept in consideration when choosing a key distribution scheme:
1. **Dynamic nature of group membership:** The distribution scheme should preferably be able to efficiently handle members joining and leaving as this necessitates changes in the session key and possibly any intermediate keying information.

2. **Security of the key distribution scheme to observations by non-members:** The rekeying messages must be assumed to be available to clever adversaries that are not members of the group. It is important that these adversaries cannot acquire the new session key or any intermediate keying information (such as key encrypting keys) without considerable computational effort.

3. **Ability to prevent member collusion:** No subset of the members should be able to collude and acquire either future session keys or the key encrypting keys of other members.

4. **Scalability of the key distribution scheme:** The communication, storage, and computational resources should have good scalability as the group size increases. In many applications the size of the group may be very large and possibly on the order of several million users. In these cases, the efficiency of the scheme to handle these large groups is critical and should not become a hindrance to providing the service. Several parameters that are relevant here are (i) communication/bandwidth needed to relay key updating messages; (ii) computational requirements needed by the sender to form the new key updating message; (iii) computational requirements needed by the receiver to extract the new key information; (iv) storage needed by the sender to keep track of key information; and (v) storage needed by the receiver for storing intermediate key material. The required communication, storage, and computational resources should not become a hindrance to providing the service as group size increases.
The significance of scalability can be illustrated by considering the following example of a multicast key distribution scheme: a multicast group consists of \( n \) users and the group center shares a key encrypting key with each user. Upon a member departure, the previous session key is compromised and a new session key must be given to the remaining group members. The GC encrypts the new session key with each user's key encrypting key and sends the result to that user. Thus, there are \( n-1 \) encryptions that must be performed, and \( n-1 \) messages that must be sent on the network. The storage requirement for each user is 2 keys while the GC must store \( n + 1 \) keys. This approach to key distribution has linear communication, computation and GC storage complexity. As \( n \) becomes large, these complexity parameters make this scheme undesirable.

The problem of designing efficient key updating schemes has seen recent attention in the literature. One approach for achieving scalability is to apply hierarchical subgroups and map the KEKs to a logical tree. The tree-based approach to group rekeying was originally presented by D.M. Wallner et al. (“Key Management For Multicast: Issues And Architectures. Internet Draft Report.” ftp://ftp.ietf.org/internet-drafts/draft-wallner-key-arch-01.txt) and independently by C. K. Wong et al. (“Secure Group Communication Using Key Graphs,” In: Proceedings of AGM SIGCOMM, Vancouver, Canada, 1998). In such schemes an \( \alpha \)-ary tree of depth \( \log_\alpha n \) is used to break the multicast group into hierarchical subgroups. Each member is assigned to a unique leaf of the tree. KEKs are associated with all of the tree nodes including the root and leaf nodes. A member has knowledge of all KEKs from his leaf to the root node. Thus, some KEKs are shared by multiple users. The addition of new members to the group amounts to adding more depth to the tree, or adding new branches to the tree. Upon member departure, the session key and all the internal node KEKs assigned to that member become compromised and must be renewed. Due to the tree structure, the communication overhead is \( O(\log n) \), while the storage for the center is \( O(n) \).

The \( O \) notation is presented to indicate that the constant factors are implementation dependent. In Poovendran, “An Information Theoretic Approach For Design And
Analysis Of Rooted Tree-Based Multicast Key Management Schemes,” *Advances in Cryptology: Crypto ’99*, pages 624-638, 1999, it was shown that the optimal key distribution for a group leads to Huffman trees and the average number of keys assigned to a member is related to the entropy of the statistics of the member deletion event.

Various modifications to the tree scheme have been proposed. In R. Canetti *et al.*, “Multicast security: a taxonomy and some efficient constructions,” (http://www.cs.utexas.edu/users/nevil/reflinks.htm), a modification to the scheme of Wallner *et al.* is presented. By using pseudo-random generators, their scheme reduces the usage of communication resources by a factor of two. Similarly, Balenson, Mc Grew and Sherman (“Key Management For Large Dynamic Groups: One-Way Function Trees And Amortized Initialization”; Internet Draft Report; http://download.nai.com/products/media/pgp/pdf/draft-irtf-smug-groupkeymgmt-00.PDF) were able to reduce the communication requirements by a factor of two using one-way function trees. The security of the Canetti *et al.* scheme can be rigorously proven, while the security of the approach using one-way function trees is based upon non-standard cryptographic assumptions and has therefore not been rigorously shown. In R. Canetti *et al.*, “Efficient communication-storage tradeoffs for multicast encryption,” *Presented Eurocrypt ’99* 1999, Canetti *et al.* examine the tradeoffs between storage and communication requirements, and present modifications to the Wallner *et al.* and Wong *et al.* schemes that achieve sublinear storage.

**B. Approaches To Multicast Key Management**

In accordance with the present invention, two approaches to multicast key management are provided. The first approach is termed a “Residue” approach. In this approach, the transmitted data is subjected to modulo analysis using previously sent key information, and the new key information is obtained as the residue of such analysis. The second approach is termed an “Interpolation” approach. In this approach, the group center uses the existing key information and interpolates to obtain the coefficients of a polynomial which includes the new key information.
Since authorized users know their own keys, they can calculate the polynomial, and obtain the new key information. Although each such approach has certain advantages in design, architecture and/or implementation, either may be used in accordance with the protocols and computer systems of the present invention.

1. Residue Approach

(a) Lower Security Key Management Schemes

A simple group communications scenario considered is shown in Figure 1. There is one Group Center (GC) and a user set \( u(j) = \{u_1, u_2, ..., u_n\} \) that may vary with time. Consider a group of \( n \) multimedia users who will share a multimedia multicast. In a lower security key distribution scheme for \( n \) users, user \( u_j \) has two key encrypting keys \( K_j \) and \( K_s \) and the session key \( K_e \). The session key \( K_e \) is used to encrypt bulk quantities of multimedia content. The KEK \( K_e \) is the root KEK and is used to encrypt messages that update \( K_e \). The remaining keys \( K_1, K_2, ..., K_n \) are KEKs that are used to protect updates of \( K_e \).

If a new user joins at time \( t \), and the previous root KEK was \( K_e(t-1) \), then the rekeying message used to update \( K_e \) is:

\[
\alpha(t) = K_e(t) + K_e(t-1)
\]  

(1)

and the group members can calculate the new root KEK using the old root KEK. The new member receives the new session key \( K_e(t) \) directly from the GC upon registration to the multicast group.

When a member departs the group, the rekeying message \( \alpha(t) \) is defined as

\[
\alpha(t) = K_e(t) + \prod_{i \in I(t)} K_i
\]  

(2)

where \( I(t) \) is the index set associated with the users of the updated member set \( U(t) \). Every legitimate user \( u_i \) can get the new root KEK \( K_e(t) \) using his/her personal key \( K_i \) by a simple modulo computation after \( \alpha(t) \) is obtained:
\[ K_e(t) = \alpha(t) \pmod{K_i} \] (3)

By doing so, the new \( K_e(t) \) can be distributed to only the legitimate group members. It should be noted that \( K_e(t) < K_i \) for \( u_i \in U(t) \) in this scheme because of the modulo computation. Therefore, the \( K_i \) should be large enough to cover the desirable range for \( K_e(t) \). In practice, the bit length of \( K_i \) is greater than the bit length of \( K_e \) by several bits and the most significant bit of all \( K_i \) is set to 1.

The basic structure of the rekeying message is easily seen to be insecure to an inside attack. Suppose that user \( u_i \), receives the message \( \alpha(t) \), then he can calculate \( K_e(t) \) and determine \( \prod_{i=1}^{n-1} K_i \). Thus, other user’s personal keys \( K_i \) can be calculated by factoring \( \prod_{i=1}^{n-1} K_i \). Factoring the product is easily possible since the \( K_i \) are typically less than 100 bits.

A natural approach to remedying this situation is to incorporate a random factor into the product. In this case, the message becomes:

\[ \alpha(t) = K_e(t) + \lambda(t) \prod_{u_i(t)} K_i \] (4)

where \( \lambda(t) \) is a random integer that is known only by the GC. This approach does not significantly improve security when an adversary collects observations of \( \alpha(t) \) at different times \( t \). This fact will be addressed shortly.

It is also desirable to allow users to depart the group briefly and be easily reinserted into the service. This can be accomplished by a simple modification to the rekeying message. If, at time \( t \), a random non-negative integer \( \mu(t) \) is generated and broadcast to all group members, the new rekeying message is defined by:

\[ \alpha(t) = K_e(t) + \lambda(t) \prod_{u_i(t)} (K_i + \mu(t)) \] (5)
Members can easily acquire the new session key by calculating $\alpha(t) \mod (K_i + \mu(t))$. If member $u_n$ wants to depart the group for a short period, then a rekeying message is formed that does not contain his id $K_n$. The new session key $K_s(t)$ will not be available to him. When the user decides to return to the service a new personal key need not be issued to him, instead he may just receive the broadcast $\mu(t)$ and $\alpha(t)$.

Assume that an inside adversary is user $n$ and that a user has departed the system. If, while he was a member of the service, he collected $T$ observations of $\alpha(t)$, then he also had the information $K_n$ and $\mu(t)$ for those $T$ observations. Thus, he would be able to calculate the corresponding $K_s(t)$, and the quantities:

$$A(t) = \frac{\alpha(t) - K_s(t)}{K_n + \mu(t)}$$ (6)

and his task is to determine a $K_i$ given these $T$ observations. If we define $g(\mu)$ as

$$g(\mu) = \prod_{i=1}^{n-1} (K_i + \mu)$$ (7)

then $g(\mu)$ is a degree $n-1$ polynomial in $\mu$. The observations can then be related to this polynomial by

$$A(t) = \lambda(t) g(\mu(t))$$ (8)

The task of the adversary is to determine the roots $-K_j$ of the polynomial $g(\mu)$ given the noisy observations $A(t)$.

The first observation is that having $A(t)$ random is essential to preventing the adversary from determining another user's personal key. If $A(t)$ is not random, then the adversary may observe $A(t) = g(\mu(t))$, and with $n$ observations can determine the polynomial $g(\mu)$ by interpolation. Having $\mu(t)$ time-varying is also essential to protecting the personal keys. If $\mu(t)$ is constant, without loss of generality one can assume $\mu(t) = 0$, then the following theorem allows an adversary to be able to acquire $\prod_{i=1}^{n-1} K_i$ in a few observations.
Theorem 1: Let \( C(t) = \lambda(t) \prod_{j=1}^{n-1} K_j \) where \( \lambda(t) \) is a random, time-varying integer and the \( K_j \) are known integers. Then

\[
\gcd(C(1), C(2), \ldots, C(T)) \rightarrow \prod_{i=1}^{n-1} K_i
\]

as \( T \rightarrow \infty \).

The convergence is very rapid. In fact, with just \( T = 6 \) observations there is a greater than 98% chance of the greatest common divisor returning \( \prod_{i=1}^{n-1} K_i \). By factoring \( \prod_{i=1}^{n-1} K_i \), one can have considerable amount of information regarding the \( K_i \).

One approach to attacking the \( A(t) = \lambda(t) g(\mu(t)) \) when \( \lambda(t) \) is random and \( \mu(t) \) time-varying is to collect \( A(t) \) observations corresponding to small \( \mu(t) \) values. If \( \mu(t) \) is small, then \( g(\mu(t)) = \prod_{i=1}^{n-1} K_i \). Therefore, averaging over these \( A(t) \) will give:

\[
E[A(t)] = E[\lambda(t)] \prod_{i=1}^{n-1} K_i
\]

(10)

If the adversary can estimate \( E[\lambda(t)] \) then he can get close to \( \prod_{i=1}^{n-1} K_i \). He may then try factoring numbers in a neighborhood of \( \prod_{i=1}^{n-1} K_i \). Since he knows that the \( K_j \) are not large, his computational burden is lessened by not trying to factor numbers that take too long to factor.

(b) Higher Security Key Management Schemes

(1) Use of “One-Way” Functions

It is desirable to have a single, homogenized message from which each user may extract the new root KEK. Such an approach provides a higher level of key management security, which may be desired in certain circumstances.
The problem of distributing keying information simultaneously to multiple users via a single broadcast message while maintaining user anonymity has been previously studied in the literature. Just et al. (M. Just et al., "On key distribution via true broadcasting," in Proc. 2nd ACM Conf. on Computer and Communications Security, pages 81-88, 1994) and Blundo et al. C. (Blundo et al., "Multiple Key Distribution Maintaining User Anonymity Via Broadcast Channels." J. Computer Security 3:309-323, 1994) each present a method using polynomial interpolation whereby the broadcast message does not have a partitioned structure like the message in Equation (19). A drawback of both of these schemes is that, when used to distribute the same key, each user's secret information is valid for only one transmission, and then is available for other group members to acquire. For example, in the polynomial scheme of Blundo et al. C. (Blundo et al., "Multiple Key Distribution Maintaining User Anonymity Via Broadcast Channels." J. Computer Security 3:309-323, 1994), the x-coordinates used are public knowledge, and a member may use knowledge of the key and other x coordinates to determine other user's secret information. This can be a problem since over time members may acquire other users' secret information and use this knowledge to enjoy the service after they have cancelled their membership. In order to use these schemes when the keying material must be updated multiple times, it is necessary to distribute to each user enough copies of private material to cover the amount of updates needed.

Accordingly, one aspect of the present invention, provides a scheme that allows private keying material to be reused while providing a homogenized message form. This scheme involves a new method of key distribution that makes use of one-way functions and a broadcast seed to protect each user's private information from compromise. To facilitate the exposition of this scheme, the parametric one-way functions that are the building blocks of its message form are described below.

**Definition 1** A parametric one-way function (POWF) \( h \) is a function from \( \mathcal{X} \times \mathcal{Y} \rightarrow \mathcal{Z} \) such that given \( z = h(x, y) \) and \( y \), it is computationally difficult to determine \( x \).
Parametric one-way functions are families of one-way functions (A. Menezes et al., *Handbook of Applied Cryptography*, CRC Press, 1997) that are parameterized by the parameter $y$. The discrete logarithm provides an example of a POWF since if $p$ is a large prime, and $x$ and $y$ are non-identity elements of $\mathbb{Z}_p^*$, the multiplicative subgroup of integers modulo $p$, it is computationally difficult to determine $x$ given $z = y^x \pmod{p}$ and $y$ and a generator of $\mathbb{Z}_p^*$ (A. Menezes et al., *Handbook of Applied Cryptography*, CRC Press, 1997).

Symmetric block ciphers also can be used to construct POWFs. Let $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, and consider a symmetric cipher $E_x(y) : \mathcal{Y} \rightarrow \mathcal{Y}$ where the subscript denotes the key used in the encryption of the plaintext $y$. Thus $\mathcal{X}$ is the key space of the cipher $E$, while $\mathcal{Y}$ is the space of plaintexts and ciphertexts. Define a hash function $f : \mathcal{Y} \rightarrow \mathbb{Z}$. Then the function $h(x, y) = f(E_x(y))$ is a POWF parameterized by $y$ since any reasonable cryptosystem can withstand a known-plaintext attack, that is knowledge of $E_x(y)$ and $y$ does not make it easy to determine the key $x$. The use of symmetric ciphers in this invention includes such ciphers as Rijndael, DES, and RC-4/5, but is not limited to these ciphers. Additionally, note that it is not necessary that the hash function $f$ have any cryptographic properties as the required cryptographic strength is provided by $E$. For the purposes of the present invention, it is assumed that parametric one-way functions exist that map sequences of $2B$ bits into sequences of $B$ bits.

(2) **Basic Message Form**

The key distribution scheme depicted in Figure 1 will be used to illustrate the basic message form of a preferred embodiment of the multicast key distribution scheme of the present invention. Suppose that at time $t-2$, the group consists of $n$ users $u_1, u_2, \ldots, u_n$. Each user $u_i$ has a personal B-bit KEK $K_i$ that is known only by the group center and user $u_i$. Additionally, all of the users share a B-bit root KEK and a session key that will vary with time.
The group center makes available a POWF \( h \) that maps a sequence \((x, y)\) of 2B bits to B bits. A new function \( f \) is defined by prepending a single 1 bit in front of the output of \( h(x, y) \), that is \( f(x, y) = 1 \| h(x, y) \). The purpose of prepending a bit is to ensure that the modulo operation used by each user will yield \( K_e(t) \).

Suppose, without loss of generality, that user \( u_n \) decides to leave at time \( t-2 \), then both \( K_d(t-1) \) and \( K_d(t-1) \) must be updated. The root KEK is updated first, and then used to encrypt the new session key. In order to update \( K_d(t-1) \), the GC first broadcasts a B-bit random seed \( \mu(t) \), or uses previous key information, such as \( K_d(t-I) \), but not limited to \( K_d(t-I) \), as \( \mu(t) \). Next, the GC forms \( K_d(t) \) and calculates the rekeying message as:

\[
\alpha_x(t) = K_x(t) + \prod_{j=1}^{n-1} f(K_j, \mu(t))
\]  
(11)

This is the message form of the residue-based method, and the new rekeying message can be conveyed using either an media-independent or a media-dependent channel. A legitimate member \( u_t \) may decode \( \alpha_x(t) \) to get \( K_d(t) \) by calculating \( \alpha_x(t) \mod f(K_t, \mu(t)) \).

It is computationally hard for one member to compute the private KEK of another member. First, consider a weak variant of this message form:

\[
\alpha_x(t) = K_x(t) + \prod_{j=1}^{n-1} K_j
\]  
(12)

This form of the message requires that \( K_d(t) < \min_i(K_i) \). Although this form of the message distributes \( K_d(t) \) to all of the needy members, it is possible for user \( u_t \) to acquire the keys \( K_j \) of other users since he may calculate:

\[
A_t = \frac{\alpha_x(t) - K_x(t)}{K_j} = \prod_{j \neq t} K_j,
\]  
(13)
and then factor $A_i$ to acquire information about the other $K_j$'s. If the keys $K_j$ correspond to symmetric encryption keys, then the key length will tend to be less than 150 bits, and factoring $A_i$ will not be difficult (A. Menezes et al., *Handbook of Applied Cryptography*, CRC Press, 1997).

By broadcasting $\mu(t)$ and using a non-reversible function $f$, the adversary is instead able to calculate:

$$A_i = \prod_{j=1} f(K_j, \mu(t))$$  \hspace{1cm} (14)

Factoring $A_i$ provides information about Since it is difficult to acquire $K_j$ given $\mu(t)$, and $f(K_j, \mu(t))$, the private user information is protected. At the next time instant, when $\mu(t+1)$ is broadcast, the adversary's knowledge of $f(K_j, \mu(t))$ does not help him in calculating $f(K_j, \mu(t+1))$, and he can extract $K_e$ only if he has the needed keys assigned to him.

2. Interpolation Approach

(a) Use of Key Distribution Scheme

The key distribution scheme depicted in Figure 1 will be used to illustrate the basic message form of a preferred embodiment of the multicast key distribution scheme of the present invention. Each user $u_i$ has a personal B-bit KEK $K_i$ that is known only by the group center and user $u_i$. Additionally, all of the users share a B-bit root KEK $K_r(t)$ and a session key $K_s(t)$ that will vary with time $t$.

If user $u_n$ decides to depart, then the GC must renew the keys $K_r(t-1)$ and $K_s(t-1)$ since they were shared by $u_n$ and the other users. The first step is to send the new $K_r(t)$ to the remaining users. In the polynomial scheme, each user $u_i$ has the distinct pair $(z_i, K_i) \in Z_p \times Z_p$, where $Z_p$ denotes the integers modulo the prime $p$. The $z_i$ are public knowledge, and are not considered as part of the secret information that the user must store. Instead, the $z_i$ is any quantity that is used to identify the user, for example a processor id. The GC has made available $f$, a POWF taking 2B
bits to B bits. The GC first broadcasts the seed $\mu(t)$ to everyone. Next, the GC associates the following quantity with each user $w_j$:

$$w_j = K_s(t) + f(K_j, \mu(t)) \quad (\text{mod } p)$$

(15)

The GC generates a degree $n-2$ polynomial $p(z)$ that interpolates the points $(z_j, w_j)$, i.e., $p(z_j) = w_j$. The GC represents $p(z)$ as:

$$p(z) = \sum_{i=0}^{n-2} c_i z^i \quad (\text{mod } p)$$

(16)

and transmits the message $\alpha_e(t) = (c_0, c_1, \ldots, c_{n-2})$ to update $K_s(t)$. This completes the action needed by the GC to update the root KEK, and the session key is then updated using $K_s(t)$ by transmitting $\alpha_s(t) = E_{K_s(t)}(K_s(t))$. A member $w_j$ can calculate $p(z_j) = w_j$ and $f(K_j, \mu(t))$, and hence can recover $K_s(t)$.

**Resilience to Attack**

There are two sources of adversaries for a key management scheme. The first type of adversary is an *external* adversary. This type of adversary is not a member of the service, but receives the encrypted content as well as the rekeying messages. In order for the external adversary to cheat the service, he must mount a successful attack against the rekeying messages so that he can acquire the session key, which is needed to decrypt the content. The second type of adversary is an *internal* adversary, who is a member that uses the rekeying messages and his knowledge of his keys to attempt to acquire another user’s keys. If an internal adversary can successfully acquire another user’s keys, he may cancel his membership to the service, and use the compromised keys belonging to another user to enjoy the service without further payment. As long as the other user remains with the service, the internal adversary will be able to receive key updates and continue to cheat the service.
In the polynomial scheme, an external adversary receives \( \alpha_e \), as well as \( \alpha_s(t) \). In order for the adversary to acquire the SK, he must mount a successful attack against the cipher used in forming the message \( \alpha_s(t) \). Careful selection of a strong cipher algorithm, such as that described by J. Daemen et al., “Aes proposal: Rijndael,” (See http://csrc.nist.gov/encryption/aes, 2000), will make a successful attack of the SK rekeying message unlikely. Even should a successful attack of the SK rekeying message take place, a future update of the SK would require a subsequent successful attack of the SK rekeying message, which is equally unlikely. Hence, a successful attack against the SK rekeying message would only be a short-lived victory for a pirate.

A second method for acquiring the session key is to attack the message \( \alpha_e \). Given the message \( \alpha_s(t) \), and knowledge of a \( z_j \), it is possible for an adversary to calculate \( w_j \). However, the adversary must either determine \( K_s(t) \) or a user’s \( f(K_j, \mu(t)) \) given \( w_j = K_s(t) + f(K_j, \mu(t)) \) (mod p). The modulo operation makes \( w_j \) independent of either \( K_s(t) \) or \( f(K_j, \mu(t)) \). Should an external adversary successfully attack \( K_s(t) \), then he may acquire the session key. However, upon the next update of the session key, he must make another successful attack upon the root KEK, which is an unlikely event.

The only method for an external adversary to be able to repeatedly acquire the SK is to mount a successful attack on a user’s personal key \( K_j \). This requires a successful determination of \( f(K_j, \mu(t)) \) given \( w_j \), which requires searching a space of order \( p \) possibilities, and then successfully attacking the one-way function to acquire \( K_j \). The strength of the one-way function should be as strong as the strength of the encryption used to protect the SK rekeying message.

With regard to the susceptibility of the original polynomial scheme of Blundo et al. (C. Blundo et al., “Multiple key distribution maintaining user anonymity via broadcast channels,” J. Computer Security, vol. 3, pp. 309-323, 1994) to internal attacks, for simplicity, it is assumed that the same key \( K \) is being distributed to all of the users. Observe that since the \( z_j \)-coordinates are public
knowledge, an internal adversary may calculate \( w_j \), evaluating the interpolating polynomial at \( z_j \). With knowledge of \( w_j \), the adversary may use his knowledge of \( K \) to determine user \( u_j \)'s private information. Thus, the polynomial scheme of Blundo et al. does not protect the private information of each user, and hence cannot be used more than once. For this reason, such a scheme has been termed a “one-time” broadcast key distribution scheme. If both the \( z_j \) coordinate and the personal key \( K_j \) are kept secret, then an adversary’s task is to search \( Z_p \) for any of the \( n \) user’s \( z_j \) coordinate. This is more difficult for an adversary to attack, but also requires both the server and the clients to store twice as much secret information.

As described above, the methods of the present invention pursue a different approach to ensuring the sanctity of each user’s private information in order to reduce the communication overhead in our protocol. An inside adversary \( u_i \) who desires to calculate another user’s key information \( K_j \) can calculate \( p(z_j) = w_j \), and therefore can calculate \( f(K_j, \mu(t)) = w_j \cdot K_i(t) \pmod{p} \). However, it is difficult for the adversary to calculate \( K_j \) given \( \mu(t) \) and \( f(K_j, \mu(t)) \) since \( f \) is a parametric one-way function. Additionally, should two or more users collude, their shared information would not provide any advantage in acquiring another users \( K_j \). They must still attack the one-way function in order to acquire \( K_j \).

(c) Anonymity of a Preferred Key Distribution Scheme Reduces Communication Overhead

The scheme described in Section II.B.2.(b) is used in constructing a protocol primitive. In the protocol primitive, there is a parent key \( K_p \), and a handful of sibling keys \( K_j \) that are used to update the parent key. Unlike the scheme described in Section II.B.2.(b), application of the protocol primitive might not use all of the sibling keys to update the parent key. This scenario might occur when the GC knows that a sibling key has become compromised or invalidated.

Suppose that there are \( a \) possible sibling keys and that \( m \) of those sibling keys are used to update the parent key. In a conventional key distribution scheme, the update to the parent key is performed by a rekeying message of the form:
\[
\alpha = \left\{ E_{K_{j_1}}(K_x) \| E_{K_{j_2}}(K_x) \| \ldots \| E_{K_{j_m}}(K_x) \right\}
\]  

(17)

where \( j_k \) denotes the sequence representing the \( m \) sibling keys used in updating parent key, and \( \| \) denotes message concatenation. In addition to the rekeying message, it is necessary to transmit the amount \( m \) of children keys, and the user ID message \( \{j_1, j_2, \ldots, j_m\} \), which specifies the portion of the rekeying message needed by a user in order to determine the new session key.

The transmission of the user ID message in the conventional scheme reveals which sibling keys are still valid. However, it requires that \( \lceil \log_2 m \rceil \) bits represent \( m \) and \( m \lceil \log_2 a \rceil \) represent \( \{j_1, j_2, \ldots, j_m\} \). The total communication overhead of the conventional scheme is thus \((m + 1) \lceil \log_2 a \rceil\) bits.

The polynomial scheme is an example of an anonymous broadcast scheme in that it does not reveal the indices of the valid sibling keys. The polynomial interpolation scheme creates a composite message that does not require any user ID message, but instead requires the broadcast of the seed \( \mu(t) \). The polynomial scheme defines the rekeying message as the output of a function \( \text{PolyInt} \), which returns the coefficients of the interpolating polynomial, thus

\[
\alpha = \text{PolyInt}\left( K, \{\xi_{j_1}, \xi_{j_2}, \ldots, \xi_{j_m}\}, \{K_{j_1}, K_{j_2}, \ldots, K_{j_m}\}, \mu(t) \right)
\]  

(18)

The input to \( \text{PolyInt} \) is the key \( K \) that is to be distributed, the set of valid non-secret ID parameters \( \{\xi_{j_1}, \xi_{j_2}, \ldots, \xi_{j_m}\} \), the set of valid sibling keys \( \{K_{j_1}, K_{j_2}, \ldots, K_{j_m}\} \), and the broadcast seed \( \mu(t) \). Given a valid sibling key and the seed \( \mu(t) \), the new parent key can be determined. On the other hand, an invalid sibling key is unable to determine the new parent key.

If the prime \( p \) used in the polynomial scheme has the same bit length as the output of one of the encryptions \( E_{K_0} \), then the size of the payload for the polynomial scheme will be the same as that of the rekeying message of the conventional scheme.
If $B_\mu$ is the bit length of the broadcast seed; then a measure of comparison between the conventional scheme and the polynomial scheme is the difference
\[(m + 1) \lceil \log_2 a \rceil - B_\mu \]. For a single sibling update of the parent node, this difference might favor the conventional approach. However, the advantage of the polynomial scheme becomes more pronounced when used in a multi-level tree as described above.

III. Operation of the Multicast

A. Key Refreshing

Refreshing the session key is important in secure communication. As a session key is used, more information is released to an adversary, which increases the chance that a SK will be compromised. Therefore, periodic renewal of the session key is required in order to maintain a desired level of content protection. By renewing keying material in a secure manner, the effects of a session key compromise may be localized to a short period of data.

The cryptoperiod associated with a session key is governed by many application specific considerations. First, the value of the data should be examined and the allowable amount of unprotected (compromised) data should be addressed. For example, the broadcast of a sporting event might allow the data to be unprotected for a short period, whereas a video conference between corporate executives would likely have stricter security requirements.

Since the amount of data encrypted using KEKs is usually much smaller than the amount of data encrypted by a session key, it is not necessary to refresh KEKs. Therefore, KEKs from the previous time interval $t-1$ carry over to the next time interval. In order to update the session key $K_{(t-1)}$ to a new session key $K_{(t)}$, the group center generates $K_{(t)}$ and encrypts it using the root KEK $K_\varnothing(t)$. This produces a rekeying message $\alpha_{(t)} = E_{K_\varnothing(t)}(K_{(t)})$, where $E_K(m)$ denotes the encryption of $m$ using the key $K$. This message is transmitted to the users using either the media-independent channel or the media-dependent channel. If the media-independent
channel is used, the multimedia content is then encrypted using $K_{c(t-1)}$ and broadcast. This broadcast may take place on a variety of different transmission media. If the transmission takes place on a packet-based network, the use of reliable transport level protocols such as Reliable Multicast Transport Protocol (RMTTP) or Scalable Reliable Multicast (SRM) is recommended. In the case when embedding is used, the message $\alpha_c(t)$ is embedded in the multimedia data to produce a composite signal $x_c(t)$, which is encrypted using the old session key $K_{c(t-1)}$ and broadcast. Only those with $K_{c(t-1)}$ may access $\alpha_c(t)$, while only those with both $K_{c(t-1)}$ and $K_{c(t)}$ may acquire $K_{c(t)}$.

B. Member Join

In multimedia services, such as pay-per-view and video conferences, the group membership will be dynamic. Members may want to join and depart the service. It is important for the key distribution scheme to be able to add new members to any group in a manner that does not allow new members to have access to previous data. In a pay-per-view system, this amounts to ensuring that members can only watch what they pay for, while in a corporate video conference there might be sensitive material that is not appropriate for new members to know.

Suppose that, during time interval $t-2$, a new user contacts the service desiring to become a group member. If there were $n-1$ users at time $t-2$ then there will be $n$ users at time $t$. During time interval $t-1$, the rekeying information must be distributed to the $n-1$ current members. Observe that both the SK and the root KEK must be renewed in order to prevent the new user from accessing previous rekeying messages and to prevent access to prior content.

The first stage of the key updating procedure requires updating the root KEK from $K_{c(t-1)}$ to $K_{c(t)}$. Since all of the members at time $t-1$ share $K_{c(t-1)}$, the group center may communicate the new KEK $K_{c(t)}$ securely to these members by forming the message $\alpha_x(t) = E_{K_{c(t-1)}}(K_{c(t)})$. The message $\alpha_x(t)$ is transmitted to the users using either the media-dependent or media-independent channel. If $\alpha_x(t)$ is
embedded in the multimedia, a composite signal \( x_e(t) \) is formed, and encrypted using \( K_e(t-1) \) and broadcast to the current members. The steps involved in using data embedding are depicted in Figures 2(a), 2(b), and 2(c). If the media-independent channel is used, the content is encrypted using \( K_e(t-1) \).

Next, the session key is updated to \( K_e(t) \). Since all of the current members have \( K_e(t) \), the session rekeying message \( \alpha_e(t) = E_{K_e(t)}(K_e(t)) \) is formed, and transmitted via either of the approaches. The content is then encrypted using \( K_e(t-1) \) and broadcast.

Finally, if data embedding is used, the key that governs the embedding must be changed to make it more difficult for the new user to extract the rekeying messages from any compromised communications that he might have observed earlier. The GC forms a new embedding key \( K_{emb}(t) \), creates the message \( \alpha_{emb}(t) = K_{emb}(t) \), and embeds this message in the multimedia stream using the previous embedding rule. The composite signal \( x_{emb}(t) \) is encrypted using \( K_e(t-1) \) and broadcast to the current group members.

Meanwhile, during time interval \( t-1 \), the new user completes registration with the service and is given the new keys \( K_e(t), K_d(t), K_{n+1} \) and \( K_{emb}(t) \). This completes the actions required during time interval \( t-1 \), and at the start of time interval \( t \) all of the \( n+1 \) members have the new keying material.

C. Member Departure

To illustrate the issues involved in member departure, consider the case when user \( u_n \) leaves the group at time frame \( t-1 \). Since user \( u_n \) knows \( K_e(t-1) \) and \( K_e(t-1) \) these keys must be renewed. First \( K_e \) is renewed. In a conventional scheme, the GC forms a new key \( K_e(t) \) and encrypts it with the keys \( K_j \) for \( j \neq n \). A single message:

\[
\alpha_e(t) = E_{K_e}(K_e(t)) \parallel E_{K_j}(K_e(t)) \parallel \ldots \parallel E_{K_{n-1}}(K_e(t))
\]  

(19)
is formed and sent to all the users using either the media-independent or media-
dependent channel. The symbol $||$ is used to denote concatenation of bitstreams. If
data embedding is used, then $\alpha_u(t)$ is embedded into the multimedia content to produce a composite signal $x_d(t-1)$. The composite signal is then encrypted using
$K_d(t-1)$ since $x_d(t-1)$ contains multimedia content that we wish to prevent casual
access to. If the media-independent channel is used, the media content is just
encrypted using $K_i(t-1)$.

In the data embedding case, those who have $K_d(t-1)$, which includes both
current group members and departing members, have access to $x_d(t-1)$. These same
entities may access $\alpha_u(t)$ since they have the embedding key $K_{emb}(t-1)$ that governs
how the data was embedded. In either the media-independent or media-dependent
case, only valid users can extract the new KEK $K_c(t)$ by decrypting the segment of
the message corresponding to their key $K_i$.

Next, the session key is updated. The GC forms a new SK $K_d(t)$ and encrypts
using the new KEK $K_c(t)$ to form:

$$\alpha_s(t) = E_{K_c(t)}(K_s(t))$$ (20)

This message is then sent to the users. If the media-independent channel is used, the
content is next encrypted using $K_i(t-1)$. Otherwise, if embedding is used, the
rekeying message $\alpha_u(t)$ is embedded in the media to produce the composite signal
$x_d(t-1)$.

Although user $u_n$ may extract $\alpha_u(t)$, he may not acquire $K_d(t)$.

If data embedding is used, then the embedding key must be updated. At the
start of a new time frame $t$, the group center chooses a new embedding key $K_{emb}(t)
that governs how future messages will be embedded. The message $\alpha_{emb}(t) = K_{emb}(t)$ is
embedded in the media using the old embedding rule governed by $K_{emb}(t-1)$ to form
a composite signal $x_{emb}(t)$ which is encrypted using the new session key $K_d(t)$ and
broadcast. Since user $u_n$ does not have $K_d(t)$, he will not have access to $\alpha_{emb}(t)$. 
During time frame t-1, user \( u_n \) still has access to the multimedia content, but at time frame \( t \), he is no longer able to enjoy the service.

IV. Scalability

When the multicasting group is very large, the key distribution scheme mentioned above has severe computational and communication overhead associated with member departures because the size of \( a_k \) increases linearly with the number of users \( n \). To solve the scalability problem, a tree-based algorithm may be used to update the SK and KEKs (C. K. Wong, et al., "Secure group communication using key graphs," SIGCOMM ‘98; C. K. Wong, et al., University of Texas at Austin, Computer Science Technical report TR 97-23; R. Poovendran, "Key Management for Secure Multicast Communications," Univ. of Maryland. College Park: PhD dissertation. August 1999). The preferred method of the present invention has the ability to handle dynamic groups, and is suitable for: multicast, key management, user/content mobility, and multimedia.

During periodic refreshes, only the session key needs to be updated, and the same protocol as presented in Section VI can be used. The manner in which the scheme operates during additions and deletions of members will now be addressed.

The GC is in charge of keeping track of the group members, and assigning them to positions on the tree. Although it is easiest to have the membership tree be a balanced tree, such is not necessary. For example, a non-balanced tree employing one-way functions may be used in a key management scheme allowing member joins and departures (D. Balenson et al., "Key Management For Large Dynamic Groups: One-Way Function Trees And Amortized Initialization," Internet Draft Report). For the purposes of the present invention, the procedure for adding members to a non-full balanced tree, and removing members from a full balanced tree will be described. If a balanced tree is full, meaning all of the leaf nodes have members associated with them, then a new layer of nodes should be spawned when adding members. Additionally, by following the example of Balenson et al. (D. Balenson et al., "Key Management. For Large Dynamic Groups: One-Way Function
Trees And Amortized Initialization,” Internet Draft Report) one can see how to make an approach handle member joins and departures for non-balanced trees. 

A binary tree is shown in Figure 3(a) and a ternary tree is shown in Figure 3(b), though in the general case, the tree can be an α-degree tree and can have arbitrarily many members. Attached to the tree above the root node is the session key $K_s$. Each node in the tree is assigned a KEK that is indexed by the path leading to itself. The symbol $\varepsilon$ is used to denote empty string, which is the path of the root node to itself. Each user is assigned to a leaf and is given the KEKs of the nodes from the leaf to the root node in addition to the session key. For example, user $u_{111}$ is assigned keys $K_{111}, K_{11}, K_1, K_o, and K_s$. All of the keys, with the possible exception of the leaf keys, may vary with time to reflect the changing dynamics of the group membership.

Consider the trees shown in Figures 3(a) and 3(b), each node in the tree is assigned a key encoding key (KEK) which is indexed by the path leading to itself. Each user is assigned to a leaf and is given the KEKs of the nodes from the leaf to the root node.

A. Member Join

Consider the binary tree depicted in Figure 3(a), which has 7 members $u_{000}$ through $u_{110}$. If user $u_{111}$ would like to join the group, the keys on the path from his leaf node to the tree's root as well as the SK, must be changed in order to prevent access to previous communications. Thus, new $K_s(t), K_o(t), K_{11}(t)$ and $K_o(t)$ must be generated by the GC. The key encrypting keys can be updated from top to bottom by using $K_o(t-1)$ to encrypt $K_s(t)$, and $K_o(t-1)$ to encrypt $K_o(t)$, and $K_{11}(t-1)$ to encrypt $K_o(t)$. Thus, all users can acquire the new root KEK, while only members $u_{100}, u_{101}, and u_{110}$ can acquire $K_o(t)$. After updating the KEKs, the session key is updated by encrypting with the new root KEK $K_o(t)$. The messages that are formed may be transmitted using either a media-independent or media-dependent channel. Meanwhile, the new user is given the new keys directly from the GC during registration.
B. Member Departure

When a member leaves the group, multiple keys become invalidated because that user shares these keys with other users. In the tree hierarchy (Figures 3(a) and 3(b)), each user \( u_i \) is given a set of keys \( S_i = \{ K_{s}, K_{f(i)}, l = L, \ldots, 0 \} \) where \( K_{f(i)} \) is the KEK indexed by \( t_i(t) \), which describes the path from the leaf to root of the tree for that user. For example, in Figure 3(b), user \( u_{000} \) (000 in ternary) has a set of keys \( S_{000} = \{ K_{s}, K_{0}, K_{00}, K_{000} \} \). If user \( j = (d_0d_1 \ldots d_{L-1})_0 \) leaves the group, the set of keys that need to be changed is denoted as \( S_j(j) = S_j - \{ K_j \} \). For example in Figure 3(b), if \( u_{222} \) left, \( S_6(222) = \{ K_6, K_{2}, K_{22} \} \). There are basically two ways to update the bad keys in \( S_j(j) \), one is from top to bottom, i.e., \( K_s \rightarrow K_{t_i(i)} \rightarrow \ldots \rightarrow K_{t_j(j)} \). Another way is from bottom to top, i.e., \( K_{t_j(j)} \rightarrow \ldots \rightarrow K_{t_i(i)} \rightarrow K_s \).

Thus, for example, in Figure 3(a), user \( u_{111} \) shares \( K_{11} \) with user \( u_{110} \). Thus, if user \( u_{111} \) departs the multicast group, the key encrypting keys \( K_{11}, K_j, \) and \( K_s \) become invalidated. These keys must be updated. Observe that \( K_{11} \) does not need to be updated since it is a private key and is not shared with any other users.

Either of two basic approaches may be followed in updating the keys during a member departure: update the keys from the root node to leaf nodes, or from leaf nodes to root node. In the first approach, the top-down approach, when user \( u_{111} \) departs, the keys are updated in the order \( K_s, K_j, \) and \( K_{11} \). The second approach, the bottom-up approach, updates the keys in the order \( K_{11}, K_j, \) and \( K_s \). After updating the key encrypting keys, the root KEK \( K_s(t) \) can be used to encrypt the new session key \( K_n(t) \) and a single message may be broadcast to all members.

This aspect of the present invention is illustrated below with respect to the residue-based method, however, it will be understood that either the residue-based method or the polynomial interpolation method may be employed, and is included in the present invention. Assume user \( u_j \) left the group, the communications overhead of these two methods would differ:
Top $\rightarrow$ Bottom: The keys that need to be changed are
$K_e \rightarrow K_{e,u(j)} \rightarrow \ldots \rightarrow K_{e(j)}$. In order to update the root KEK, the minimal set of KEKs that cover the users that need the new root key must be determined. If user $j$ departs, the minimal set of KEKs that cover the users that need an updated key $K_w$ may be designated as $S^w_j$. For example in Figure 3(b), if user $u_{222}$ left, then the KEKs that would cover the users needing $K_e$ would be $S^e_{222} = \{K_{b}, K_{1}, K_{20}, K_{21}, K_{220}, K_{221}\}$. The rekeying message $\alpha_e(t)$, which is used to distribute the new root key, is defined as:

$$\alpha_e(t) = K_e(t) + \prod_{K_{i} \in S^e_j} f(K_i, \mu(t))$$

A legitimate user $u_i$ can obtain a new key $K_e$ by performing a modulo computation involving the KEK from $S^e_j$ that he knows.

Other KEKs can be changed similarly. To change a KEK $K_w$, for $K_w \in S_b(j)$ the minimal covering set $S^w_j$ is determined. and the message $\alpha_w(t)$ is calculated by

$$\alpha_w(t) = K_w(t) + \prod_{K_{i} \in S^w_j} f(K_i, \mu(t))$$

In Figure 3(a), to update $K_e(t)$ after user $u_{222}$ leaves the group, $S^e_{222} = \{K_{20}, K_{21}, K_{220}, K_{221}\}$. It can be verified that the total number of multiplications used to change all of the compromised keys is

$$C_w = \sum_{i=1}^L i(a-1) \left(\frac{\log_a n (\log_a n + 1)}{2}\right)$$

where $L = \lceil \log_a n \rceil$.

Bottom $\rightarrow$ Top: The keys that need to be changed are
$K_j \rightarrow \ldots \rightarrow K_{j+1} \rightarrow K_x$, where $j$ represents $u(j)$. The key $K_{j+1}$ of the parent node of
user $j$ is changed first, then key $K_{j2}$ is changed next, and $K_e$ is changed finally. If KEK $K_j, t = 1, \ldots, L - 2$ is already changed to $K_j(t)$, then to change KEK $K_{j1}$, denote $S_{K_{j1}}^c(t)$, the set of keys of the children nodes of key $K_{j1}$ and the rekeying message for key $K_{j1}(t)$ is

$$\alpha(K_{j1}) = K_{j1}(t) + \prod_{K_i \in S_{K_{j1}}^c(t)} f(K_i, \mu(t))$$  \hspace{1cm} (24)

It can be verified that $a - 1$ leaf keys are used to change the KEK $K_j$ and $a$ KEKs are used to change each of the other KEKs and session key. Then the total number of multiplications used to change all the bad keys is:

$$C_{br} = aL - 1 = a \log_a N - 1$$  \hspace{1cm} (25)

The computation complexity of bottom-top is much lower than top-bottom method. But the top-bottom method changes the SK faster than does the bottom-top method.

Focusing on how to update these keys using the top-down approach in conjunction with the new message form when user $u_{111}$ departs from the binary tree shown in Figure 3(a), first, a random seed $\mu(t)$ is broadcast to all members. It will be available to user $u_{111}$. Next, the root KEK $K_d(t-1)$ will be updated. In order to do this, the message:

$$\alpha_e(t) = K_e(t) + f(K_o(t-1), \mu(t)) f(K_{10}(t-1), \mu(t)) f(K_{110}, \mu(t))$$  \hspace{1cm} (26)

is formed and broadcast. Next, $K_{j(t-1)}$ is updated by forming the message:

$$\alpha_i(t) = K_i(t) + f(K_{10}(t-1), \mu(t)) f(K_{110}(t-1), \mu(t))$$  \hspace{1cm} (27)

and broadcasting. The last KEK to update is $K_{11}(t-1)$. This can be done by sending the message:
\[ \alpha_{11}(t) = K_{11}(t) + f(K_{110}(t-1), \mu(t)). \] (28)

Upon updating the KEKs, the session key may then be updated. To do this, the root KEK is used to encrypt \( K_{0}(t) \) and the resulting message is broadcast.

In order to update the keys from a bottom-up approach, the random seed is broadcast, and then \( K_{11}(t-1) \) is updated via:

\[ \alpha_{11}(t) = K_{11}(t) + \prod_{j=0}^{0} f(K_{111}(t-1), \mu(t)) \] (29)

The next key that is updated is \( K_j(t-1) \). Since the two users beneath \( K_j \) share a common key that is not invalidated by the departure of member \( u_{111} \), the communication and computation may be reduced by using this key to update \( K_j \).

The resulting message:

\[ \alpha_i(t) = K_i(t) + \prod_{j=0}^{1} f(K_{ij}(t), \mu(t)) \] (30)

is broadcast. Since \( K_{110}(t-1) \) is still valid, \( K_{110}(t) \) has been implicitly updated to equal \( K_{110}(t-1) \). To update \( K_{0}(t-1) \), the new key \( K_i(t) \) as well as the old key \( K_0(t) = K_0(t-1) \) may be used, forming the message:

\[ \alpha_z(t) = K_z(t) + \prod_{j=0}^{1} f(K_j(t), \mu(t)). \] (31)

Finally, the session key is updated by encrypting the new session key \( K_0(t) \) using the new root KEK \( K_0(t) \), and broadcasting the message:

\[ \alpha_z(t) = E_{K_0(t)}(K_z(t)). \] (32)

The amount of multiplications needed to update all of the KEKs using the top-down approach and the bottom-up approach will differ. Assume that \( n \) users exist and that keys have been assigned to each of these users using an \( a \)-ary tree. If the tree is a full, balanced tree with \( L = \log_a n \) levels, then the amount of
multiplications needed to update the KEKs during a member departure using a top-down approach is:

\[ C_{td} = \sum_{i=1}^{L} (a-1) \log_a n (\log_a n + 1) \]

(33)

Similarly, the amount of multiplications needed to update the KEKs using a bottom-up approach is:

\[ C_{bu} = aL - 1 = a \log_a n - 1 \]

(34)

The amount of communication needed for each of these schemes is directly related to the amount of multiplications performed. If each KEK is B bits long, and a rekeying message requires M multiplications, then the message size will be M(B + 1) bits. Therefore, the bottom-up approach to renewing the keys requires less computation and communication. However, if the SK needs to be updated sooner, one may wish to use a top-down approach since it allows one to update the root KEK first, the session key next, and finally the remaining KEKs.

If the key distribution is taking place via an media-dependent channel, then the embedding key must be updated also. In this case, the new embedding key \( K_{emb}(t) \) can be embedded in the media stream using the previous embedding key \( K_{emb}(t-1) \). The composite signal will be encrypted using the new session key \( K_s(t) \). Only those with both \( K_s(t) \) and \( K_{emb}(t-1) \) will be able to acquire the new embedding key. The new user, although he may receive the video, does not need to extract the new embedding key since it will have been given to him, in addition to information governing when to start using it, by the GC.

V. Implementation Issues

A. Scalable Protocol of a Preferred Key Distribution Scheme

In Section II.B.2.(c), the basic scheme for distributing keys during member departures was described. The basic residue-based or polynomial interpolation scheme had linear communication requirements during member departures. The
scalability of a preferred protocol, and its ability to provide renewal of security levels, handle membership changes, provide a mechanism for reinserting valid members, and allow for the transferal of access rights are now considered.

In order to achieve improved scalability, a tree-based key hierarchy as depicted in Figure 3(a), and Section IV is employed. As previously stated, in general, the tree can be an α-degree tree. Attached to the tree above the root node is the session key $K_r$. Each node of the tree is assigned a KEK, which is indexed by the path leading to itself. Additionally, in the polynomial scheme each node has a non-secret ID variable $z_n$, which is used as a non-secret parameter for the PolyInt function. The symbol $e$ is used to denote the root node. Each user is assigned to a leaf of the tree and is given the KEKs of the nodes from the leaf to the root node. Additionally, all users share the session key $K_r$. Thus, user $u_{111}$ is assigned keys $K_{111}, K_{11}, K_{1}, K_e$, and $K_r$.

In the protocol that follows, the GC transmits messages to the users via a broadcast channel. It is assumed that each user has an upstream channel with minimal bandwidth that is available to convey messages to the GC, such as informing the GC of the intent to depart the service.

This aspect of the present invention is illustrated below with respect to the polynomial interpolation method, however, it will be understood that either the residue-based method or the polynomial interpolation method may be employed, and is included in the present invention. The messages that the GC broadcasts to the users must have a standardized structure that is known to all receivers. There are two basic message formats as depicted in Figures 4(a) and (b). The first (Figure 4(a)) contains three components while the second (Figure 4(b)) has five components. The function $B()$ is used to denote the bit length of its operand, thus $B(\sigma)$ is the amount of bits needed to represent $\sigma$. The variable Operation ID flags the user which protocol primitive is about to be performed. Only five primitive operations are used, and one may therefore represent Operation ID using a 3-bit string. Table 1 below maps the bit strings and the primitives:
<table>
<thead>
<tr>
<th>Bit ID</th>
<th>Primitive</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>Primative-1</td>
</tr>
<tr>
<td>001</td>
<td>Primative-2</td>
</tr>
<tr>
<td>010</td>
<td>Primative-3</td>
</tr>
<tr>
<td>011</td>
<td>Primative-4</td>
</tr>
<tr>
<td>100</td>
<td>Primative-5</td>
</tr>
</tbody>
</table>

It is assumed that the tree has degree $a$ and $L$ levels. The amount of multicast group members $n$ is limited by the amount of leaf nodes on the tree. Thus $n \leq a^L$.

**B. Basic Protocol Primitives**

The following five basic operations are involved in building a system that allows for the update and renewal of the key hierarchy:

1. **Primitive-1 (Update SK):** This basic operation uses the current root KEK $K_e$, to update the session key via the rekeying message:

   $$\alpha = E_{K_e(t)}[K_s(t)]$$  \hfill (35)

   The message format is depicted in Figure 4(a). We assume that the maximum size that $\alpha$ can be is 256 bits, and therefore 8 bits are needed to represent $B(\alpha)$. This choice of bit length for $\alpha$ would allow for the use of encryption algorithms with a key size of up to 256 bits.

2. **Primitive-2 (Transmit Seed):** The broadcast seed is used in the polynomial scheme to provide protection of secret information. Additionally, it plays a role in reducing the communication overhead associated with flagging the users which part of the message is intended for them. The broadcast of the seed $\mu(t)$ does not require encryption to protect it. The message format for the transmission of the broadcast seed is depicted in Figure 4(a). Here $\alpha = \mu(t)$, and $B(\alpha)$ is the amount of bits needed to represent $\mu(t)$. Again, it is assumed that the maximum size of $\alpha$ is 256 bits, and that 8 bits are used to represent $B(\alpha)$. 
3. **Primitive-3 (Self Update):** It is often necessary for a node, indexed by the $a$-ary symbol $\sigma$, to have its associated key updated using the key at the previous time instant. Thus, one would go from $K_\sigma(t-1)$ to $K_\sigma(t)$ by the following message

$$\alpha = E_{K_\sigma(t-1)}[K_\sigma(t)]$$  \hspace{1cm} (36)

In this case, it is desirable to flag the receivers so advise them of the node being updated. This requires the transmission of the $a$-ary representation of the node, as well as the amount of bits needed to represent the node. This is depicted in **Figure 4(b)** by the $B(\alpha)$ and $\sigma$ components of the message. The rest of the message contains the bit length of the message $\alpha$ and the actual rekeying message $\alpha$. Since the maximum depth of the tree that needs to be represented is $L-1$ and the tree is an $a$ degree tree, the maximum amount of bits needed to represent $\sigma$ is

$$\lceil \log_2 a \rceil (L-1) + 1,$$

where the addition of 1 bit was included to account for the need to represent the empty string $\epsilon$ as a possible choice for $\epsilon$. In order to represent $B(\sigma)$, we use $\left\lceil \log_2 \left( \lceil \log_2 a \rceil (L-1) + 1 \right) \right\rceil$ bits. The maximum bit length for $\alpha$ is 256 bits, and 8 bits are used to represent $B(\alpha)$.

4. **Primitive-4 (Update Parent):** It is also desirable for the children nodes to update the key of their parent nodes. If $\sigma$ is the symbol representing the parent node to be updated, then the following message is used:

$$\alpha = \text{PolyInt}\left(K_\sigma(t),\left\{K_{\text{Child}(\sigma)}(t)\right\},\left\{\mu(t)\right\}\right)$$  \hspace{1cm} (37)

The function $\text{Child}(\sigma)$ is defined to denote the set of valid children nodes of $\sigma$. For example, for a binary tree and $\sigma = 00$, and both children nodes are valid, then $\text{Child}(\sigma) = \{000, 001\}$. Thus, the message $\alpha$ uses the keys of valid children nodes to update $K_\sigma(t)$. Observe that this message requires that $\mu(t)$ has already been broadcast using Primitive-2, or that the choice of $\mu(t)$ is implicitly known. The message form is depicted in **Figure 4(b)**, where again the bit length of $\sigma$ and the actual symbol $\sigma$ to the recipients are transferred, followed by the bit length of $\alpha$ and the rekeying message $\alpha$. The same bit allocation is used for $\sigma$ and $B(\sigma)$ as in Primitive-3.
However, the maximum length for $\alpha$ is $aB_{KEK}$, and therefore $\lceil \log_2 aB_{KEK} \rceil$ bits are needed to represent $B(\alpha)$.

5. **Primitive-5 (Reaffirming Parent):** In some operations, it is useful to have a sibling node reaffirm the value of a parent node's key. The function $Par(\sigma)$ is defined to denote the symbol corresponding to the parent of the node indexed by $\sigma$. To reaffirm the value of a parent node's key, the following message is transmitted:

$$\alpha = E_{K_c(\sigma)} \left[ K_{Par(\sigma)} (t) \right]$$  \hspace{1cm} (38)

The message form is depicted in Figure 4(b), and follows the same structure as used in Primitive-3.

C. **Advanced Protocol Operations of a Preferred Key Distribution Scheme**

More advanced protocol operations can be constructed using the primitive operations described above. In particular, they may be used to facilitate the operations of addition to the membership, deletion of a user from the membership, reinsertion of a member into the system, and the transferal of access rights from one user to a new user.

Additionally, the primitive operations can be used to perform periodic renewal of keying material. Primitive-1 provides a method for performing periodic refreshing of the session key. Refreshing the session key is important in secure communication. As a session key is used, more information is released to an adversary, which increases the chance that an SK will be compromised. Periodic renewal of the session key is thus desirable in order to maintain a desired level of content protection, and can be used to localize the effects of a session key compromise to a short period of data. Since the amount of data encrypted using KEKs is usually much smaller than the amount of data encrypted by a session key, it is not necessary to refresh KEKs as often. However, the periodic renewal of a KEK can be performed using Primitive-3.
1. **Member Join:** It is desirable to be able to add new members to any group in a manner that does not, allow new members to have access to previous data. Suppose that, a new user contacts the service desiring to become a group member. The new client sends the GC a message detailing the client’s credentials, such as identity information, billing information, and public key parameters that the GC may use to communicate with the new client. Mutual authentication between the new client and the GC should be performed. A public key infrastructure, such as X.509 certificates (ITU-T Recommendation X.509 (1997), "The directory: Authentication framework," 1997) may be used for this purpose. Upon verification of the new user's information, the GC assigns the client to an empty leaf of the key tree. For simplicity of presentation, assume that the tree has empty slots. If the tree is already full, then the user may either be turned away, or an additional layer must be added or augmented to the tree using a separate operation. The GC then issues the new client his keys via a communication separate from the communications sent to the current group members, as well as informing the new user the time at which those keys will become valid.

Meanwhile, the GC updates the current members of the multicast group. Suppose that the GC plans to insert the new member into the leaf node indexed by the symbol $\omega$. Then the SK as well as the KEKs on the path from the parent node of $\omega$ to the root node $\epsilon$ must be renewed. The following algorithm describes how this procedure can be accomplished using the protocol primitives. The notation $\text{Par}^j(\omega)$ to denote the parent function applied $j$ times to $\omega$. Thus, $\text{Par}^L(\omega)$ is the grandparent of $\omega$.

\[
\text{for } j = 1 : L \text{ do} \\
\quad \sigma = \text{Par}^j(\omega) \\
\quad \text{Update } K_{\sigma(t-1)} \rightarrow K_{\sigma(t)} \text{ using Primitive-3;} \\
\text{endfor} \\
\text{Update SK using Primitive-1;} 
\]

2. **Member Departure:** Members will also wish to depart the service, and must be prevented from accessing future communication after their departures.
Assume that user $u_w$ contacts the GC wishing to depart the service. Upon authenticating the user’s identity, a procedure that the GC may enact to remove member $u_w$ and update the keys of the remaining members is

\[
\text{Generate random } \mu(t) ;
\]
\[
\text{Broadcast } \mu(t) \text{ using Primitive-2} ;
\]
\[
\text{for } j=1 : L \text{ do}
\]
\[
\sigma = \text{Par}^i(\omega) ;
\]
\[
\text{Determine valid children of } \sigma: \text{Child}(\sigma) ;
\]
\[
\text{Update } K_{\sigma}(t-1) \rightarrow K_{\sigma}(t) \text{ using Primitive-4} ;
\]
\[
\text{endfor}
\]
\[
\text{Update SK using Primitive-1} ;
\]

3. **Member Reinsertion**: It might often occur that a valid member, denoted by index $\omega$, misses the rekeying messages needed to update the key hierarchy. The client must notify the GC that he missed rekeying messages using an upstream (client to server) channel. Upon verification of the user’s identity, the GC performs the member reinsertion operation, which sends the new user the specific keys he needs to be able to resume the service.

If the service provider has a downstream (server to client) channel available to communicate with the user, then service provider may use this channel to send the needed keys by encrypting them with the user’s personal key $K_{\omega}$.

In many scenarios, however, after the initial contact with the service provider, the client has a low-bandwidth channel for upstream communication, and only the broadcast channel available for downstream communication. In these cases, although only a single user needs the rekeying messages, the rekeying messages must be multicast. Since this user has a valid private key $K_{\omega}$, the GC can start with this key to provide the user the current key associated with the parent node of $\omega$. Thus, $K_{\text{Par}(\omega)}(t)$ is provided to the user. One can then proceed up the tree, using the sibling key to convey the current status of the parent key. A procedure for this operation is as follows:

\[
\text{for } j=1 : L \text{ do}
\]
\[
\sigma = \text{Par}^i(\omega) ;
\]
\[
\text{Convey parent key } K_{\sigma}(t) \text{ to siblings using Primitive-5} ;
\]
Convey current SK using Primitive-1;

An added bonus of using the sibling key to convey the current status of the parent key is that other users may observe these rekeying messages to reaffirm the validity of some of their keys.

4. **Transferal of Rights:** Suppose that user $u_\omega$ wishes to give his rights to another user who is not currently a member. The new user will be denoted by $u_{\omega_B}$ to indicate that he will take over the keys on the path from $\omega$ to the root node. For the purpose of calculating parent and sibling relationships, $\omega$ and $\omega_B$ are identical, thus $\text{Par}(\omega) = \text{Par}(\omega_B)$.

In order to transfer access rights, both users must contact the GC, who performs an authentication procedure to verify that the transferal is legitimate. Then, using a secure channel, the GC gives to user $u_{\omega_B}$ its own personal key $K_{\omega_B}$.

One method for creating a secure channel is to use public key cryptography. $K_{\omega_B}$ replaces $K_\omega$ on the key tree. All of the keys that belonged to $u_\omega$ must be changed to prevent $u_\omega$ from accessing content that he has given up the right to access. A procedure for transferring access rights is as follows:

```
Generate random $\mu(t)$;
Broadcast $\mu(t)$ using Primitive-2;
for $j = 1 : L$ do
    $\sigma = \text{Par}(\omega_B)$;
    Determine valid children of: $\text{Child}(\sigma)$;
    Update $K_{\sigma(t-1)} \rightarrow K_{\sigma(t)}$ using Primitive-4;
endfor
Update SK using Primitive-1;
```

Observe that the algorithm for transferring rights is nearly identical with the algorithm for removing a member from a group. The difference lies in the fact that user $u_{\omega_B}$ is considered a valid user, and hence is a valid child of its parent.

The procedure for user $u_\omega$ to reclaim his access privileges is similar. This time, only user $u_\omega$ is required to contact the GC requesting that he regain his access.
privileges. The GC performs an authentication procedure to guarantee that the identity of \( u_w \) is truthful, and then replaces \( K_{\omega_B} \) with \( K_{\omega_w} \). The KEKs and SK are changed according to the above algorithm, with \( \omega \) replacing \( \omega_B \).

D. Architecture Considerations

In preferred embodiments, the key distribution schemes of the present invention operate in a manner that reflects the following architectural considerations:

1. Optimization of Tree Degree for Communication

The amount of communication that a rekeying protocol requires affects the speed at which the rekeying scheme can handle membership changes. It is therefore desirable to minimize the size of the communication used by the key management scheme. In particular, since the two most important operations performed by a multicast key management protocol are membership joins and membership departures, it is appropriate to focus on optimizing the tree degree in terms of these two operations.

To illustrate this, a “worst-case” analysis of the communication requirements for member join and member departure operations is presented. It is observed that member join and member departure operations lead to conflicting optimality criteria. Since a real system will have to cope with both member joins and member departures, a method is provided that jointly considers the departure and join operations, and presents optimization results when both member join and departure operations are equally weighted.

The degree of the tree is denoted by \( \alpha \), and the number of levels in the tree by \( L \). \( B_{SK} \) shall denote the bit length of the session key, \( B_{KEK} \) shall denote the bit length of the key encrypting keys, and \( B_\mu \) shall denote the bit length of the broadcast seed \( \mu(t) \).
2. Worst-Case Analysis

It is easy to see that, for a given tree, the scenario that produces the most communication for the member join operation occurs when one node on each level from the root to level L-1 must be updated. In this case, all of the KEKs on the path from one user to the root must be refreshed. The amount of communication needed to update the tree for this worst-case scenario is calculated below.

The member join operation consists of two types of operations: updating the KEKs, and updating the SK. In order to update the KEKs, Primitive-3 is used $L$ times. Each step of the loop must send the quintuple (operation ID, bit length of update node $B(\sigma)$, node ID $\sigma$, bit length of the update message $B(\alpha)$, update message $\alpha$). The symbol $\sigma$ starts near the bottom of the tree, and through application of the Parent function moves toward the root of the tree.

In order to represent the symbol $\sigma$ during the $j$th iteration of the loop, one needs to convert from base $a$ to base 2 and hence $B(\sigma) = \left\lceil \log_2 a \right\rceil (L - j) + 1$ bits.

In addition, $B(\sigma)$ must be sent, which requires $\left\lceil \log_2 \left( \left\lceil \log_2 a \right\rceil (L-1) + 1 \right) \right\rceil$ bits. Here the addition of 1 was to allow for the need to represent the empty string $\varepsilon$ as a possible choice for $\sigma$. Similarly, in each stage of the loop the rekeying message $\alpha$ has bit length $B(\alpha) = B_{KEK}$ and since the maximum key length has been fixed to be 256 bits, 8 bits are required to represent $B(\alpha)$. The update to the session key requires sending the ID flag, $B(\alpha)$ and $\alpha$. Therefore, the amount of bits needed to update the session key is $3 + 8 + B_{SK}$. The total amount of bits needed to update the key tree during a member join is

$$C_{M} = \sum_{j=1}^{L} \left( 3 + \left\lceil \log_2 \left( \left\lceil \log_2 a \right\rceil (L - 1) \right) \right\rceil + \left\lceil \log_2 a \right\rceil (j - 1) + 1 + 8 + B_{KEK} \right) + 3 + 8 + B_{SK}$$

$$= 8 + 3(L + 1) + L \left( 9 + B_{KEK} + \left\lceil \log_2 \left( \left\lceil \log_2 a \right\rceil (L - 1) + 1 \right) \right\rceil \right) + B_{SK} + \frac{L(L-1)}{2} \left\lceil \log_2 a \right\rceil$$
The amount of communication needed in the member departure case can be similarly calculated. The main difference between member join and member departure is that there are three operations: the broadcasting of $\mu(t)$, the updating of the KEKs, and the updating of the SK. The most communication occurs when $a-1$ nodes on level $L$ must be used to update the key on level $L-1$, and $a$ nodes are used to refresh each of the remaining KEKs on the path from the departing member to the root node. In this case, the communication is

$$C_{MD} = 16 + 3(L + 2) + B_{SK} + B_{\mu} + (L-1)aB_{KEK}$$

$$+ (L)\lceil \log_2 aB_{KEK} \rceil + (a-1)B_{KEK} + L$$

$$+ L\lceil \log_2 \left(\lceil \log_2 a \rceil (L-1) + 1 \right) \rceil + \frac{L(L-1)}{2}\lceil \log_2 a \rceil$$

The worst case amount of communication required to update an $a$-degree key tree was calculated as a function of the number of users $n$ with the amount of tree levels set to $L = \lceil \log_2 n \rceil$. In the calculations, $B_{SK}$ was set to equal $B_{KEK} = B_{\mu} = 64$ bits. 64 bits was selected as the key size since such a key length can provide strong levels of security when used with some ciphers, such as RC5. The amount of communication required for different choices of the degree of the tree $a$ during a member join is depicted in Figure 5(a). For each choice of $a$ the curves are monotonic, and the curves exhibit a stair-step behavior due to the ceiling operators in the formula. Figure 5(a) shows the general trend that less communication is required during member join operations as the degree of the tree increases.

Conversely, Figure 5(b) shows the amount of communication needed during the worst case of a member departure operation. In this case, the larger tree degrees are definitely not advantageous. It is also evident that a binary tree is not optimal when considering member departure. In fact, the values of $a = 3$ and $a = 4$ appear to be the best choice, with optimal choice fluctuating depending on $n$. 

In some application scenarios, the key tree might start out relatively empty, and the amount of member join operations become greater than the amount of member departure operations. In this case, the membership grows towards the tree capacity, and the communication required for the member join operation is more critical than the communication for member departure. On the other hand, some scenarios might start out with a nearly full key tree, and the member departure operation would outweigh the member join operation.

It is therefore desirable to have a communication measure that would run the gamut between the extremes of “only member join communication,” and “only member departure communication.” This can be accomplished by considering the convex combination of $C_{MD}$ and $C_{MJ}$. Let $\lambda$ denote the probability of a member departure operation, and assume that $1-\lambda$ is the probability of a member join operation, then the combined communication measure $C_C$, given by

$$C_C = \lambda C_{MD} + (1-\lambda) C_{MJ}$$

weights the member departure and member join operations according to their likelihood. For example, when $\lambda = 0$ the emphasis is entirely placed on the member join operation, while $\lambda = 1$ corresponds to when the emphasis entirely placed on the member departure operation. The case of $\lambda = 0.5$ corresponds to equal emphasis on the two operations, which is depicted in Figure 6. From Figure 6, it can be seen that the choice of $a = 4$ stands out as the best choice for $n > 10000$ when equally weighting the member join and member departure operation.

4. Binomial Occupancy Model

Since it is very difficult to calculate the amount of communication needed during membership changes when a specific amount of users $n$ are placed on the tree, a stochastic model has been devised that allows one to study the behavior of the system when there are varying amounts of occupancy. The model assumes that the
leaf nodes of the $\alpha$-degree key tree with $L$ levels are occupied according to i.i.d. Bernoulli trials with a probability of occupancy $q_L$. This implies that the occupancy $n$ is modeled according to a binomial distribution with mean occupancy $q_L^\alpha$ and variance $q_L(1-q_L)^\alpha$. Hence, when $q_L$ is higher, the tree is closer to being at maximum occupancy.

The average amount of communication required for member join when the probability of a node being occupied is $q_L$. Let $a$ denote the $\alpha$-ary representation of the joining member. The siblings of $\sigma$ are denoted by $\tau_1, \tau_2, \ldots, \tau_{a-1}$. The random variable $Z_{L-1}$ is defined as:

$$Z_{L-1} = \begin{cases} 0 & \text{if any } \tau_k \text{ is occupied} \\ 1 & \text{if no } \tau_k \text{ are occupied} \end{cases}$$

Hence, $P(Z_{L-1} = 1) = 1 - (1-q_L)^{a-1}$, and the expected value of $Z_{L-1}$ is given by $E(Z_{L-1}) = 1 - (1-q_L)^{a-1}$.

A similar procedure may be performed for the other levels. We denote the $j$-siblings as those nodes $\tau$, such that $Par^j(\tau) = Par^j(\sigma)$. For level $L-j$, the random variable $Z_{L-j}$ is defined as:

$$Z_{L-j} = \begin{cases} 0 & \text{if any } j\text{-sibling node } \sigma \text{ is occupied} \\ 1 & \text{if no } j\text{-sibling nodes } \sigma \text{ are occupied} \end{cases}$$

In this case, $P(Z_{L-j} = 1) = 1 - (1-q_L)^{d-1}$, and the expected value of $Z_{L-j}$ is $E(Z_{L-j}) = 1 - (1-q_L)^{d-1}$.

The average communication requirements for member join is then given by:

$$\overline{C_{M_j}} = \left( \sum_{j=1}^{L} E(Z_{L-j}) \left[ 3 + \left\lceil \log_2 \alpha \right\rceil (L-1)+1 \right] + \left\lceil \log_2 \alpha \right\rceil (L-j)+1+8 \right) + 3 + 8 + B_{sk}$$
\[ = \left( \sum_{j=1}^{L-1} \left( 1 - (1 - q_L)^{a - 1 + j} \right) \left( 12 + \left\lceil \log_2 \left( \left\lceil \log_2 a \right\rceil \right) \right\rceil (L - 1) + 1 \right) \right) + 11 + B_{\text{SK}} \]

The model can be applied to calculate the average amount of communication needed during member departure. For this purpose, the departing member is assumed to be indexed by the \( a \)-ary symbol \( \sigma \). The siblings of \( \sigma \) are labeled by \( \tau_1, \tau_2, \ldots, \tau_{a-1} \). The random variable is \( X_k \) is denoted by
\[
X_k = \begin{cases} 
0 & \text{if } \tau_k \text{ is occupied} \\
1 & \text{if } \tau_k \text{ is not occupied}
\end{cases}
\]

Let \( Y_L = \sum_{k=1}^{a-1} X_k \), which is the random variable corresponding to the amount of occupied sibling nodes of \( \sigma \) at level \( L \). The probability that \( i \) sibling leaves at level \( L \) are occupied is given by:
\[
P(Y_L = i) = \binom{a-1}{i} q_L^i (1 - q_L)^{a-1-i}
\]

\( Y_L \) is thus a binomial random variable with expected value \( E(Y_L) = (a-1)q_L \).
Hence, the average number of nodes to be updated at level \( L \) is \( (a-1)q_L \).

At level \( L - 1 \), the parent node of the departing member will automatically be used in updating the next higher level. Since the probability of a node at level \( L \) being occupied is \( q_L \), the probability that a node on level \( L - 1 \), other than \( \text{Par}(\sigma) \), being occupied is
\[
q_{L-1} = 1 - (1 - q_L)^a
\]
The siblings of $\text{Par}(\sigma)$ may now be denoted by $\tau_1, \tau_2, \ldots, \tau_{a-1}$, and the random variable $X_k$ is denoted by:

$$X_k = \begin{cases} 0 & \text{if } \tau_k \text{ is occupied} \\ 1 & \text{if } \tau_k \text{ is not occupied} \end{cases}$$

By defining the random variable $Y_{L-1}$ to be the amount of sibling nodes of $\text{Par}(\sigma)$ that are occupied, $E(Y_{L-1}) = (a-1)q_{L-1}$. Since $\text{Par}(\sigma)$ is to be included in the updating, a 1 is added. Thus, the expected number of nodes on level $L-1$ that must be updated is $1 + (a-1)q_{L-1}$. Similarly, this calculation may be performed for level $j$, where $q_j = 1-(1-q_L + 1)^{a}$, and the expected number of nodes on level $j$ to be updated is $1 + (a-1)q_j$.

In order to calculate the average amount of communication for the member departure operations, the expected amount of communication associated with the overhead and the payload of the message are both considered. The average communication for the overhead includes the amount of communication needed to send the operation id, the node id, and the bit length of the update message. This calculation can be done using the expected value of $Z_{L-j}$. The average communication for the payload is calculated using the expected number of nodes on level $j$ to be updated.

The average amount of communication for $n$ users on an $a$-degree tree with $L$ levels is therefore given by

$$\overline{C_{MD}} = 3 + 8 + B_{\mu} + 3 + 8 + B_{gK}$$

$$+ \left( \sum_{j=1}^{L} (1-(1-q_L)^{a-j}) \left( 3 + \left[ \log_2 \left( \log_2 a \right) \left( L - 1 \right) + 1 \right] \right) \right.$$

$$+ \left[ \log_2 a \right] \left( L - j \right) + 1 + \left[ \log_2 aB_{\text{KX}} \right] \right)$$
\[ + \ell \sum_{j=1}^{a-1} (1+(a-1)q_j) E_{KEK}(K_{j+1}) \]

The mean message size for member join and member departure operations as parameterized by \( q \) when the tree degree is \( a = 4 \) and there are 6, 8 and 10 levels was calculated. The key sizes were chosen to be \( B_{SK} = B_{KEK} = B_{\mu} = 64 \) bits. In Figures 7(a) and (b), the mean communication is indicated as a function of \( q \) (Figure 7(a) shows member join; Figure 7(b) shows member departure). One can see that the expected communication rapidly increases as the probability \( q \) becomes slightly greater than 0. In the member join operation, the communication levels off to a flat plateau as the probability of occupancy increases. For the member departure operation, the mean communication also increases rapidly for \( q < 0.1 \), but then grows less dramatically for higher \( q \). From these two curves, one can infer that a key tree that is roughly half occupied does not have considerably different communication requirements than the worst-case communication requirements, which occur when \( q = 1 \). This supports the use of the assumptions of the above-presented worst-case scenarios for optimizing the tree degree.

5. Communication Overhead

As mentioned above, one motivation for using the broadcast seed is that it reduces the amount of communication overhead associated with notifying the users of the rekeying messages that are intended for them during member departures. This concept is explored below in the framework of a tree-based scheme.

Consider an \( a \) degree tree with \( n \) users. In a general tree-based scheme, when a user departs, all of the keys on the path from the departing member’s leaf to the root key are updated. To update a key associated with a particular node \( \sigma \), one must determine the keys associated with populated children nodes. These keys are then used to encrypt the update, and the rekeying message is then of the form:

\[ \alpha = \left\{ E_{K_h}(K_{\sigma}) \parallel E_{K_{\sigma_1}}(K_{\sigma}) \parallel \ldots \parallel E_{K_{\sigma_n}}(K_{\sigma}) \right\} \]
The sequence \( \{j_k\} \) is used to denote index the symbols of the valid children nodes. In addition to sending the rekeying message, the number of valid children nodes \( m \), and the sequence \( \{j_1, j_2, \ldots, j_m\} \) are sent.

The worst case scenario for communication overhead in updating a tree is when \( a \) of the children nodes are used to update each parent node. In this case, the communication overhead required is

\[
C_o = (a+1)\lceil \log_2 a \rceil \lceil \log_2 n \rceil
\]

This equation is obtained by considering both the communication needed to send the amount of valid children nodes, and the symbols for each valid child node.

This amount of communication overhead was calculated for different group sizes \( n \) and different tree degrees \( a \). The resulting amount overhead is depicted in Figure 8. In Figure 8, a baseline is drawn corresponding to \( B_\mu = 64 \) bits, which is the amount of communication overhead required if one uses the Member Departure protocol of Section IV.B. Examining the case of \( a = 4 \), which corresponds to the optimal value of the tree-degree as previously determined, shows that for values of \( n \) > 10000, the Member Departure protocol described above requires less communication overhead in the worst case scenario. Additionally, observe a higher degree tree is better suited to scenarios in which more users are joining than departing, then the efficiency of the Member Departure protocol is even more pronounced.

The use of a broadcast seed can gain further improvement if one elects to use \( \mu(t) = K_i(t-1) \). In this case, the broadcast seed does not have to be sent since it is known by the remaining users. Therefore, there is no communication overhead associated with updating during member departure, and one may consider the baseline at \( B_\mu = 0 \). In this case, the benefits of using a broadcast scheme becomes even more pronounced.
6. Computational Complexity

As discussed above, one advantage of the broadcast schemes of the present invention is that they reduce the amount, of communication overhead associated with sending flagging messages. As is apparent from the above discussion, a message form like Equation 17 takes less computation to form than a message form like Equation 18 assuming that calculating $E_K(K_o)$ has comparable computation as $f(K_o,l(l))$. Hence, to rekey using the message form requires more computation than when using a conventional rekeying message structure.

The worst-case computational complexity of this kind of rekeying form are compared below with residue-based rekeying. For such analysis, it is sufficient to focus on the update of the KEKs during a member departure since the other operations are identical.

In the residue-based rekeying form, each level of the key tree is rekeyed by calculating the product of $n$ numbers, each requiring $B + 1$ bits to represent, and the addition of a $B$ bit number with a $(B + 1)^a$ bit number. Using the fact that adding a $k$ bit number to an $l$ bit number requires $\max(k,l)$ bit operations, and multiplying a $k$ bit number by an $l$ bit, number requires $O(\text{kl})$ bit operations (N. Koblitz, A Course on Number Theory and Cryptography; Springer-Verlag, 2nd edition, 1994), one finds that for an $L$ level tree, the computational complexity of the residue-based message form is $O(LB^n)$.

Similarly, in the scheme presented above, $L$ levels of KEKs are to update. At each level of the tree one calculates the coefficients of a degree $\alpha-1$ interpolating polynomial, except at the bottom level where one calculates the coefficients of a degree $\alpha-2$ polynomial.

In order to calculate the coefficients of a $s$-degree interpolating polynomial, one may employ the Newton form of the interpolating polynomial (K. Atkinson, An Introduction to Numerical Analysis, John Wiley & Sons, 2nd edition, 1989). Algorithm 1 is a modification of the polynomial interpolation algorithm of G.
Golub and C. Van Loan. Matrix Computations, The Johns Hopkins University Press, 3rd edition, 1996, that can be used to determine the coefficients $b_j$ of the $s$-degree polynomial that interpolates the points $(x_j, y_j) \in \mathbb{Z}_p \times \mathbb{Z}_p$ where 

$j \in \{0, 1, \ldots, s\}$. The algorithm writes the $b_j$ values into the input array values $g_j$.

5 for $k = 0:s-1$ do 
  for $j = s-1:k+1$ do 
  \[ g(j) = (g(j)-g(j-1))(x(j-k-1))^{-1} \pmod{p} \] 
  endfor 
endfor 

10 for $k = s-1:-1:0$ do 
  for $j = k:s-1$ do 
  \[ g(j) = g(j)-g(j+1)x(k) \pmod{p} \] 
  endfor 
endfor 

Algorithm 1: Algorithm for determining the coefficients of an interpolating polynomial.

This algorithm requires addition, multiplication, inversion, and modulo operations to take place: modulo $p$. The most intensive operation of these is that of inverting a number. Assume that the prime $p$ is chosen to have $B$ bits, then the amount of bits operations needed to calculate the inverse of a number modulo $p$ using the Euclidean algorithm is $O(B^2)$ (N. Koblitz, A Course in Number Theory and Cryptography; Springer-Verlag, 2nd edition, 1994). The above algorithm requires \[ \frac{s(s+1)}{2} \] inversions in order to determine a degree $s$ interpolating polynomial. Therefore, the amount of bit operations needed to update an $L$ level degree $a$ key tree, using the polynomial interpolation scheme is $O(a^2LB^3)$.

Comparing the two computational estimates $O(LB^0)$ and $O(a^2LB^3)$ indicates that for higher tree degrees $a$, the scheme based upon polynomial interpolation requires fewer bit operations and is more computationally efficient.
In sum, in order to address the problem of managing keys for securing multicasts, protocols are described that are suitable for dynamic group environments. Two basic forms for constructing rekeying messages, the residue-based method and the polynomial interpolation method, are presented which make use of one-way functions to securely maintain keying information to users of a multicast session. Advanced protocol operations that update the keys during member joins, member departures, and the transferal of access rights were built using basic protocol operations referred to herein as protocol primitives. The invention provides a method for renewing session keys and key encrypting keys needed to control access to content. By using either the basic protocol operations, or more advanced protocol operations, the session key or key encrypting keys can be refreshed when a key's lifetime expires due to age or changes in group membership. It is also evident that if users were to collude, they would not be able to calculate the identity of keys for which they did not have proper access. Users may survive accidents or move across terminals by sending a request for reinsertion to the server, upon which the server performs the member reinsertion protocol operation. A description is additionally provided of a protocol operation that allows users to transfer their access rights to other parties. The server can revoke access to an individual by using the member departure operation to remove the member from the key hierarchy. Additionally, the described protocols use a tree-structured key hierarchy in order to achieve desirable communication requirements during changes in the group membership.

One feature of the schemes of preferred embodiments of the present invention is that they use either the residue-based method or polynomial interpolation in conjunction with a broadcast seed and one-way functions to handle member departure operations. It was observed that higher tree degrees are best for member join operations, whereas a tree degree of 3 or 4 was best for the member departure operation. When equally weighting the join and depart operations, a degree 4 tree stood out as optimal. The communication overhead of the polynomial interpolation scheme is reduced in comparison to a model conventional scheme. A comparison between the communication overhead of such schemes and the overhead
of an example conventional scheme that used ID messages to flag the users which parts of the rekeying message were intended for them, shows that as group size and tree degree increased, the communication overhead for a conventional scheme increased and ultimately became more burdensome than sending the broadcast seed.

For example, when the group size was \( n = 100000 \) and the tree degree was \( a = 4 \), the communication overhead in the conventional scheme was approximately 25% more than the overhead associated with a broadcast seed of size \( B_p = 64 \) bits.

Additionally, if one uses the previous session key \( K_s(t-1) \) as the seed \( \mu(t) \), then no communication overhead is associated with the protocol during member departures.

A study of the communication needed when using the architecture of the protocols of preferred embodiments of the present invention to perform member joins and member departures observed that these two operations are the most important operations that a multicast server will have to face when operating in dynamic environments. The communication requirements of the member join and member departure operations lead to conflicting tree design considerations. By explicitly computing these two quantities as functions of the degree of the tree and computing the communication overheads, tree selection criterion have been identified. From such computations, the communication during a member join is reduced when using a higher degree tree, while the optimal tree degree for a member departure is either \( a = 3 \) or \( a = 4 \). Consideration of the average of the communication for the two operations gave strong support to choosing \( a = 4 \) as the optimal tree degree. A stochastic population model was presented that allows one to study the mean behavior of the architecture of the protocols of preferred embodiments of the present invention for varying amounts of users. It is observed that for both the join and departure operation, the amount of communication needed to update the key tree rapidly increases as the tree approaches 10% population.

Above 10% occupancy, the communication needed for both operation stabilizes. The computational requirements of the tree-based rekeying schemes using polynomial interpolation and residue arithmetic were also compared. Estimates of the amount of bit operations needed in both cases indicate that the polynomial interpolation scheme required less computation for higher tree degrees.
VI. **Key Distribution for Multimedia**

There are two types of channels available for distributing the key information needed to secure multimedia multicasts, which are depicted in Figure 9. The first approach is to use a *media-independent* channel. The term “media-independent” is intended to denote that a separate channel needs to be used to convey the keying material. As an example, the specification of the MPEG-4 bitstream allows for the distribution of keying information via Intellectual Property Management and Protection Descriptors (IPMP-Ds) and Intellectual Property Management and Protection Elementary Streams (IPMP-ESs) (C. Herpel *et al.*, “Mpeg-4 Systems: Elementary Stream Management And Delivery,” In: A. Puri and T. Chen, editors, Multimedia Systems, Standards, and Network, pages 367-405. Marcel Dekker Inc., 2000).

A *media-dependent* approach to transmitting the rekeying information exists when small amounts of information can be embedded in the data. In these cases, the rekeying information may be embedded in the content and distributed to those who receive the data.

Data embedding, or digital steganography, techniques allow an information signal to be hidden in another signal, known as the “cover signal,” without dramatically distorting the cover signal. Effective data embedding techniques are those that can invisibly embed data in the cover signal, allow for easy extraction of the embedded information, and achieve a high embedding rate.

Many data embedding methods can be described by (B. Chen *et al.*, “Digital watermarking and information embedding using dither modulation,” IEEE Second Workshop on Multimedia Signal Processing, pp. 273-278, 1998). A general information embedding scheme is described in Figure 10. There is a host signal $x$ in which one wishes to embed some information $m$. The host signal could, for example, be a vector of pixel values or Discrete Cosine Transform (DCT) coefficients from an image. A data embedding function $H(x, m)$ maps $x$ and $m$ to a composite signal $s$ subject to some distortion constraint. The composite signal is
passed through a channel, where $s$ is subjected to various common signal processing
operations such as lossy compression, addition of noise, and attempts to remove the
embedded information. The decoder reconstructs $\hat{x}$ and extracts information
$\hat{m}$ after $y$ is received. It is desired that $\text{prob}(\hat{m} = m) \approx 1$ when the distortion satisfies

$$D(x, \hat{x}) \leq D_{\text{max}}.$$ 

In the literature, most of data hiding schemes focus on the embedding of
digital watermarks into multimedia data for the integrity of copyright and ownership
that requires security, imperceptibility and robustness. In this case, the embedding
technique must also be robust to attempts to remove or destroy the watermark. As a
result, the information rate embedded is small because of robustness.

Many papers exist on embedding information and watermarks in multimedia
Journal on Selected Areas in Communications, 16(4):525-540, May 1998; I. Cox et
al., “Secure Spread Spectrum Watermarking For Multimedia,” IEEE Tran. on Image
Watermarking Of Mpeg–2 Coded Video In The Bitstream Domain,” IEEE Int. Conf.
Acoustic Speech and Signal Proc. ’97, pages 2621–2624, 1997, the authors describe
a method for inserting digital watermarks into the bitstream of MPEG–2 coded
video. They found that they could embed a watermark of 1.25 to 125 bytes/second
in NTSC signals. In addition, their method operates in the compressed domain and
does not require complete decoding in order to embed information. Another method
for embedding information in video was presented in A. Westfeld et al.,
Steganography In A Video Conferencing System.” In: Proc. 2nd International
Workshop on Information Hiding, pages 32-47, 1998. That method used special
phase properties of cameras and scanners to alter the DCT coefficients in a manner
that is indistinguishable to viewers. The authors applied their method to a video
conferencing system and demonstrated that they were able to communicate
additional side information such as textual messages.
J. Song et al. ("A Data Embedding Scheme For H.263 Compatible Video Coding," *IEEE ISCAS* 4:390-393, June 1999) comprises another example of a data embedding scheme with a high embedding rate, that is compatible with standards such as H.263 and MPEG-2. In one embodiment, the present invention provides another such example. Such data embedding techniques use the fractional-pel motion vector as the cover signal for the embedded data, and are able to embed a high bit rate information signal into a video bitstream with an acceptable visual quality degradation.

Data embedding can also be used to convey side information, such as embedding messages in the content. Generic data structures are not well suited for hiding information. The properties of the data type must be exploited in order to achieve good embedding. Multimedia data types, such as speech, image, and video are well suited for embedding information since introducing a small amount of distortion in their waveforms does not significantly alter perceptual quality.

Associated with many embedding schemes is an embedding key that governs how the information is embedded into the cover signal. For example, 2 bits of information can be embedded per macroblock (J. Song et al. ("A Data Embedding Scheme For H.263 Compatible Video Coding," *IEEE ISCAS* 4:390-393, June 1999), and these 2 bits are embedded by mapping the motion vector to one of 4 regions. There are $4! = 24$ different ways to do this. One may therefore associate an embedding key $K_{emb}$ with one of these 24 different methods. If a user has the key associated with how the data was embedded, then he may extract the information signal in the multimedia data.

The key messages used in either the media-independent or media-dependent cases are almost identical. When using the media-independent approach, only the information needed to update the SK and KEKs needs to be transmitted. However, when using a media-dependent approach, an additional key, known as the embedding key, must be updated.
By using data embedding to convey the rekeying messages, an additional layer of security is available to the system. When data embedding is used, an external adversary will not only have to attack the SK and KEKs, but he will also have to attack the key governing the embedding rule in order to acquire rekeying messages. It is important that the key length of the embedding key be sufficiently long to make it difficult for the adversary to search the embedding key space. A preferred scheme for embedding key information is presented in Section VII, below.

The differences in the major operations of multicast key management: key refreshing, member joins, and member departures in a media-dependent and media-independent channel are discussed below.

VII. Data Embedding for Key Distribution in Multimedia Multicast

In this section, a new method for distributing the key information needed in secure multimedia multicast applications is described. In particular, the method can be used in conjunction with multimedia multicast applications such as video conferencing in which real-time video is coded and multicasted to a dynamic group. The key management schemes of the present invention can be used for any multicast communication regardless of the source information format. The rekeying messages are distributed explicitly to group members using a separate channel, or during breaks in data transmission.

In accordance with a preferred embodiment of the present invention, the use of an external channel is bypassed, and the data itself is employed as the mode of conveyance of key information. Since multimedia data is well suited for embedding data, key distribution is performed by embedding the rekeying messages within the multimedia data. By doing so, the network traffic associated with rekeying can be reduced and access to the media content is controlled by the combination of cryptography and steganography. This section will first briefly introduce data embedding, and describe the data embedding scheme that is used in preferred embodiments of the present invention. Next, the generic use of data embedding proposed for key distribution is described. The technical issue of synchronizing the
embedded bitstream is discussed for the specific data embedding technique used in accordance with preferred embodiments of the present invention, and a simple method for ensuring synchronization is described.

A. Methods of Data Embedding

As discussed above, most data hiding schemes focus on the embedding of digital watermarks into multimedia data for the integrity of copyright and ownership that requires security, imperceptibility and robustness. As a result, the information rate embedded is small. In accordance with the methods of the present invention, it is desirable to have a higher embedding rate than is employed in such schemes in order to convey keying messages to the group members in the multimedia multicast streams. The objective is to use data embedding for key message delivery instead of robustness for watermarking. In addition, the data embedding scheme of the present invention is designed to be compatible with standards such as H.263 and MPEG-2. This data embedding technique uses the fractional-pel motion vector as the cover signal for the embedded data. An advantage of this scheme is its ability to embed a high bit rate information signal into a video bitstream with a acceptable visual quality degradation (J. Song et al., “A data embedding scheme for h.263 compatible video coding.” Proc. of IEEE International Symposium on Circuits and Systems, vol. -1, pp. 390-393. June 1999). This method for data embedding will be used to demonstrate the feasibility of the multimedia multicast key distribution schemes of the present invention. The following section describes how this data embedding method would work for an H.263 encoder.

In the H.263 encoder; for every INTER-mode coded macroblock (MB), an integer-pixel motion vector (MV) is found first by motion estimation as in Figure 11, where the block (with dashed line) at integer-pixel A in the previous frame K is the motion prediction of the current MB (with solid line) in frame K+1. Then the half-pixel prediction (L-T. R. H263. “Version 2. Video coding for low bit rate communication.” Jan. 1998) is found by looking for the motion vector \( (D_x, D_y) \) with minimal sum of absolute difference (SAD) among half-pixels 1 ~ 8 and A, as illustrated in Figure 12. In a preferred embodiment of the present invention, data is
embedded by changing the half-pixel motion estimation. This can preferably be accomplished as follows:

MV's in H.263 can be classified into four ordered sets \( S_m, m \in \{0, 1, 2, 3\} \), which are \((I, I), (I, H), (H, I), (H, H)\) depending on the locations of an MV’s x and y components (at integer or half-pixel location of Figure 12) as shown in Table 2, where I stands for integer pixel and H for half-pixel.

<table>
<thead>
<tr>
<th>Set</th>
<th>( B_nB_{n+1} )</th>
<th>((d_x, d_y))</th>
<th>Pixel Locations</th>
</tr>
</thead>
<tbody>
<tr>
<td>( S_0 )</td>
<td>00</td>
<td>(I, I)</td>
<td>A</td>
</tr>
<tr>
<td>( S_1 )</td>
<td>01</td>
<td>(I, H)</td>
<td>2,7</td>
</tr>
<tr>
<td>( S_2 )</td>
<td>10</td>
<td>(H, I)</td>
<td>4,5</td>
</tr>
<tr>
<td>( S_3 )</td>
<td>11</td>
<td>(H, H)</td>
<td>1,3,6,8</td>
</tr>
</tbody>
</table>

Then two bits \( B_nB_{n+1} \) can be embedded in one MV by specifying the set \( S_m \) which an MV will belong to using \( m = 2B_n + B_{n+1} \), the MV \((D_x, D_y)\) of one MB in frame \( k \) is finally determined by:

\[
(D_x, D_y) = \arg \min_{(d_x, d_y) \in S_m} \sum_{i=0}^{15} \sum_{j=0}^{15} |f^k(i, j) - f^{k-1}(i + d_x, j + d_y)|
\]  

(39)

where \((d_x, d_y)\) is MV candidate corresponding to the pixels in set \( S_m \).

In other words, we find the MV not from the location with minimum SAD among half-pixels \( 1 \sim 8 \) and pixel-integer A, but match the MV components \( D_x \) and \( D_y \) to an MV set specified by the two bits to be embedded. The residual block will be DCT transformed and quantized as normal.

The extraction of the embedded data at decoder is straightforward: For an INTER-mode coded SIB. Its MV is decoded first, then the two bits \( B_nB_{n+1} \) embedded are extracted from the index of the set this MV belongs to as shown in Table 2.

The embedding rate of the scheme depends on the dimensions of video frame, video frame rate, motion activity of the sequence and video quality.
requirement. For example, for QCIF video of resolution 176 x 144, there are 99 macroblocks of size 16-by-16 in one frame. This means that one can embed at most 99 x 2 = 198 bits per frame. In fact, because some MBs in a frame are not coded (i.e., use MBs in the previous frame at the same spatial location), or are INTRA-mode coded, the number of INTER-mode coded MBs is less than 99. As a result, the number of bits one can embed in a frame is equal to two times the number of INTER-mode coded MBs.

B. Key Distribution Using Data Embedding

For multimedia multicast applications such as video conferencing, pay-per-view, etc., the generated $\alpha(t)$ rekeying message is preferably broadcast to group members by a data embedding scheme as shown in Figure 13.

In the normal case, the multimedia bitstream is encrypted by current SK $K_d(t-1)$ and then broadcast to users. Each legitimate user can decrypt the cipher text using current SK $K_d(t-1)$ and display the multimedia signals after source decoder. The SK $K_d(t-1)$ needs to be changed if the SK has expired or there is membership change. In this case the GC generates a new SK $K_d(t)$ and rekeying message $\alpha(t)$ as defined above. The rekeying message $\alpha(t)$ is embedded in the multimedia bitstream during source encoding. The compressed bitstream is still encrypted by the old SK $K_d(t-1)$. Each user can extract the rekeying message $\alpha(t)$ and obtain the new SK $K_d(t)$ while decoding the decrypted bitstream using old SK $K_d(t-1)$. The new SK $K_d(t)$ can be used for future secure group communication after a synchronized delay.

The advantages of a key distribution scheme using such data embedding are twofold: (a) The rekeying message can be distributed to group members without explicit use of the limited network resource; therefore network traffic for rekeying message can be reduced. (b) The proposed scheme is generic for all multimedia data embedding schemes – any data embedding technique can be used as long as it has a sufficient embedding rate.
C. Synchronization of Key Distribution Using Data Embedding

It is desirable to ensure that each member can extract the key message embedded in the video bitstream by GC and that the new session key can be used for encryption and decryption in a synchronized way.

If a bit string $B^L = \{b_l, l = 1, L\}$ will be embedded in the video bitstream, then in each P-frame, only the first $N_i$ Inter mode MBs can be used for data embedding, where $N_i$ is a system parameter, which trades off between the embedding bitrate and visual quality degradation. The MV of the first Inter MB in a P-frame is used to embed two bits $b_0b_1$ that signify synchronization information as in Table 3. For example, if $b_0b_1 = 01$, it means that bits $b_l, l = 1, \ldots, 2(N_i-1)$ in $B^L$ is embedded in the following $N_i$ Inter MBs of the current P-frame. To deal with variable $L$, let $N_i = \left\lfloor \log_2 L_{\text{max}} \right\rfloor$, $L_{\text{max}}$ is the maximal value of $L$. $N_i$ bits whose value is equal to $L$ and $B^L$ is concatenated. The sender and receiver can synchronize the bits embedded if the length $N_i$, $N_e$ and the meaning of $b_0b_1$ are known by both GC and receivers.

<table>
<thead>
<tr>
<th>$b_0b_1$</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>No bits encoded in this frame</td>
</tr>
<tr>
<td>01</td>
<td>Beginning of a bit string embedded</td>
</tr>
<tr>
<td>10</td>
<td>Middle of a bit string embedded</td>
</tr>
</tbody>
</table>

Assuming a block cipher such as DES is used for encryption, the video bitstream can be segmented into many cipher blocks. Preferably, the new session key is used to encrypt (or decrypt) the bitstream blocks following the block that contains the picture starting code (PSC) of a frame exactly after key message embedding (key message extraction) is finished. By doing so, the group members can extract the rekeying message and start to use the new session key for decryption which is synchronized to the encryption at GC.
D. System Feasibility Study

In this section, the issues related to the feasibility of using a key management system for multicast multimedia are discussed. When designing a cost effective system, one should consider the balance between computation, communication, and storage resources.

One of the primary advantages for using a tree-based key distribution scheme is that it achieves good scalability in the amount of communication needed to update the network. The need for using a tree-based key distribution scheme becomes more pronounced as the group size increases. If the group size is small, for example less than 10 users, there might not be any benefit from using a tree-based key distribution scheme, and one might want to consider the simple key distribution scheme discussed in Section IV. However, the $O(\log n)$ communication needed by most tree-based schemes makes the use of a tree-based scheme essential when the group size is several thousand or more users.

Another issue that should be considered is the amount of storage needed by the GC and each individual user. If each user has extremely limited storage, then the simple distribution scheme of Section IV might be appropriate. However, although a tree-based scheme may require more storage for each user, and a factor more storage for the GC, typically this is not as important of a consideration as communication resources.

As an example, in the scheme presented in Section IV, the amount of multiplications (computation) needed to update the KEKs for the bottom-up approach was calculated to be $C_{bu} = a \log_a n - 1$. The communication needed is proportional to the amount of computation needed. The amount of storage needed by the GC to keep track of the KEKs is:

$$S = \frac{a^{l+1} - 1}{a - 1}$$  \hspace{1cm} (40)
keys, while the amount of storage needed by each user is $\log_a n + 2$ keys.

Next, one should consider the channel that one is transmitting the keys across. Whether transmitting via a media-independent channel or a media-dependent channel, there is a channel rate that governs how quickly the keying information may be distributed. For example, suppose one is transmitting the rekeying information for the scheme of Section IV via an media-dependent channel. If one denotes $R$ as the embeddable channel rate (in bits/second), $B_{KEK}$ to be the key length of a KEK, $B_s$ to be the key length of the session key, $B_\mu$ the bit length of the random seed $\mu(t)$, and $B_{emb}$ to be the key length governing the data embedding rule, then the amount of time needed to update the entire system of keys is:

$$T = \frac{C_{emb} B_{KEK} + B_s + B_{emb} + B_\mu}{R}.$$  \hfill (41)

Since $T$ is related to the bit size of each of the keys, it is therefore related to the security levels protecting the service. This amount of time corresponds to the amount of time the departing member may still enjoy the service before no longer being able to decode the data (e.g., the video stream). If one desires to increase the level of protection of the multimedia, then $B_s$ must be increased, which leads to an increase in the amount of time needed to refresh the entire set of keys. Similarly, if one desires to increase the difficulty that an adversary would have in decoding rekeying messages, then one would need to increase $B_{KEK}$, which would also increase $T$.

In designing a system, these tradeoffs should be weighed and considered from a realistic point of view. Although it might be desirable to have extreme protection of the content, in a dynamic group, it may not be practical that it take an hour to update the set of keys.

To demonstrate these considerations, simulation results are presented using a preferred embodiment of the data embedding scheme of the present invention. The
degradation of the visual quality when different amounts of bits embedded per frame were measured for the Foreman and Miss America QCIF video sequences. The H.263 TMN-11 video codec was used with annexes D, I, J, F turned on (ITU-T Rec. H263. Version 2, Video Coding For Low Bitrate Communication. Jan. 1998). The bit rate in the simulation is 64kbps with a frame rate 10fps, and every 12th frame is INTRA coded. The peak signal-to-noise ratio (PSNR) of luminance component with different data embedding rates are compared with the PSNR of luminance without embedding. In simulations, the four cases compared correspond to when the number of bits embedded in a P-frame is upper bounded by 20, 40, 60 and no constraint (maximal). The PSNR differences are shown in Figure 14(a) for Foreman and Figure 14(b) for Miss America. Their average PSNR differences are also listed in Table 4. In all cases, the PSNR degradation of Luminance is within 1dB for both Foreman and Miss America, which normally cannot be detected by human visual system for video applications. Additionally, data embedding at half-pel motion estimation at most degenerates the video coding performance back to integer-pel motion estimation without data embedding.

<table>
<thead>
<tr>
<th>Table 4: Average PSNR Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td><strong>Foreman</strong></td>
</tr>
<tr>
<td><strong>Miss America</strong></td>
</tr>
</tbody>
</table>

Using this data embedding scheme in conjunction with the bottom-up approach to member departure discussed above, the amount of time needed to refresh the entire network of keys for a tree of degree $a = 2$, and $n = 2^{20}$ (approximately one-million users) can be calculated. For such calculation, the following values are employed, $B_{KEK} = 56$ bits, $B_s = .56$ bits, $B_{p} = 56$ and $B_{emb} = 20$ bits as the bit lengths for the various keys. These values for $B_{KEK}$, $B_s$, $B_p$ and $B_{emb}$ were chosen since they correspond to the key size of the popular block cipher DES. The resulting times needed to refresh the keys are presented in Figure 15. The curves illustrate an inverse relationship between time needed to refresh all keys and the amount of bits embedded per frame. Using these curves, one can determine the necessary embedding rate needed to refresh the keys in time $T$. For example, if one
had a video service of QCIF images with a frame rate of 20 frames/second, and desired to refresh the keys during member departure in $T = 5$ seconds, then 25 bits must be embedded per frame.

E. Performance of Data Embedding at Fractional-Pel Motion Vector

It is also desirable to analyse how the performance of video coding will be affected because of data embedding at fractional-pel motion vector.

A generalized hybrid video coding scheme is shown in Figure 16. The generalized hybrid coder combines a DPCM algorithm (differential pulse code modulation) along the motion-trajectory of the picture contents with a 2-D spatial intra frame encoder. The prediction value $\hat{s}$ takes into account a displacement estimate $(\hat{d}_x, \hat{d}_y)$ that is obtained by motion estimation based on the signal $s$. Since $s$ is not available at the receiver, $(\hat{d}_x, \hat{d}_y)$ has to be transmitted. The prediction error $e$ is encoded by the intra-frame source encoder that eliminates spatial redundancy front signal $e$. At the receiver, an intraframe source decoder generates the reconstructed prediction error $e'$, which differs from $e$ by some quantization noise. The transmitter contains a replication of the receiver in order to be able to generate the same prediction values $s'$ at the receiver. Regardless of the specific implementation of the intraframe source encoder, the accuracy of the displacement estimate has an important influence on the minimum bit rate that can be achieved by a hybrid coder for a given distortion. In J. R. Jain et al., "Displacement measurement and its application in interframe image coding," IEEE Trans. Communications, vol. COM-29, pp. pp.1799-1801, Dec. 1981, the rate-distortion function for a hybrid coding scheme related to that in Figure 16 has been presented using an intraframe DCT and Max quantization.


\[ \sigma_e^2 = E\{e^2\} - E^2\{e\} \]  

(42)

can be used to evaluate the performance of MCP. Figure 17 shows the general form of MCP, where \((x,y)\) denotes spatial coordinates. The prediction signal \(\hat{s}(x,y)\) is obtained from the samples of the reconstructed previous frame \(r(x,y)\), which is only available at sampling location \((x_i, y_i) \in \Pi\), where \(\Pi\) is the set of sampling positions. It is assumed that the input video signal \(s(x,y)\) has a power spectral density \(\Phi_s(\omega_x, \omega_y)\), and that the current frame can be predicted up to some residual noise \(n(x,y)\) of power spectral density \(\Phi_n(\omega_x, \omega_y)\) by translating the reconstructed previous frame \(r(x,y)\) by the true displacement \((d_x, d_y)\). If one assumes that the noise \(n(x,y)\), the signal \(s(x,y)\), and the displacement estimation error:

\[
\begin{pmatrix}
\Delta d_x \\
\Delta d_y
\end{pmatrix} =
\begin{pmatrix}
d_x \\
d_y
\end{pmatrix} -
\begin{pmatrix}
hat{d}_x \\
hat{d}_y
\end{pmatrix}
\]  

(43)

\[
\Phi_{ee}(\omega_x, \omega_y) = \Phi_x(\omega_x, \omega_y)
\]

\[
\cdot \left( 1 + |F(\omega_x, \omega_y)|^2 - 2\mathcal{R}\{F(\omega_x, \omega_y) P(\omega_x, \omega_y)\} \right)
\]

\[
+ \Phi_{m}(\omega_x, \omega_y)|F(\omega_x, \omega_y)|^2
\]

(44)

where $F(\omega_x, \omega_y)$ is the Fourier transform of the combined spatial filtering characteristic of the predictor, $\mathcal{R}\{\cdot\}$ denotes the real part of a complex number, and $P(\omega_x, \omega_y)$ is the band-limited 2-D Fourier transform of the continuous probability density function (pdf) $p(\Delta d_x, \Delta d_y)$ of the displacement error $(\Delta d_x, \Delta d_y)$,

\[
P(\omega_x, \omega_y) = \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} p(x, y) e^{-j\omega_xx-j\omega_yy} dx dy \forall |\omega_x| < \pi, |\omega_y| < \pi .
\]

Equation (44) allows one to study the influence of the displacement error pdf on the prediction error variance Equation (42), which can be calculated on the basis of Parseval's relation:

\[
\sigma_e^2 = \frac{1}{4\pi^2} \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} \Phi_{m}(\omega_x, \omega_y) d\omega_x d\omega_y
\]

(46)

The precise, shape of the displacement error pdf has hardly any influence on the variance of the motion compensated prediction error, $\sigma_e^2$, as long as the displacement error variance $\sigma_e^2$ does not change.

For a perfect displacement estimator that always estimates the true displacement, the displacement error $(\Delta d_x, \Delta d_y)$ is entirely due to rounding. The
displacement error will be uniformly distributed between $\pm 1/2 \beta X$ and $\pm 1/2 \beta Y$, where $\beta = 1$ for integer-pel accuracy, $\beta = 1/2$ for half-pel accuracy, etc. For a sampling grid with balanced horizontal and vertical resolution, $X = Y$, the minimum displacement error variance in moving areas is:

$$\sigma_{d_d}^2 = E\left[ (\Delta d_x)^2 + (\Delta d_y)^2 \right]$$

$$= \frac{1}{\beta X} \frac{1}{\beta Y} \int_{-\frac{\beta Y}{2}}^{\frac{\beta Y}{2}} \int_{-\frac{\beta X}{2}}^{\frac{\beta X}{2}} (x^2 + y^2) \, dx \, dy$$

$$= \frac{(\beta X)^2}{6}$$

(47)

Without loss of generality, assume that the half-per motion estimation based MV after rounding is at integer-per A as shown in Figure 12. The displacement error $(\Delta d_x', \Delta d_y')$ of the proposed data embedding scheme is due to the rounding to one of those locations specified by the two bits to be embedded with minimal prediction error. Assume the bits to be embedded are i.i.d. process with $p(b_n = 0) = p(b_n = 1) = 1/2$, $n = 0, 1, \ldots$, then one analyses the variance of displacement estimation error with data embedding in four cases under the assumption that, the prediction distortion function monotonically increases as one moves away from the optimal MV location along any direction in each of the four quadrants (J. R. Jain et al., “Displacement measurement and its application in interframe image coding,” IEEE Trans. Communications, vol. COM-29, pp.1799-1801, Dec. 1981).

- If the two bits to be embedded are “00” with probability 1/4, the MV is at $A$. The corresponding displacement error variance is:

$$\sigma_{00}^2 = \frac{(\beta X)^2}{6}$$

(48)

- If the two bits to be embedded are “01” with probability 1/4, the MV is selected from location 2 and 7 with smaller prediction error. As mentioned above, one
assumes that the location with smaller prediction error is always closer to the optimal MV location than other locations. The displacement error is now uniformly distributed as

\[ \Delta d_x \in \left[ -\frac{\beta X}{2}, -\frac{\beta X}{2} \right], \Delta d_y \in \left[ -\frac{\beta Y}{2}, -\frac{\beta Y}{2} \right] \cup \left[ \frac{\beta Y}{2}, \beta Y \right]. \]

The corresponding displacement error variance is

\[ \sigma_{0i}^2 = \frac{1}{\beta X} \frac{1}{\beta Y} 2 \int_{-\beta X/2}^{2 \beta X/2} \frac{\beta X}{2} \frac{\beta Y}{2} \left( x^2 + y^2 \right) dx dy = \frac{2(\beta X)^2}{3} \]  

(49)

- Similarly, if the two bits to be embedded are “10” with probability 1/4, the MV is selected from location 4 and 5 with smaller prediction error. The displacement error is now uniformly distributed as

\[ \Delta d_x \in \left[ -\beta X, -\frac{\beta X}{2} \right] \cup \left[ \frac{\beta X}{2}, \beta X \right], \Delta d_y \in \left[ -\frac{\beta Y}{2}, \frac{\beta Y}{2} \right]. \]

The corresponding displacement error variance is

\[ \sigma_{10}^2 = \frac{1}{\beta X} \frac{1}{\beta Y} 2 \int_{-\beta X/2}^{2 \beta X/2} \frac{\beta Y}{2} \frac{\beta X}{2} \left( x^2 + y^2 \right) dx dy = \frac{2(\beta X)^2}{3} \]  

(50)

- Finally, if the two bits to be embedded are “11” with probability 1/4, the MV is selected from location 1,3,6, and 8 with minimal prediction error. The displacement error is now uniformly distributed as

\[ \Delta d_x \in \left[ -\beta X, -\frac{\beta X}{2} \right] \cup \left[ \frac{\beta X}{2}, \beta X \right], \Delta d_y \in \left[ -\frac{\beta Y}{2}, \frac{\beta Y}{2} \right] \cup \left[ \frac{\beta Y}{2}, \beta Y \right]. \]

The corresponding displacement error variance is

\[ \sigma_{11}^2 = \frac{1}{\beta X} \frac{1}{\beta Y} 2 \int_{-\beta X/2}^{2 \beta X/2} \frac{\beta Y}{2} \frac{\beta X}{2} \left( x^2 + y^2 \right) dx dy = \frac{7(\beta X)^2}{6} \]  

(51)

And the displacement error variance in moving area is:

\[ \sigma_{\text{dis,EB}}^2 = \frac{1}{4} \sigma_{00}^2 + \frac{1}{4} \sigma_{01}^2 + \frac{1}{4} \sigma_{10}^2 + \frac{1}{4} \sigma_{11}^2 = \frac{4(\beta X)^2}{6} = 4\sigma_{\text{dis}}^2 \]  

(52)
Which shows that data embedding at fractional-pel motion vector increases the variance of motion estimation error by four times, i.e., the variance of motion compensated prediction error using data embedded motion estimation at βX accuracy is equivalent to the variance of motion compensated prediction error without data embedding at 2 βX motion estimation accuracy.

VIII. Extensions to Multilayered Services

In many application environments, the multimedia data is distributed in a multi-layered form. For example, in an HDTV broadcast, users with a normal TV receiver can still receive the current format, while other users with an HDTV receiver can receive both the normal format and the extra information needed to achieve HDTV resolution. As another example, the MPEG-4 standard allows for multiple media streams corresponding to different object planes to be composited. In either of these cases, it will be desirable for service providers to separately control access to the different layers of media. The key management schemes must therefore be considered separately. As an example of how this can be done using the tree-based scheme of Section IV, the problem of managing keys for two levels of service corresponding to a low quality and high quality service are considered. Those of ordinary skill will, in light of the present specification be able to apply the teachings of this section to more than two levels of service, or to more than two layers/objects of multimedia content; such extensions being within the scope of the present invention.

For the purposes of such explanation, the multimedia data stream is considered to consist of two layers, which are denoted as $D^l$ and $D^h$. $D^l$ provides the low resolution service only, while high-quality service can be obtained by receiving both the base-layer $D^l$ and the refinement layer $D^h$. The GC will have two session keys $K^l_s(t)$ and $K^h_s(t)$. $K^l_s(t)$ is used to encrypt $D^l$ and $K^h_s(t)$ is used to encrypt $D^h$. Similarly, each internal node in the key tree has two KEKs $K^l_o(t)$ and $K^h_o(t)$, where $o$ is the index of the nodes in the tree (Figure 19). Group members who want to receive the lower quality service will be assigned the low-layer session key, as
well as low-layer keys from the root to the leaf that stands for this member. Group members who want to receive high quality service will be assigned both the low-layer and high-layer keys. The rekeying scheme is similar to the one layer case described above.

**Refreshing The Low-Quality Session Key:** The new session key associated with the low-quality level may be refreshed by encrypting with the root low-quality KEK \( K'_z(t) \) and transmitting the message

\[
\alpha'_z(t) = E_{K'_z(t)}(K'_z(t)).
\]

**Refreshing The High-Quality Session Key:** The procedure for refreshing the high-quality session key is identical to the procedure for refreshing the low-quality session key, but using \( K'_z(t) \) and \( K'_z(t) \) instead.

**New Member Joins Low-Quality Service:** A new member may desire to join the low level service. In this case, the low-quality session key and KEKs must be renewed, which can be done by applying the procedure of Section IV.A.

**New Member Joins High-Quality Service:** A new member may desire to join the high level service. In this case, both the low-quality and high-quality keys must be renewed. To do this, the procedure of Section IV.A is applied twice, once for the low-quality keys, and once for the high-quality keys.

**High-Quality User Leaves The Group:** In this case, both session key \( K'_z(t-1) \) and \( K'_z(t-1) \) and corresponding KEKs for both \( D'h \) and \( D'h \) have to be changed. This can be done using the algorithms in Section IV.B. twice.

**Low-Quality User Leaves The Group:** In this case, only session key \( K'_z(t-1) \) and corresponding KEKs for base-layer \( D'h \) need to be changed, which can be done using the algorithms in Section IV.B once on the appropriate low-layer keys.
**Low-Quality User Changes To High-Quality:** In this case, the high-layer SK $K^h_{i}(t-1)$ as well as the high-layer KEKs must be changed to prevent the user from accessing the past high quality service. The new SK $K^h_{i}(t)$ and KEKs keys from root to the leaf are directly given by the GC to this user during registration to the new level of service.

**High-Quality User Change To Low-Quality:** The session key $K^h_{i}(t-1)$ and corresponding KEKs for high-layer have to be changed to prevent this user from accessing the future high quality information. This can be done using the algorithms in Section IV once on the high-layer KEKs.

Having now generally described the invention, the same will be more readily understood through reference to the following examples, which are provided by way of illustration, and are not intended to be limiting of the present invention, unless specified.

**Example 1**

**MPEG-4 Specific Issues**

In this Example, the use of multicast key management schemes in the MPEG-4 IPMP framework is discussed. Although, for purposes of illustration, the use of the Residue Approach to key management and distribution is described with respect to the MPEG-4 IPMP, it will be appreciated that the Interpolation Approach to key management and distribution may be equivalently employed in such a framework. The MPEG-4 IPMP framework provides a powerful and flexible method for distributing the rekeying messages needed for maintaining access control to MPEG-4 content.

The MPEG-4 IPMP architecture is detailed in ISO/IEC 14496–1. The IPMP architecture uses a separation between the IPMP system and the standardized components of MPEG-4. By using IPMP control points in the MPEG-4 object/stream model, it is possible to develop many non-normative solutions to controlling access to content, monitoring copyright issues, and managing patent

The preferred scheme of the present invention can be exemplified by reference to a secure MPEG-4 content multicast application. Such an application is similar to the secure content delivery application described in N2614, but uses multicast key management schemes to deliver content securely to groups of users. As described below, media-dependent and media-independent forms of key distribution can be incorporated into the IPMP architecture. Moreover, as described below, the multicast management operations of Member Join and Member Departure are powerful primitives that allow for many of the IPMP issues detailed in Annex A of the Call for Proposals to be easily addressed.

A. MPEG-4 Secure Multicasts

The applicability of the present invention can be seen by considering the secure MPEG-4 multicast scenario depicted in Figure 18. In such application scenario, there is the service provider, an MPEG-4 server, which is combining multiple multimedia objects from many different content providers. The service provider distributes the encrypted and combined content to many multimedia terminals using possibly many different telecommunication infrastructures. The clients can have different terminal profiles, for example, some may be using televisions to enjoy the service while other might be using hand-held devices.

The differences in terminal profiles means that the users may want to access only some of the media objects, and probably not the same objects as another user.
It is reasonable to assume that clients pay only for those multimedia objects that they consume. Therefore, it is desirable that the service provider be able to control access to the objects separately.

Each media object has associated with it a key tree. These key trees correspond to the arrangement of keys associated key management schemes. It is not necessary to use the same key size of tree, or even the same key management scheme for the different objects. However, standardizing the key management scheme has a facilitating affect on user mobility.

The server has a database that stores multiple key trees. The server is responsible for tracking the subscription of users to different objects, as well as placing the users on the different key trees. It is not necessary that a user have the same location on different trees. The server has encryption engines that are used to encrypt the bulk content with corresponding session keys, as well as key management units devoted to calculating the rekeying messages that must be distributed to users. Additionally, the server has other cryptographic tools at its disposal, such as random number generators and message authentication codes.

Each client, on the other hand, maintains a personal key database. This key database stores the keys that the user needs to decode the content to which he is subscribed. The client also has encryption/decryption engines used to decrypt content, as well as components needed to extract new keying information.

The keying messages may be distributed using the MPEG-4 IPMP framework by two different approaches: using the media-independent IPMP-ESs, or using a media-dependent approach such as data embedding.

1. Media-Independent

The most natural approach to distributing the keys in the MPEG-4 IPMP architecture is to use the elementary streams. In Figure 20, the use of a media-independent method for distributing keys with MPEG-4 IPMP is depicted. During the multicast of MPEG-4 content to the users, the server distributes the keying
messages via the IPMP-ESs, while keys are associated with content by the use of IPMP-DSs.

Each client has IPMP systems associated with the different streams of content that he is consuming. It is possible that an IPMP system can govern the access to multiple streams. The client’s IPMP systems extract the keys needed to update the key database. The keys that were transmitted are used to decrypt the content during the time interval in which they are valid. The plaintext versions of the content is passed to the appropriate codec modules for decoding, and composited. The IPMP systems also prevent non-decrypted streams from being passed onto the decoding phase.

The issue of delay and synchronizing the keys with the content is important. The MPEG-4 format provides natural synchronization by associating each atom with a time period. Also, during member changes, there will be a short delay before invalidated users can no longer receive content. When using IPMP-ES to distribute the keys, it is possible to send the keys needed to decrypt the content for time frame $t$ during time frame $t$.

2. Media-Dependent

Many of the media streams may have the ability to have data embedded in them. Data embedding techniques that are compliant with the MPEG-4 format can be used to transmit the keys. The client side of key distribution using data embedding is depicted in Figure 21. Note that it remains possible to transmit some of the keys via the IPMP bitstream; Figure 21 depicts the possibility of using the IPMP-ES for some of the objects by leaving the IPMP bitstream in the diagram.

In the scheme shown in Figure 21, each client's IPMP system extracts the keys needed to update the key database by first decrypting the content using the current keys, decoding the content bitstreams, and extracting the keying messages. The keying messages are used by the IPMP systems to determine the new keys, and update the client's key database.
The issue of delay between the time of a membership change and the refreshing of the key network becomes more pronounced when using the media-dependent channel. Decryption must take place prior to the ability to extract any key information. Therefore, it is not possible to transmit keys needed to decode time frame $t$ during time frame $t$. As a consequence, there is an implicit delay between the time a member departs and the time he is no longer able to enjoy the service. However, as noted above, there is the advantage that there is an additional layer of security against attacks by external adversaries associated with using the embedding rule.

**B. MPEG-4 IPMP Issues**

The Call for Proposals (N3543) set down a list of requirements in its Annex A that a proposed technology should address. Table 5 provides a list of the requirements as stated and enumerated in Annex A of N3543. The second column states whether the requirement can be satisfied using a multicast key management approach, particularly one such as the preferred scheme of the present invention, or whether the requirement is a business dependent requirement that can be addressed through additional effort. The third column states whether further discussion is provided below.
Table 5

<table>
<thead>
<tr>
<th>Requirement No.</th>
<th>Requirement Status</th>
<th>Further Discussion Provided</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Satisfied</td>
<td>Yes</td>
</tr>
<tr>
<td>2</td>
<td>Satisfied</td>
<td>Yes</td>
</tr>
<tr>
<td>3</td>
<td>Business Development</td>
<td>No</td>
</tr>
<tr>
<td>4</td>
<td>Satisfied</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>Satisfied</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>Business Development</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>Satisfied</td>
<td>Yes</td>
</tr>
<tr>
<td>8</td>
<td>Satisfied</td>
<td>No</td>
</tr>
<tr>
<td>9</td>
<td>Satisfied</td>
<td>Yes</td>
</tr>
<tr>
<td>10</td>
<td>Satisfied</td>
<td>No</td>
</tr>
<tr>
<td>11</td>
<td>Satisfied</td>
<td>Yes</td>
</tr>
<tr>
<td>12</td>
<td>Satisfied</td>
<td>Yes</td>
</tr>
<tr>
<td>13</td>
<td>Satisfied</td>
<td>Yes</td>
</tr>
<tr>
<td>14</td>
<td>Satisfied</td>
<td>Yes</td>
</tr>
<tr>
<td>15</td>
<td>Business Development</td>
<td>No</td>
</tr>
<tr>
<td>16</td>
<td>Satisfied</td>
<td>Yes</td>
</tr>
<tr>
<td>17</td>
<td>Satisfied</td>
<td>No</td>
</tr>
<tr>
<td>18</td>
<td>Satisfied</td>
<td>No</td>
</tr>
<tr>
<td>19</td>
<td>Satisfied</td>
<td>No</td>
</tr>
<tr>
<td>20</td>
<td>Satisfied</td>
<td>No</td>
</tr>
<tr>
<td>21</td>
<td>Satisfied</td>
<td>No</td>
</tr>
</tbody>
</table>

It is to be emphasized that the following content-key invariant is useful in the following sections: there is a unique correspondence between sets of keys and substrings of multimedia content. This association is established by the time period in which the set of keys is used.

1. The solution shall support access to and interaction with content while keeping software and hardware requirements to a minimum

As discussed above, multicast key management can be used to receive and play an MPEG-4 multicast. Users may also wish to record a service that they have paid for. In this case, the receiver who wishes to record the service writes the complete multimedia stream to file, including the IPMP-ESs. When the user wishes to playback a recorded stream, he inputs his initial keys, which should be stored in a cryptographically secure container on the terminal or in a smart card. This is not an additional requirement since the cryptographic keys used for encryption must be
securely stored for any IPMP scheme. Using the initial keys, the client’s terminal will be able to use the IPMP-ESs and IPMP-Ds to reconstruct the dynamics associated with key changes, and therefore be able to decrypt and decode the content for playback. This approach can also be accomplished when using data embedding to transfer the rekeying messages.

In order to edit a recorded service, the content must be decrypted and decoded. After the edits are applied to the content, the content streams can be re-encrypted using the session keys extracted from the rekeying messages. The MPEG-4 bitstream format makes it easy to associate keys with time frames of a media timeline. Due to the content-key invariance, deletion of a portion of the content will also delete a corresponding unique set of keys. The remaining keys are then securely stored.

It should also be mentioned that other IPMP frameworks, such as introducing watermarks to indicate that a content is a copy, should be used in conjunction with multicast key distribution for allowing interaction with content.

2. **Solution shall support easy interaction with content from different sources without the swapping of physical modules**

In this case, each service provider maintains its own key data base and encrypts its content with session keys. At the client side, the MPEG-4 terminal receives the bitstreams from each source, extracts appropriate keys, and maintains a database that associates keys with specific content streams.

4. **The solution shall support the protection of user privacy**

The discussion of this issue is covered in more detail as a consequence of the discussion of Requirement 5.

5. **The solution shall support service models in which the end user’s identity is not disclosed to the service/content provider**

User privacy and user anonymity are easily supported using the multicast framework. The content providers may be separated from the service provider, or
they may be the same institution. Similarly, the group key manager, i.e. the GC, can be separated from the service provider. When using a service model where the service provider is distinct from the GC, an additional layer exists that separates the user from the service provider. If the GC is an authority trusted by the client then user privacy and anonymity can be maintained.

Additionally, the very nature of multicast communications provides a decoupling between the senders and receivers. For example, in IP multicast (S. Keshav, "An Engineering Approach to Computer Networking: ATM Networks, the Internet, and the Telephone Network. Addison Wesley, 1997), such a separation naturally exists. Senders and receivers are associated with each other by sharing a class D IP address. A member wishing to send to the group sends messages to the shared address. The decoupling of the sender and receivers provides receiver anonymity.

7. The solution shall support the content and the end user's rights to interact with it to survive common accidents

Common accidents, such as a system crash, might mean that the client misses several rekeying messages needed to update his key database. In order for the client to be able to survive such an accident, upon reboot of the terminal, the client sends a signal upstream to the service provider using either MPEG-4 upstreams or via a separate channel. This signal needs to inform the server of the state of the client's key database, perhaps by using a timestamp to state the last time frame during which he had a valid key database. The GC can determine what keys he needs and use either the multicast MPEG-4 bit stream or a separate channel to distribute any needed keys. This can be accomplished securely by encrypting the keys with the user's unique private key (i.e., a leaf node key).

9. The solution shall support content mobility across MPEG-4 terminals, e.g. end users should be able to move to a different terminal and keep their rights to interact with the content

When a user moves from one terminal to another, he must log into the new terminal and send a signal to the GC that he has moved. This signal must involve
some method to authenticate the user, such as using a password. The GC then securely distributes the current keys that belong to that user on the key tree. Secure distribution is accomplished by encrypting with the user’s unique private key. Observe that this requirement and its solution are very similar to those of Requirement 7.

11. The solution shall support content and the end user's rights to interact with it to survive changing to a different type of MPEG-4 hardware

When a user changes to a different type of MPEG-4 hardware, the terminal profile might change. This means that it might not be possible for the user to consume all portions of the content for which he is paying. Therefore, when logging into the new terminal, he must be able to specify which content layers he wishes to consume. The terminal will only maintain the keys associated with these contents, and he will experience a subset of the content to which he has subscribed. In the case in which a user moves to a more powerful MPEG-4 terminal, he may experience all of the content to which he has subscribed.

12. The solution shall support the transferring, temporarily or permanently, content and the rights to interact with it to another party

The transfer of content and rights is an important problem that can be addressed using the multicast key distribution framework. Consider an application scenario in which content is being multicast to many users, and that the keys are arranged on a tree as in Figure 3(a). User $u_A = u_{111}$ would like to give his membership to user $u_B$. In order to do this, $u_A$ contacts the GC stating that he would like to transfer rights to $u_B$. The GC must authenticate this claim. The GC then issues a new private key $K_{111}^B$ to user $u_B$. The GC then replaces $K_{111}$ with $K_{111}^B$ on the tree, and performs a modified member join operation that renews the keys $K_{11}$, $K_i$, $K_e$ and $K_s$ from the bottom up. The modified member join operation is similar to the member departure protocol, but is used to insert a member into a position in a key tree that is already occupied. First, $K_{111}^B$ and $K_{111}$ are used to update $K_{11}$ via:
\[ \alpha_{11}(t) = K_{11}(t) + f(K_{110}(t), \mu(t)) f(K^g_{111}(t), \mu(t)). \] (53)

The remainder of the member departure protocol is then followed, updating \( K_I, K_e \) and \( K_d \) from the bottom-up. Because the updating is from the bottom-up, and uses \( K_{111}^g \), \( u_d \) will no longer be able to access the communication.

If user \( u_d \) would like to have the rights transferred back to him, he must send an authenticated message to the GC. The GC performs a modified member join operation that replaces \( K_{111}^g \), and updates the remaining keys from bottom-up. This action effectively removes \( u_B \) from the service since his keys are no longer valid.

13. **The solution shall enable content owners to control which of their assets are available when, where, and under what conditions**

The access control to the content can be easily governed using the multicast key management framework. If the service provider decides that certain users should not be allowed to access specific content, the service provider may perform member departure operations to remove such users from the key tree associated with this content. At a later time, the service provider may use member join operations to allow members back into the service.

14. **The solution shall support persistent security over time and renewability of that security**

The use of a multicast key distribution scheme allows for the periodic refreshing of the session key. The SK’s cryptoperiod can be shortened if the provider feels that the value of the content is great, or if a threat to the sanctity of the SK is perceived.

15. **The solution shall enable content owners to change business rules whenever and however they wish**

The business rule does not play an explicit role in the distribution of keys to users. The manner in which the content owners and service providers charge the users is independent of the keying strategy. It is easy, however, to envision that some object layers may be unprotected, and free for any to access, while other layers
would be protected and access to such layers would be available only to users who pay to access them.

In summary, the secure distribution of multimedia multicasts necessitates the distribution and management of keying material. Two different approaches for distributing the keys needed to protect multimedia multicasts are presented, corresponding to whether the keying information is distributed by a media-independent channel, or embedded in the content itself. By embedding the keying information in the multimedia content, access to the keying information is protected by an additional layer of security against external attacks.

The rekeying messages need to make efficient usage of communication resources, and must be robust to attacks by both non-members and members. The present invention provides a new form for the rekeying messages that employs one-way functions and a broadcast seed. It can withstand collusion and allows for user specific information to be reused. When the group size becomes large, efficient usage of communication resources is achieved by mapping the message form to a logical tree. The scalable message form of the preferred scheme of the present invention is used to illustrate that the amount of time needed to update the entire network of keys is related to the key lengths used, and the embeddable channel rate. Through the use of multiple key trees, multicast key distribution schemes can be extended to protect multiple layers of multimedia content in an efficient manner.

The multicast key distribution schemes of the present invention satisfy the requirements of MPEG-4 IPMP architecture. The preferred scheme of the present invention use computationally efficient one-way functions to provide robustness to attacks. Both the media-independent and media-dependent approaches to distributing keying information are possible using the IPMP framework. The use of member join and member departure operations allows for the system to handle cases in which the users and content are mobile, allows for users to survive common accidents such as system crashes, and allows users to transfer rights to other users.
While the invention has been described in connection with specific embodiments thereof, it will be understood that it is capable of further modifications and this application is intended to cover any variations, uses, or adaptations of the invention following, in general, the principles of the invention and including such departures from the present disclosure as come within known or customary practice within the art to which the invention pertains and as may be applied to the essential features hereinbefore set forth. The references cited above are hereby incorporated herein in their entirety.
What is claimed is:

1. A computer-facilitated protocol for accomplishing the dynamic management of key information between a group center and one or more group members in order to provide a secure multicast of multimedia data, wherein said protocol permits said Group Center to provide key information to said group members through the use of a parametric one-way function and a broadcast seed, or by subjecting the transmitted data to modulo analysis using previously sent key information.

2. The computer-facilitated protocol of claim 1, wherein said protocol permits said Group Center to provide key information to said group members through the use of a parametric one-way function and a broadcast seed.

3. The computer-facilitated protocol of claim 2, wherein said broadcast seed is a random broadcast seed.

4. The computer-facilitated protocol of claim 3, wherein said broadcast seed is key information previously transmitted to said one or more group members.

5. The computer-facilitated protocol of claim 2, wherein said key information comprises a session key, and one or more personal key encrypting keys.

6. The computer-facilitated protocol of claim 5, wherein said protocol employs a tree-based algorithm to update said session key and said personal key-encrypting key(s).

7. The computer-facilitated protocol of claim 5, wherein said key information is provided using a media-independent channel.

8. The computer-facilitated protocol of claim 5, wherein said key information additionally comprises an embedding key.

9. The computer-facilitated protocol of claim 8, wherein said key information is provided using a media-dependent channel.
10. The computer-facilitated protocol of claim 2, wherein said parametric one-way function \( h \) maps a sequence \((x, y)\) of 2\( B \) bits to \( B \) bits.

11. The computer-facilitated protocol of claim 2, wherein said parametric one-way function additionally comprises a function \( f(x, y) = 1|| h(x, y) \), wherein \( f \) optionally prepends a single 1 bit in front of the output of \( h(x, y) \).

12. The computer-facilitated protocol of claim 11, wherein \( f \) prepends a single 1 bit in front of the output of \( h(x, y) \).

13. The computer-facilitated protocol of claim 11, wherein \( f \) does not prepend a single 1 bit in front of the output of \( h(x, y) \).

14. The computer-facilitated protocol of claim 2, wherein said parametric one-way function is used in calculating a degree \( n-2 \) polynomial \( p(z) \) that interpolates the points \((z_j, w_j)\), where \( z_j \) is a quantity that identifies user \( j \), and \( w_j = K_x (t) + f (K_{j'}, \mu(t)) \) (mod \( p \)).

15. The computer-facilitated protocol of claim 1, wherein said protocol permits said Group Center to provide key information to said group members by subjecting the transmitted data to modulo analysis using previously sent key information.

16. The computer-facilitated protocol of claim 15, wherein in said modulo analysis, said Group Center forms \( K_x (t) \) and calculates the rekeying message as: \( \alpha_x (t) = K_x (t) + \prod_{i=1}^{n-2} f (K_{i'}, \mu(t)) \).

17. The computer-facilitated protocol of claim 15, wherein said key information comprises a session key, and one or more personal key encrypting keys.

18. The computer-facilitated protocol of claim 17, wherein said protocol employs a tree-based algorithm to update said session key and said personal key-encrypting key(s).
19. The computer-facilitated protocol of claim 17, wherein said key information is provided using a media-independent channel.

20. The computer-facilitated protocol of claim 17, wherein said key information additionally comprises an embedding key.

21. The computer-facilitated protocol of claim 20, wherein said key information is provided using a media-dependent channel.

22. The computer-facilitated protocol of claim 1, wherein said multimedia data is provided to said group member in multi-layered form.

23. The computer-facilitated protocol of claim 1, wherein said protocol accomplishes the dynamic management of key information in conjunction with the MPEG-4 Intellectual Property Management protocol.

24. The computer-facilitated protocol of claim 1, wherein bits containing key data are embedded in said multimedia data by changing half-pixel motion estimation to an integer-pixel motion vector of said multimedia data.

25. The computer-facilitated protocol of claim 24, wherein two bits of key data $B_n, B_{n+1}$ are embedded in one motion vector of said multimedia data by specifying a set $S_m$ which a motion vector will belong to using $m = 2B_n + B_{n+1}$, wherein the motion vector $(D_x, D_y)$ of one multiblock in frame $k$ is determined by:

$$
(D_x, D_y) = \arg \min_{(d_x, d_y) \in S_m} \sum_{i=0}^{15} \sum_{j=0}^{15} f^k(i, j) - f^{k-1}(i + d_x, j + d_y)
$$

where $(d_x, d_y)$ is a motion vector candidate corresponding to the pixels in set $S_m$.

26. A computer system comprising a Group Center computer and at least one group member computer, wherein in said system, said Group Center computer securely transmits multimedia data to at least one group member computer wherein said security is accomplished through the use of a
parametric one-way function and a broadcast seed, or by subjecting the transmitted data to modulo analysis using previously sent key information.

27. The computer system of claim 26, wherein said security is accomplished through the use of a parametric one-way function and a broadcast seed.

28. The computer system of claim 27, wherein said broadcast seed is a random broadcast seed.

29. The computer system of claim 28, wherein said broadcast seed is key information previously transmitted to said one or more group members.

30. The computer system of claim 27, wherein said key information comprises a session key, and one or more personal key encrypting keys.

31. The computer system of claim 30, wherein said protocol employs a tree-based algorithm to update said session key and said personal key-encrypting key(s).

32. The computer system of claim 29, wherein said key information is provided using a media-independent channel.

33. The computer system of claim 29, wherein said key information additionally comprises an embedding key.

34. The computer system of claim 32, wherein said key information is provided using a media-dependent channel.

35. The computer system of claim 27, wherein said parametric one-way function $h$ maps a sequence $(x,y)$ of $2B$ bits to $B$ bits.

36. The computer system of claim 27, wherein said parametric one-way function additionally comprises a function $f(x,y) = 1||h(x,y)$, wherein $f$ optionally prepends a single 1 bit in front of the output of $h(x,y)$. 
37. The computer system of claim 36, wherein \( f \) prepends a single 1 bit in front of the output of \( h(x, y) \).

38. The computer system of claim 36, wherein \( f \) does not prepend a single 1 bit in front of the output of \( h(x, y) \).

39. The computer system of claim 27, wherein said parametric one-way function is used in calculating a degree \( n-2 \) polynomial \( p(z) \) that interpolates the points \( (z_j, w_j) \), where \( z_j \) is a quantity that identifies user \( j \), and
\[
w_j = K_x(t) + f(K_j, \mu(t)) \quad \text{(mod } p)\,.
\]

40. The computer system of claim 26, wherein said security is accomplished by subjecting the transmitted data to modulo analysis using previously sent key information.

41. The computer system of claim 40, wherein in said modulo analysis, said Group Center forms \( K_e(t) \) and calculates the rekeying message as:
\[
\alpha_e(t) = K_x(t) + \prod_{i=1}^{n-1} f(K_i, \mu(t))\,.
\]

42. The computer system of claim 40, wherein said key information comprises a session key, and one or more personal key encrypting keys.

43. The computer system of claim 42, wherein said protocol employs a tree-based algorithm to update said session key and said personal key-encrypting key(s).

44. The computer system of claim 42, wherein said key information is provided using a media-independent channel.

45. The computer system of claim 42, wherein said key information additionally comprises an embedding key.

46. The computer system of claim 45, wherein said key information is provided using a media-dependent channel.
47. The computer system of claim 26, wherein said multimedia data is provided to said group member in multi-layered form.

48. The computer system of claim 26, wherein said protocol accomplishes the dynamic management of key information in conjunction with the MPEG-4 Intellectual Property Management protocol.

49. The computer system of claim 26, wherein bits containing key data are embedded in said multimedia data by changing half-pixel motion estimation to an integer-pixel motion vector of said multimedia data.

50. The computer system of claim 49, wherein two bits of key data $B_nB_{n+1}$ are embedded in one motion vector of said multimedia data by specifying a set $S_m$ which a motion vector will belong to using $m = 2B_n + B_{n+1}$, wherein the motion vector $(D_x, D_y)$ of one multiblock in frame $k$ is determined by:

$$
(D_x, D_y) = \arg\min_{(d_x, d_y) \in S_m} \left( \sum_{i=0}^{15} \sum_{j=0}^{15} |f^k(i, j) - f^{k-1}(i+d_x, j+d_y)| \right)
$$

where $(d_x, d_y)$ is a motion vector candidate corresponding to the pixels in set $S_m$.

51. A computer specially adapted by software to be able to function as a group center computer and to securely transmit multimedia data to at least one group member computer wherein said software permits said specially adapted computer to accomplish said security through the use of a parametric one-way function and a broadcast seed, or by subjecting the transmitted data to modulo analysis using previously sent key information.

52. The specially adapted computer of claim 51, wherein said security is accomplished through the use of a parametric one-way function and a broadcast seed.

53. The specially adapted computer of claim 52, wherein said broadcast seed is a random broadcast seed.
54. The specially adapted computer of claim 52, wherein said broadcast seed is key information previously transmitted to said one or more group members.

55. The specially adapted computer of claim 52, wherein said key information comprises a session key, and one or more personal key encrypting keys.

56. The specially adapted computer of claim 55, wherein said protocol employs a tree-based algorithm to update said session key and said personal key-encrypting key(s).

57. The specially adapted computer of claim 55, wherein said key information is provided using a media-independent channel.

58. The specially adapted computer of claim 55, wherein said key information additionally comprises an embedding key.

59. The specially adapted computer of claim 58, wherein said key information is provided using a media-dependent channel.

60. The specially adapted computer of claim 52, wherein said parametric one-way function $h$ maps a sequence $(x, y)$ of $2B$ bits to $B$ bits.

61. The specially adapted computer of claim 60, wherein said parametric one-way function additionally comprises a function $f(x, y) = 1 \| h(x, y)$, wherein $f$ optionally prepends a single 1 bit in front of the output of $h(x, y)$.

62. The specially adapted computer of claim 52, wherein said parametric one-way function is used in calculating a degree $n-2$ polynomial $p(z)$ that interpolates the points $(z_j, w_j)$, where $z_j$ is a quantity that identifies user $j$, and $w_j = K_z(t) + f(K_j, \mu(t)) \pmod{p}$.

63. The specially adapted computer of claim 51, wherein said security is accomplished by subjecting the transmitted data to modulo analysis using previously sent key information.
64. The specially adapted computer of claim 63, wherein in said modulo analysis, said Group Center forms \( K_e(t) \) and calculates the rekeying message as: \( \alpha_e(t) = K_e(t) + \prod_{i=1}^{n} f(K_i, \mu(t)) \).

65. The specially adapted computer of claim 63, wherein said key information comprises a session key, and one or more personal key encrypting keys.

66. The specially adapted computer of claim 65, wherein said protocol employs a tree-based algorithm to update said session key and said personal key-encrypting key(s).

67. The specially adapted computer of claim 65, wherein said key information is provided using a media-independent channel.

68. The specially adapted computer of claim 65, wherein said key information additionally comprises an embedding key.

69. The specially adapted computer of claim 68, wherein said key information is provided using a media-dependent channel.

70. The specially adapted computer of claim 49, wherein said multimedia data is provided to said group member in multi-layered form.

71. The specially adapted computer of claim 51, wherein said protocol accomplishes the dynamic management of key information in conjunction with the MPEG-4 Intellectual Property Management protocol.

72. The specially adapted computer of claim 51, wherein bits containing key data are embedded in said multimedia data by changing half-pixel motion estimation to an integer-pixel motion vector of said multimedia data.

73. The specially adapted computer of claim 72, wherein two bits of key data \( B_n B_{n+1} \) are embedded in one motion vector of said multimedia data by specifying a set \( S_m \), which a motion vector will belong to using \( m = 2B_n + \).
$B_{n+k}$, wherein the motion vector $(D_x, D_y)$ of one multiblock in frame $k$ is determined by:

$$(D_x, D_y) = \arg \min_{(d_x, d_y) \in S_m} \sum_{i=0}^{15} \sum_{j=0}^{15} | f^k(i, j) - f^{k-1}(i + d_x, j + d_y) |$$

where $(d_x, d_y)$ is a motion vector candidate corresponding to the pixels in set $S_m$.

74. A computer specially adapted by software to be able to function as the recipient of securely transmitted multimedia data, wherein said software permits said specially adapted computer to accomplish said receipt through the analysis of a parametric one-way function and a broadcast seed transmitted from a group center computer or through the analysis of data subjected to modulo analysis by a group center computer using previously sent key information.

75. The specially adapted computer of claim 74, wherein said security is accomplished through the use of a parametric one-way function and a broadcast seed.

76. The specially adapted computer of claim 75, wherein said broadcast seed is a random broadcast seed.

77. The specially adapted computer of claim 76, wherein said broadcast seed is key information previously transmitted to said one or more group members.

78. The specially adapted computer of claim 75, wherein said key information comprises a session key, and one or more personal key encrypting keys.

79. The specially adapted computer of claim 78, wherein said protocol employs a tree-based algorithm to update said session key and said personal key-encrypting key(s).

80. The specially adapted computer of claim 78, wherein said key information is provided using a media-independent channel.
81. The specially adapted computer of claim 78, wherein said key information additionally comprises an embedding key.

82. The specially adapted computer of claim 81, wherein said key information is provided using a media-dependent channel.

83. The specially adapted computer of claim 75, wherein said parametric one-way function $h$ maps a sequence $(x,y)$ of $2B$ bits to $B$ bits.

84. The specially adapted computer of claim 75, wherein said parametric one-way function additionally comprises a function $f(x,y) = 1\| h(x,y)$, wherein $f$ optionally prepends a single 1 bit in front of the output of $h(x,y)$.

86. The specially adapted computer of claim 75, wherein said parametric one-way function is used in calculating a degree $n-2$ polynomial $p(x)$ that interpolates the points $(z_j, w_j)$, where $z_j$ is a quantity that identifies user $j$, and $w_j = K_z(t) + f(K_z, \mu(t)) \pmod{p}$.

87. The specially adapted computer of claim 74, wherein said security is accomplished by subjecting the transmitted data to modulo analysis using previously sent key information.

88. The specially adapted computer of claim 87, wherein in said modulo analysis, said Group Center forms $K_z(t)$ and calculates the rekeying message as: $\alpha_e(t) = K_z(t) + \prod_{n=1}^{n-1} f(K_z, \mu(t))$.

89. The specially adapted computer of claim 87, wherein said key information comprises a session key, and one or more personal key encrypting keys.

90. The specially adapted computer of claim 89, wherein said protocol employs a tree-based algorithm to update said session key and said personal key-encrypting key(s).
91. The specially adapted computer of claim 89, wherein said key information is provided using a media-independent channel.

92. The specially adapted computer of claim 89, wherein said key information additionally comprises an embedding key.

93. The specially adapted computer of claim 89, wherein said key information is provided using a media-dependent channel.

94. The specially adapted computer of claim 74, wherein said multimedia data is provided to said group member in multi-layered form.

95. The specially adapted computer of claim 74, wherein said protocol accomplishes the dynamic management of key information in conjunction with the MPEG-4 Intellectual Property Management protocol.

96. The specially adapted computer of claim 74, wherein bits containing key data are embedded in said multimedia data by changing half-pixel motion estimation to an integer-pixel motion vector of said multimedia data.

97. The specially adapted computer of claim 86, wherein two bits of key data $B_nB_{n+1}$ are embedded in one motion vector of said multimedia data by specifying a set $S_m$ which a motion vector will belong to using $m = 2B_n + B_{n+1}$, wherein the motion vector $(D_x, D_y)$ of one multiblock in frame $k$ is determined by:

$$
(D_x, D_y) = \arg \min_{(d_x, d_y) \in S_m} \sum_{i=0}^{15} \sum_{j=0}^{15} \left| f^k(i, j) - f^{k-1}(i + d_x, j + d_y) \right|
$$

where $(d_x, d_y)$ is a motion vector candidate corresponding to the pixels in set $S_m$. 
Fig. 3A

Invalidate Keys

Joining/Departing
Member
Fig. 4A

<table>
<thead>
<tr>
<th>Operation ID</th>
<th>$B(\alpha)$</th>
<th>$\alpha$</th>
</tr>
</thead>
</table>

Fig. 4B

<table>
<thead>
<tr>
<th>Operation ID</th>
<th>$B(\sigma)$</th>
<th>$\sigma$</th>
<th>$B(\alpha)$</th>
<th>$\alpha$</th>
</tr>
</thead>
</table>
Fig. 7A

Bits

Probability of Occupancy Q

- \(a = 4, L = 6\)
- \(a = 4, L = 8\)
- \(a = 4, L = 10\)

Fig. 7B

Bits

Probability of Occupancy Q

- \(a = 4, L = 6\)
- \(a = 4, L = 8\)
- \(a = 4, L = 10\)
Fig. 13

GC:

$K_s^{\text{new}}$ → $f(K_s^{\text{new}})$ → $\alpha$ → Encryption

Video → H263/MPEG2 Encoder/Embedding → Delay → $K_s$ → Network

Member:

$\alpha$ → $\eta(\alpha)$ → $K_s^{\text{new}}$ → Decryption

Decoded Video → H263/MPEG2 Decoder/Extraction → Delay → $K_s$ → $K_s^{\text{new}}$
Fig. 17

\[ r(x,y) \xrightarrow{\text{Interpolator}} \tilde{f}(x,y) \xrightarrow{\text{Warp}} \tilde{r} \xrightarrow{\text{Resampler}} r_w \xrightarrow{\text{Filter}} f_2(x,y) \xrightarrow{\text{Final}} \hat{r}(x,y) \]
INTERNATIONAL SEARCH REPORT

A. CLASSIFICATION OF SUBJECT MATTER

<table>
<thead>
<tr>
<th>IPC(7)</th>
<th>H04L 9/00</th>
</tr>
</thead>
<tbody>
<tr>
<td>US CL</td>
<td>713/163,192,156</td>
</tr>
</tbody>
</table>

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

| U.S.         | 713/163,192,156 |

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

ALFRED J. MENEZES, HANDBOOK OF APPLIED CRYPTOGRAPHY

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

WEST; EAST

C. DOCUMENTS CONSIDERED TO BE RELEVANT

<table>
<thead>
<tr>
<th>Category</th>
<th>Citation of document, with indication, where appropriate, of the relevant passages</th>
<th>Relevant to claim No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>US 6,049,878 A (CARONNI et al) 11 April 2000</td>
<td>1-6, 26-31, 51-56, 65, 66, 74-79, 89, 90</td>
</tr>
<tr>
<td></td>
<td>Abstract; fig 2-5; col. 3, lines 34-50; col. 4, lines 23-33 and 66-67;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>col. 5, lines 1-12 and 22-33; col. 6, lines 55-59; col. 7, lines 40-67;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>col. 9, lines 25-50; col. 10.</td>
<td></td>
</tr>
</tbody>
</table>

Further documents are listed in the continuation of Box C.

Date of the actual completion of the international search

17 OCTOBER 2001

Date of mailing of the international search report

07 DEC 2001

Name and mailing address of the ISA/US Commissioner of Patents and Trademarks

Box PCT

Authorized officer

Washington, D.C. 20251

ALBERT DECKY

Facsimile No. (703) 305-9230

Telephone No. (703) 305-4169

Form PCT/ISA/210 (second sheet) (July 1998)
<table>
<thead>
<tr>
<th>Category</th>
<th>Citation of document, with indication, where appropriate, of the relevant passages</th>
<th>Relevant to claim No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>US 5,748,736 A (MITTRA) 05 May 1998, col. 4-7, col. 8, lines 5-66; col. 9, lines 1-50; col. 10-14</td>
<td>7-13,15, 17-24,40, 32-38,91, 42-49,92, 57-61,63, 67-72,93, 80-85,87, 94-96</td>
</tr>
</tbody>
</table>