CERIAS Tech Report 2003-42

Protection Of Multicast Scalable Video By Secret Sharing: Simulation Results

by Ahmet M. Eskicioglu and Scott Dexter and Edward J. Delp

Center for Education and Research in
Information Assurance and Security,
Purdue University, West Lafayette, IN 47907-2086

Protection of multicast scalable video by secret sharing: simulation results

Ahmet M. Eskicioglu[*a], Scott Dexter[†a], Edward J. Delp[‡b]
[a]Department of Computer and Information Science, CUNY Brooklyn College
2900 Bedford Avenue, Brooklyn, NY 11210
[b]Video and Image Processing Laboratory (*VIPER*), School of Electrical and Computer Engineering,
Purdue University, West Lafayette, IN 47907

ABSTRACT

Security is an increasingly important attribute for multimedia applications that require prevention of unauthorized access to copyrighted data. Two approaches have been used to protect scalable video content in distribution: Partial encryption and progressive encryption. Partial encryption provides protection for only selected portions of the video. Progressive encryption allows transcoding with simple packet truncation, and eliminates the need to decrypt the video packets at intermediate network nodes with low complexity.

Centralized Key Management with Secret Sharing (CKMSS) is a recent approach in which the group manager assigns unique secret shares to the nodes in the hierarchical key distribution tree. It allows the reconstruction of different keys by communicating different activating shares for the same prepositioned information. Once the group key is established, it is used until a member joins/leaves the multicast group or periodic rekeying occurs. In this paper, we will present simulation results regarding the communication, storage and processing requirements of the CKMSS scheme applied to scalable video. In particular, we have measured the rekey message sizes, storage capacity, and processing times needed by the server for each join/leave request and periodic rekey event.

Keywords: scalable video, video compression, partial video encryption, multimedia, secret sharing, prepositioned secret share, secure multicasting, key graph, hierarchical key distribution tree.

1. INTRODUCTION

With the unprecedented developments in computing and networking technologies, there is a growing demand in digital multimedia services that can be delivered to a variety of devices ranging from low powered PDAs to the fastest personal computers. Efficient coding techniques allow a substantial reduction in the storage or transmission requirements for multimedia data (text, animation, images, audio and video). As video is the most critical component of any multimedia service, several compression standards have been established including H.261, H.263, MPEG-1, MPEG-2 and MPEG-4. To have an idea of the need for video compression, let us consider an example: The CCIR 601 recommendation for digitized NTSC television specifies 480 active lines per picture and 720 active pixels per line. With 30 frames/sec and (4-2-2) subsampling format, the resulting rate exceeds 20 Mbytes:

760 x 480 = 364,800 pixels
30 frames/sec = 10, 944,000 pixels/sec
4-2-2 format (16 bits/pixel) = 175,104,000 bits/sec ≅ 20.9 Mbytes

Today's networks for multimedia content distribution are heterogeneous in nature. The Internet, in particular, is full of uncertainties with a wide range of channel capacities and client device capabilities for display, computation and

[*] eskicioglu@sci.brooklyn.cuny.edu
[†] dexter@sci.brooklyn.cuny.edu
[‡] ace@ecn.purdue.edu

communication. This leads to a set of requirements that cannot be met by traditional video coders that commonly encode and transmit video at a uniform bit rate.

Scalable video compression is the encoding of a single video stream in different layers, each layer with its own bit rate. The availability of multiple substreams of increasing visual quality allows the video codec to adapt to different client capabilities and network conditions. The major types of scalability include:

Spatial scalability: Provides different spatial resolutions. The lowest spatial resolution video is reconstructed from the base layer. The enhancement layers are coded using interpolation and the base layer.

Temporal scalability: Provides different frame rates. The base layer is independently coded to obtain the minimal temporal rate. The enhancement layers are created using temporal prediction with respect to the base layer.

SNR scalability: Provides same spatial resolution but varying video quality. At the base layer, the frequency domain coefficients are coarsely quantized to achieve basic image quality. For the reconstruction of the enhancement layers, finer quantization steps are used.

Region-of-interest scalability: This is a special type of scalability that can provide spatial, temporal or SNR scalability within a particular object or a region of the video sequence. The basic idea is to choose objects or regions of interest to the viewer and offer as many enhancement layers as needed.

The video coding standards MPEG-2, MPEG-4 and H.263+ include scalability modes.

For many multimedia services, security is an essential requirement to prevent unauthorized access to copyrighted content. With the transition from analog to digital technologies, content providers seek assurance for the protection of their intellectual property [1,2]. Presently, encryption appears to be the only tool that can be used alone to provide confidentiality in applications such as video conferencing, Pay TV and on-line video games.

Although robust encryption algorithms exist in the cryptographic literature, their speed is not compatible with the real-time processing requirements of huge amounts of data involved in multimedia services. To reduce the computational complexity of full video encryption, several schemes have been proposed [3,4,5,6,7,8,9]. Two publications evaluate the performance of these proposals and discuss the tradeoffs among several metrics such as security level, encryption speed and compression efficiency [10,11].

For the protection of scalable video, two approaches are base layer protection [12] and progressive encryption [13]. The first paper argues that temporal and SNR scalability are not suitable for partial encryption and chooses spatial scalability as the subject of an experiment. The conclusion of the study is that the protection obtained from simple base layer encryption of a scalable encoded video based on a spatial resolution pyramid is comparable to the best known partial MPEG encryption method. The second paper proposes a method for secure scalable streaming (SSS). SSS encodes video into secure scalable packets that can be streamed to heterogeneous clients through hybrid (wired and wireless) networks. Progressive encryption is characterized by the property that the first portion of the data is encrypted independently while the later portions are encrypted based on earlier portions. This allows transcoding to be performed at intermediate network nodes with simple packet truncation and without video decryption. In the SSS architecture, although the payload is encrypted progressively, the header data is left unencrypted and contains information such as the recommended truncation points within the encrypted packets.

Scalable video codecs partition a video stream into a base layer and multiple enhancement layers. The business model used in a particular multimedia service may require protection of the base layer and any number of the enhancement layers. Several scenarios may be envisaged that dictate the number of layers to be protected. In Pay-Per-View applications, the previews would normally be transmitted without encryption to create a purchase interest. The level of protection for the full movie depends on the access conditions determined by the value of content and other criteria. In general, each layer should be encrypted with a different key if it needs to be accessed independently.

There are three primary ways of delivering a multimedia service to clients: *Unicast*, *broadcast* and *multicast*. Unicasting involves point-to-point communication between a server and a client device, broadcasting requires

transmitting the same data to the entire client population and multicasting is an efficient distribution mechanism from a source to a large group of clients. IP multicast uses the notion of a group of members identified with a given group address. When a message is sent to this group address, the network uses a multicast routing protocol to replicate the message at intermediate nodes and forward copies to the group members. Secure multicast communication [14,15] is achieved by using a group key shared by all the members of the group. The way the group key is generated and delivered to group members is influenced by a number of factors such as the multicast application type and group dynamics. Regardless of the approach used, the important desired attributes for a key management system are forward access control, backward access control and minimal storage, communication and computational requirements.

The literature on multicast security includes many key management schemes. One important group of schemes makes use of hierarchical key distribution trees. The *Centralized Tree-Based Key Management* (CTKM) scheme has been developed in three separate publications in the same time period [16,17,18].

A recent multicast security paper [19] introduces a new approach based on secret sharing in which the group manager assigns unique secret shares to the nodes in the hierarchical key distribution tree. Called the *Centralized Key Management with Secret Sharing (CKMSS)*, it is a prepositioned shared secret scheme that allows the reconstruction of different keys by communicating different activating shares for the same prepositioned information. Once the group key is established, it is used until a member joins/leaves the multicast group. It can also be changed by periodic rekeying if the content value is high. The CKMSS scheme has been extended to the protection of scalable video [20].

In this paper, we will present simulation results regarding the computational, storage and processing requirements of the CKMSS scheme applied to scalable video. Our results measure the storage capacity, the rekey message size and the processing time needed by the server per join/leave request based on the following parameters: the initial group size, the degree of the tree, the size of the share sets assigned to the nodes, and the number of layers of scalable video.

2. SECRET SHARING AND SCALABLE VIDEO

Secret sharing schemes form a particular group of multi-party protocols for key establishment [21]. They provide a reliable mechanism for the protection of cryptographic keys without increased risk of disclosure. They also enable distribution of trust or control in critical activities such as launching of a missile and opening bank vaults.

Definition: A (*t, n*) *threshold scheme* (*t*≤ *n*) is a method that enables a trusted dealer to divide a secret *S* into *n* secret shares S_i, ($1 \le i \le n$) in such a way that at least *t* shares are required to reconstruct *S*. It is assumed that each S_i is securely distributed to user P_i and stored as confidential information. A *perfect* threshold scheme is a threshold scheme in which a knowledge of (*t*-1) or fewer shares does not provide any advantage to the opponent to find the secret.

In Shamir's (*t, n*) threshold scheme [22], the secret *S* is defined to be the coefficient a_0 of a random (*t*-1)-degree polynomial

$$f(x) = (a_{t-1}x^{t-1} + \ldots + a_1 x + a_0) \bmod p$$

over the finite Galois Field GF(*p*). The trusted dealer performs the following tasks to share the secret among *n* users:

1. Choose a prime *p* larger than *n* and the secret *S*.
2. Construct *f(x)* by selecting (*t*-1) random coefficients $a_1, \ldots, a_{t-1}$.
3. Compute the shares S_i by evaluating *f(x)* at *n* distinct points.
4. Securely distribute S_i to user P_i ($1 \le i \le n$).

The secret *S* can be recovered by constructing the polynomial

$$f(x) = \sum_{i=0}^{t-1} y_i \prod_{0 \le j \le t-1, j \ne i} (x - x_j)/(x_i - x_j).$$

from any *t* of the *n* shares, and computing *f*(0).

Shamir's threshold scheme has some desirable properties that will be exploited later:

(i) It is a perfect threshold scheme.
(ii) The size of each share does not exceed the size of the secret *S*.
(iii) Different levels of control can be created by assigning different number of shares to users.
(iv) The security does not rely on unproven mathematical assumptions.

In a (*t, n*) prepositioned secret sharing scheme, the *n* secret shares are stored by the participants in advance of the activation of the scheme [23,24]. Even if all of the *n* pieces are exposed, the secret key cannot be recovered until some additional information is provided. In our implementation, we will have $n = t$-1, i.e., the scheme is designed to allow the recovery of the secret by requiring only one more piece (the "activating" share).

The process of encrypting three layers of a multicast scalable video is depicted in Figure 1. Each layer is encrypted with a different symmetric key. The members of the multicast group will need three keys for decrypting the entire video stream. These keys will be renewed after each join or leave operation and periodic rekeying. The alternatives for generating multiple group keys are presented in [20].

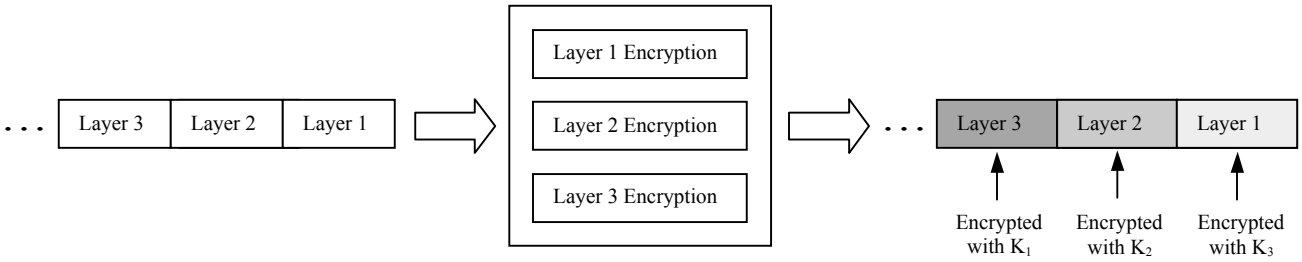

Figure 1. Encryption of scalable video

The CKMSS scheme is in the class of *hierarchical tree based schemes* where a single entity (i.e., the group manager) controls the entire group. The group manager creates and stores a k-ary tree structure with a unique set of shares assigned to each node. The *height h* of the tree is the length of the longest directed path in the tree, and the *degree d* of the tree is the maximum number of incoming edges of a node in the tree. The *n* leaves of the tree contain the *n* symmetric keys the manager has established with the members of the group. Each member keeps a subset of the manager's set of share sets. The elements of the set owned by a member are those found along the directed path from the member to the root of the tree, including the leaf set and the root set. The group manager encrypts the multicast data with the group key. The members generate the group key using their own set of shares at the root node and the activating share sent by the group manager. All the other sets in the tree are auxiliary sets required for efficient set updates.

For scalable video, the two alternative proposals in [20] allow the encryption of different layers with different keys but all members have access to all keys. Use of multiple keys undoubtedly increases the cryptanalytic strength of the scheme but does not help create new business models. We now extend the original CKMSS scheme so that each layer can be accessed independently of the others:

JOIN: In response to a join request from a potential member, the group manager engages in a mutual authentication protocol. If the request is accepted, the manager establishes an individual share for the member and unicasts a protected message. In this message, additional shares (based on the new member's purchase order) are inserted; the individual share will correspond to the base layer and the others to the enhancement layers.

LEAVE: As the leaving member still has the shares for the enhancement layers, he will be able to decrypt those layers. This is not a major business concern, however, since the most relevant information in terms of image perception is concentrated in the base layer. The enhancement layers have incremental contribution to video quality. Experiments show that a reconstructed frame with an undecodable base layer has no commercial value

[12]. In an alternative architecture, the members use a tamper-proof device (e.g., a smart card) which can be programmed to automatically delete the shares of the leaving member. In real-life applications, such a device will be needed to handle the purchase transactions and other sensitive data [2].

PERIODIC REKEY: As in the original scheme, only an activating share is multicast. The members use this share and their own layer shares to construct the layer keys.

Example: Figure 2 shows a simple 4-ary tree with 16 members. Each node is assigned a set of shares. Depending on the type of the operation (join, leave or periodic rekey), some of these shares will change. We will consider three operations to explain the events in the simulation. In this example and in the rest of the paper, we will study the *group-oriented strategy* for the construction of rekey messages and their secure distribution to members. This type of strategy allows the group manager (GM) to construct a single message that contains all the new sets (except the sets for the joining members, which are unicast). We will assume a scalable video with 3 layers, each layer encrypted with a different key. Let $s_{1\text{-}16}$ be the share needed for the protection of the base layer. For each of the enhancement layers, the member purchases an additional share $s_{1-16}^{(i)}$, i=1,2.

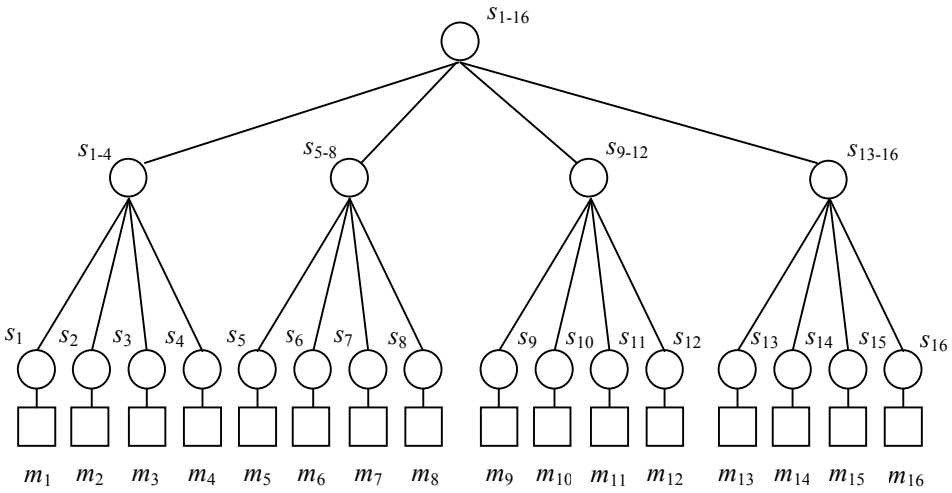

Figure 2. Hierarchical tree for secret sharing (tree degree = 4)

(a) Member 16 joins the tree

The GM performs the following operations:

(i) Label the joining member 16.
(ii) Establish s_{16} with the member, create a new member node and a new set node, and attach the set node to the existing "joining point."
(iii) Change $s_{1\text{-}15}$ to $s_{1\text{-}16}$ and $s_{13\text{-}15}$ to $s_{13\text{-}16}$.
(iv) Construct and send the below two messages (The first is multicast to members 1-15, the second is unicast to member 16):

$GM \rightarrow \{m_1, \ldots, m_{15}\}$: AS, $\{s_{1-16}\}_{k_{1-15}}$, $\{s_{13-16}\}_{k_{13-15}}$

$GM \rightarrow m_{16}$: AS, $\{s_{1-16}, s_{1-16}^{(1)}, s_{1-16}^{(2)}, s_{13-16}\}_{k_{16}}$

In the above messages, AS is the activating share, and the fresh keys $k_{1\text{-}15}$, $k_{13\text{-}15}$ and k_{16} are obtained using the activating share and the sets $s_{1\text{-}15}$, $s_{13\text{-}15}$ and s_{16}, respectively. Depending on their access rights, the members construct a subset of

the next set of group keys $k_{1-16}, k_{1-16}^{(1)}, k_{1-16}^{(2)}$ when they receive the new activating share together with the encrypted multimedia data.

(b) Member 16 leaves the tree

The GM performs the following operations:

(i) Delete both the member (leaf) node and the set node for member 16.
(ii) Replace $s_{13\text{-}16}$ at the "leaving point" by $s_{13\text{-}15}$ and $s_{1\text{-}16}$ by $s_{1\text{-}15}$.
(iii) Construct and multicast the below message to the remaining fifteen members:

$GM \rightarrow \{m_1, \ldots, m_{15}\}$: AS, L_0 , L_1, where

L_0: $\{s_{1-15}\}_{k_{1-4}}$, $\{s_{1-15}\}_{k_{5-8}}$, $\{s_{1-15}\}_{k_{9-12}}$, $\{s_{1-15}\}_{k_{13-15}}$ and L_1: $\{s_{13-15}\}_{k_{13}}$, $\{s_{13-15}\}_{k_{14}}$, $\{s_{13-15}\}_{k_{15}}$.

In the above message, AS denote the activating share. The fresh keys $k_{1\text{-}4}$, $k_{5\text{-}8}$, $k_{9\text{-}12}$, $k_{13\text{-}15}$, k_{13}, k_{14} and k_{15} are obtained using the activating share and the sets $s_{1\text{-}4}$, $s_{5\text{-}8}$, $s_{9\text{-}12}$, $s_{13\text{-}15}$, s_{13}, s_{14} and s_{15}, respectively. Depending on their access rights, the members construct a subset of the next set of group keys $k_{1-15}, k_{1-15}^{(1)}, k_{1-15}^{(2)}$ when they receive the new activating share together with the encrypted multimedia data.

(c) Periodic key change

The GM performs the following operation:

(i) Construct and multicast the below message to the entire group.

$GM \rightarrow \{m_1, \ldots, m_{16}\}$: AS,

where AS is the activating share needed to reconstruct any subset of the group keys $k_{1-16}, k_{1-16}^{(1)}, k_{1-16}^{(2)}$.

3. SIMULATION

In this paper, we will present simulation results regarding the computational, storage and processing requirements of the CKMSS scheme applied to scalable video. In particular, we have:

1. Explored a particular algorithm for generating multiple keys. The root shares are used to define *n* disjoint subsets, one subset for each key. As each node can be assigned a different number of shares, this is a convenient arrangement.
2. Measured the storage capacity, the rekey message sizes, and the processing times needed by the server per join/leave request based on the following parameters: the degree of the tree, the initial group size, the size of the share sets assigned to the nodes, and the number of layers of scalable video.

We performed a number of experiments to evaluate the performance of the CKMSS scheme. The simulation code was developed in C++ and run on a single user Windows environment using an Intel Pentium 4 processor with a speed of 1.8GHz. A public-domain implementation of the AES cipher by Szymon Stefanek [25] and a secret reconstruction routine by Baltimore Technologies [26] were used in the simulation.

In Shamir's scheme, we defined the shares to have the format $(x,y)=(i,S_i)$, where i is not a public index but a part of the secret share. Both of the coordinates x and y are 4-byte values, making the total size of a share 8 bytes.

The four design parameters are:

1. Degree of the set tree
2. Initial group size
3. Number of shares assigned to each node
4. Number of layers of scalable video

Each of these four parameters was allowed to vary as the others were kept constant. Table 1 shows the range of values and the constants for the parameters. Both the average processing times and the average message sizes were measured. Processing a request involves first locating the leaf corresponding to the joining/leaving member, then traversing the path from this leaf to the root. At each step, the activating share is combined with the shares stored at the node to produce an encryption key; this key is in turn used to encrypt newly generated random shares. The contents of the nodes are also updated to contain these new random shares.

Parameters	**Range of values**	**Constants**
Degree of the set tree	2, 3, 4, …, 20	4
Initial group size	2^5, 2^6, 2^7, …, 2^{14}	16384
Number of shares per node	1, 2, 3, …, 10	2
Number of layers of scalable video	1, 2, 3, …, 10	1

Table 1. Parameter values

In the experiments, the group manager first builds the initial tree using the given number of join requests. It then updates the tree in response to the subsequent join/leave requests. These requests are generated randomly according to the ratio 1:1.

In a tree of degree d and height h, the number of nodes is $d^0 + d^1 + ... + d^h$. This is a geometric progression which sums up to $\{(1-d^{(h+1)})/(1-d)\}$. Since $n = d^h$ for a full and balanced tree, the number of nodes in the tree is expected to be around $\{(nd-1)/(d-1)\}$ after each tree update.

Degree of the set tree

Table 2 and Figure 3 show how the processing time and the message size vary with the set tree degree. One important observation from this result is the optimal degree of the tree which is 4. This confirms the result obtained in [17]. Note that as the degree of the tree is increased, the processing time per join goes down and the processing time per leave goes up (at a higher rate). When a new member is accepted to the group, all the keys along the path from the joining point to the root node have to change (backward access control). The server's encryption cost for the join operation is $2(h-1)$; this implies that the cost is decreased as the tree degree is increased (which effectively decreases the tree height) [19]. When a member leaves the group, all the keys along the path from the leaving point to the root node have to change (forward access control). The server's encryption cost for the leave operation is $d(h-1)$; this implies that the cost is increased with an increase in tree degree [19].

Average rekey message size																			
Message type\ Tree degree	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
Join multicast	448	283	224	193	179	159	156	149	142	133	127	125	124	122	121	120	118	115	111
Leave multicast	863	819	863	933	1042	1087	1219	1308	1381	1419	1496	1601	1710	1809	1914	2013	2083	2159	2203
Unicast	244	160	128	112	107	96	94	90	87	83	79	79	78	77	77	76	75	73	71

Table 2. Message size versus set tree degree

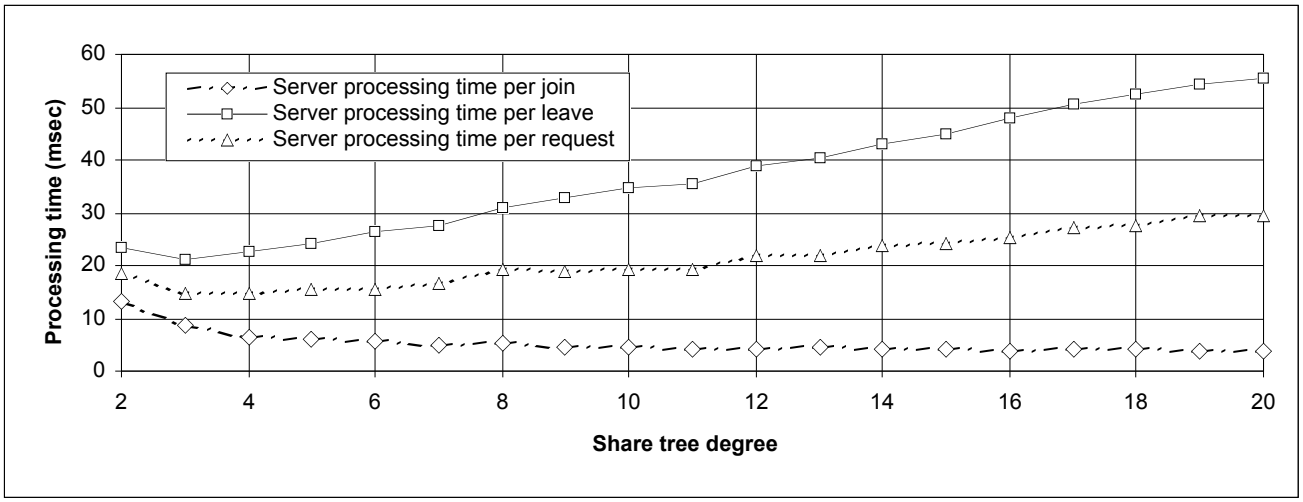

Figure 3. Processing time versus set tree degree

Initial group size

Table 3 and Figure 4 show how the processing time and the message size vary with the initial group size. The horizontal axis in Figure 4 is in log scale. When extrapolated, this implies that the CKMSS scheme is scalable to large groups because the processing time per request increases almost linearly with the logarithm of the group size.

Average rekey message size										
Message type\ Initial group size	**32**	**64**	**128**	**256**	**512**	**1024**	**2048**	**4096**	**8192**	**16384**
Join multicast	89	104	119	130	151	160	181	192	213	224
Leave multicast	294	365	438	479	565	605	693	735	820	863
Unicast	60	69	76	82	93	96	107	112	122	129

Table 3. Message size versus initial group size

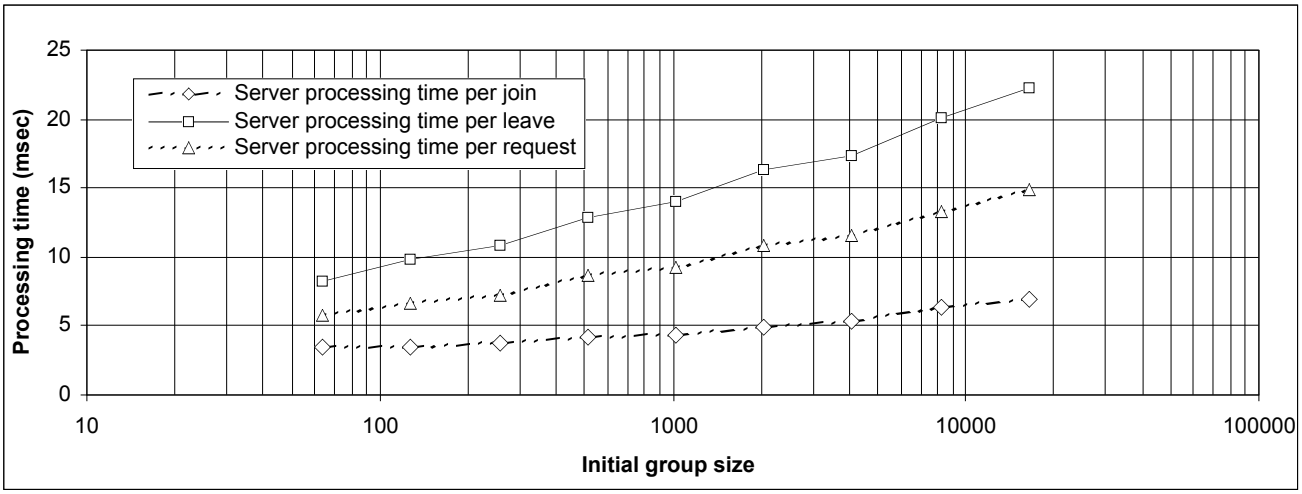

Figure 4. Processing time versus initial group size

Number of shares per node

Table 4 and Figure 5 show how the processing time and the message size vary with the number of shares per node. The CKMSS scheme deviates from linear behavior in this case. The nonlinear cost, however, is justifiable as it results in increased security levels.

In Table 4, the "pairing" of multicast message sizes for 2i and 2i+1 (i=2,3,…,10) shares is due to the characteristics of AES encryption, which generates ciphertext in 16-byte blocks. Because the shares (i.e., the plaintext) are 8-byte quantities, the ciphertexts corresponding to 2i and 2i+1 shares will have the same length.

Average rekey message size										
Message type\ # of shares	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Join multicast	112	224	224	336	337	450	451	562	560	672
Leave multicast	432	863	863	1295	1296	1727	1727	2160	2159	2591
Unicast	65	128	177	242	291	357	405	469	513	577

Table 4. Message size versus number of shares per node

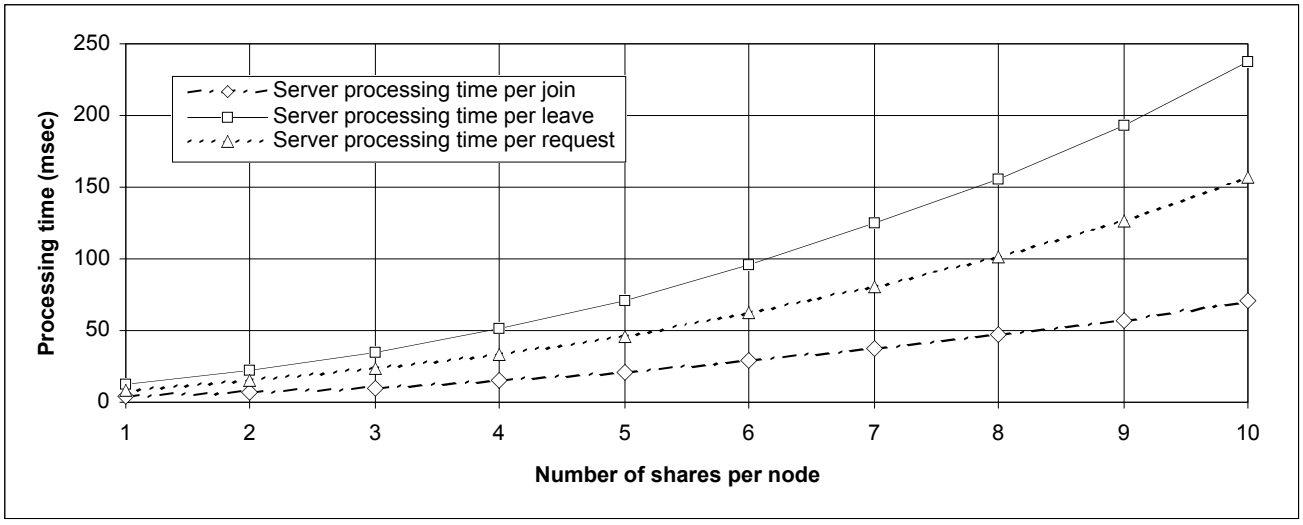

Figure 5. Processing time versus number of shares per node

Number of layers of scalable video

The additional cost for scalable video is a small increase in the size of the unicast message when a member joins the group. If the video has *n* layers, *n* shares are unicast to the joining member. If we consider the fact that the highest number of layers in a commercial application would be expressed by a small integer number, the increase in the unicast message is negligible.

Finally, for periodic rekeying, the workload for the server is minimal. Only the activating share is multicast. There is no encryption cost and the message size is constant at 8 bytes. This is in contrast with the CTKM scheme where, for a tree of degree *d*, *d* encrypted messages need to be multicast (The total size of the message depends on the size of the cipher key).

4. CONCLUSIONS

We have presented simulation results that show the behavior of the CKMSS scheme. The behavior can be summarized as follows:

The proposed algorithm for scalable video is convenient and introduces insignificant computational and communication overhead for any number of layers. A member's join request also includes the number of video layers to be accessed. The group manager, in turn, unicasts the shares corresponding to these layers. A minor caveat is that each leaving member will still have access to the enhancement layers. Experimental results show that protection of only the base layer can provide good security as the enhancement layers add incremental value to video quality. If, however, a tamper-proof security device is used, the shares can be deleted as the member leaves the group.

The optimal tree degree is 4. The processing time is about the same in the neighborhood of 4 (for 3 and 5) and gradually increases for higher degrees.

The processing time per request increases almost linearly with the logarithm of the group size. The highest population for a group tested was 16384. Nevertheless, there is substantial evidence that with extrapolation, the server cost is reduced from $O(n)$ to $O(\log(n))$ for much larger populations.

The computational cost is mildly nonlinear as we increase the number of shares per node. Depending on the security level needed for a given application, this cost can be justified.

Leave operations dominate over the join operations. Therefore, both the computational and communication loads are higher to process the leave requests.

For periodic rekeying, the multicast message size is constant irrespective of the size of the group. Furthermore, the server does not need to encrypt the message.

A natural extension of this work is to investigate the storage, computational and communication requirements for the group members. Other areas of application of the CKMSS scheme is a current area of research.

ACKNOWLEDGMENTS

We would like to thank Sajjad Ahmed, Kirk Hylton, Kevin Lewis, Sheng Li and Yevgeniy Tsekhanskiy for their help in testing several software components and contributing to this work. Our thanks also go to Dr. Gerald Weiss for a very useful discussion on the data structures used in the simulation code.

REFERENCES

[1] A. M. Eskicioglu and E. J. Delp, "Overview of Multimedia Content Protection in Consumer Electronics Devices," *Signal Processing: Image Communication*, **16(5)**, pp. 681-699, April 2001.

[2] A. M. Eskicioglu, J. Town and E. J. Delp, "Security of Digital Entertainment Content from Creation to Consumption," will appear in a special issue of *Signal Processing: Image Communication* in March 2003.

[3] T. B. Maples and G. A. Spanos, "Performance Study of a Selective Encryption Scheme for the Security of Networked, Real-time Video," *Proceedings of 4th International Conference on Computer Communications and Networks*, Las Vegas, NV, September 20-23, 1995.

[4] L. Tang, "Methods for Encrypting and Decrypting MPEG Video Data Efficiently," *Proceedings of the 4th ACM International Multimedia Conference*, pp. 219-230, Boston, MA, November 18-22, 1996.

[5] L. Qiao and K. Nahrstedt, "A New Algorithm for MPEG Video Encryption," *Proceedings of the 1st International Conference on Imaging Science, Systems and Technology*, pp. 21-29, Las Vegas, NV, June 30 - July 3, 1997.

[6] T. Kunkelmann and R. Reineman, "A Scalable Security Architecture, for Multimedia Communication Standards," *Proceedings of the 4th IEEE International Conference on Multimedia Computing and Systems*, Ottawa, Canada, June 3-6, 1997.

[7] C. Shi and B. Bhargava, "A Fast MPEG Video Encryption Algorithm," *Proceedings of the 6th International Multimedia Conference*, Bristol, UK, September 12-16, 1998.

[8] C.-P. Wu and C. -C. Jay Kuo, "Efficient Multimedia Encryption via Entropy Codec Design," *IS&T/SPIE 13th Annual Symposium on Electronic Imaging, Proceedings of SPIE, Vol. 4314*, San Jose, CA, January 2001.

[9] W. Zeng and S. Lei, "Efficient Frequency Domain Selective Scrambling of Digital Video," *IEEE Transactions on Multimedia*, 2002.

[10] I. Agi and L. Long, "An Empirical Study of Secure MPEG Video Transmissions," *Proceedings of the Internet Society Symposium on Network and Distributed System Security*," pp. 137-144, San Diego, CA, February 22-23, 1996.

[11] L. Qiao and K. Nahrstedt, "Comparison of MPEG Encryption Algorithms," *International Journal on Computer and Graphics, Special Issue on Data Security in Image Communication and Network*, **22(3)**, 1998.

[12] T. Kunkelmann and U. Horn, "Partial Video Encryption Based on Scalable Coding," *5th International Workshop on Systems, Signals and Image Processing*, Zagreb, Croatia, June 1998.

[13] S. J. Wee and J. G. Apostolopoulos, "Secure Scalable Video Streaming for Wireless Networks," *IEEE International Conference on Acoustics, Speech and Signal Processing*, Salt Lake City, UT, May 7-11, 2001.

[14] T. Hardjono and G. Tsudik, IP Multicast Security: Issues and Directions, *Annales de Telecom*, July-August 2000, pp. 324-334.

[15] A. M. Eskicioglu, "Multimedia Security in Group Communications: Recent Progress in Wired and Wireless Networks," *Proceedings of the IASTED International Conference on Communications and Computer Network*s, pp. 125-133, Cambridge, MA, November 4-6, 2002.

[16] D. Wallner, E. Harder and R. Agee, "Key Management for Multicast: Issues and Architectures" RFC 2627, June 1999.

[17] C. K. Wong, M. G. Gouda and S. S. Lam, "Secure Group Communications Using Key Graphs," Department of Computer Sciences, The University of Texas at Austin, Technical Report TR-97-23, July 1997.

[18] G. Caronni, M. Waldvogel, D. Sun and B. Plattner, "Efficient Security for Large and Dynamic Groups," Technical Report No. 41, Computer Engineering and Networks Laboratory, Swiss Federal Institute of Technology, February 1998.

[19] A. M. Eskicioglu and M. R. Eskicioglu, "Multicast Security Using Key Graphs and Secret Sharing," *Proceedings of the Joint International Conference on Wireless LANs and Home Networks (ICWLHN 2002) and Networking (ICN 2002)*, pp. 228-241, Atlanta, GA, August 26-29, 2002.

[20] A. M. Eskicioglu and E. J. Delp, "An Integrated Approach to Encrypting Scalable Video," *Proceedings of the 2002 IEEE International Conference on Multimedia and Expo*, Lausanne, Switzerland, pp. 573-576, August 26-29, 2002.

[21] J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1997.

[22] A. Shamir, "How to share a secret," *CACM*, **22(11)**, November, pp. 612-613, 1979.

[23] G. J. Simmons, "How to (really) share a secret," *Advances in Cryptology – CRYPTO '88 Proceedings*, pp. 390-448, Springer-Verlag, 1990.

[24] G. J. Simmons, "Prepositioned shared secret and/or shared control schemes," *Advances in Cryptology – EUROCRYPT '89 Proceedings*, pp. 436-467, Springer-Verlag, 1990.

[25] http://www.esat.kuleuven.ac.be/~rijmen/rijndael/

[26] http://www.baltimore.ie/