PAPER Special Section on Recent Advances in Multimedia Signal Processing Techniques and Applications

Layered Multicast Encryption of Motion JPEG2000 Code Streams for Flexible Access Control

Takayuki NAKACHI[†a)], **Kan TOYOSHIMA**[†], **Yoshihide TONOMURA**[†], ***and*** **Tatsuya FUJII**[†], ***Members***

SUMMARY In this paper, we propose a layered multicast encryption scheme that provides flexible access control to motion JPEG2000 code streams. JPEG2000 generates layered code streams and offers flexible scalability in characteristics such as resolution and SNR. The layered multicast encryption proposal allows a sender to multicast the encrypted JPEG2000 code streams such that only designated groups of users can decrypt the layered code streams. While keeping the layering functionality, the proposed method offers useful properties such as 1) video quality control using only one private key, 2) guaranteed security, and 3) low computational complexity comparable to conventional non-layered encryption. Simulation results show the usefulness of the proposed method.

key words: *layered multicast, broadcast encryption, Motion JPEG2000*

1. Introduction

The rapid spread of the high-speed IP network infrastructure has stimulated the development of applications that use high quality video communications. Multicast delivery will be strongly demanded since the number of receivers will dramatically increase. To ensure that all destinations receive the same digital data, the multicast network uses the IP multicast protocol.

Considering commercial use, strong security functions must be supported to prevent the leakage of streams to invalid receivers. In the multicast network, broadcast encryption [1]–[5] is useful for indicating the intended destinations. Broadcast encryption makes a public key for the source and individual private keys for each destination. Only the destinations belong to the subset nominated by the source can decipher the stream by their individual private keys.

Here, we consider the layered multicast encryption of Motion JPEG2000 code streams [6], which support the distribution of different quality videos to multiple multicast groups. Motion JPEG2000 generates hierarchical code streams and has flexible scalability, such as resolution or SNR. JPEG2000 code streams are protected by the JPEG2000 security (JPSEC) standard [7]. Many secure methods have been proposed for supporting JPSEC [8]–[11]. However, most of these methods consider only the unicast environment. A simple way to achieve layered multicast encryption is to apply broadcast encryption to each layer. However, this demands several private keys to control video quality and so is not practical.

Manuscript received May 1, 2011.
Manuscript revised August 31, 2011.

[†]The authors are with NTT Network Innovation Laboratories, NTT Corporation, Yokosuka-shi, 239–0847 Japan.
a) E-mail: nakachi.takayuki@lab.ntt.co.jp

DOI: 10.1587/transinf.E95.D.1301

In this paper, we propose a layered multicast encryption scheme by combining common key encryption and broadcast encryption. For the new scheme, we clarify the issues raised by the combination. We offer our solutions and evaluate their effectiveness for practical use. Specifically, we discuss 1) the security issues of the hybrid scheme, and 2) its complexity in terms of key size and header size. An experiment demonstrates the validity and usefulness of the proposed scheme in terms of computational complexity, and key and header size. As a result, while keeping the layered functionality, the proposed method offers useful properties such as 1) control of video quality with just one private key, 2) guaranteed security, and 3) low computational complexity comparable to conventional non-layered encryption.

The paper is organized as follows. Section 2 overviews the JPEG2000 core coding system and JPSEC standard. In Sect. 3, we introduce the layered multicast encryption scheme. Its complexity and security are discussed in Sect. 4. Section 5 shows the results of simulations. Conclusions are given in Sect. 6.

2. JPEG2000 Overview

In this section, we briefly outline the JPEG2000 core coding system (Part1) [6] and JPSEC: Secure JPEG2000 (Part8) [7].

2.1 JPEG2000 Core Coding System

JPEG2000 Part1 is the baseline compression standard. Its encoding procedure is shown in Fig. 1. First, input images are decomposed into several subbands by applying Discrete Wavelet Transform (DWT). The lifting-based discrete wavelet transform is adopted in JPEG2000. Two-dimensional transformation is done by applying a one-dimensional wavelet transform to the rows and columns of each image. The wavelet coefficients are then quantized. The quantized coefficients are coded by the Embed-

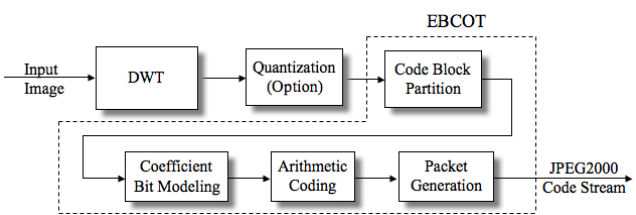

Fig. 1 JPEG2000 encoder.

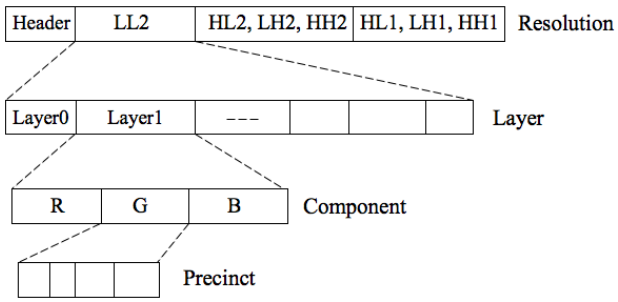

Fig. 2 RLCP structure.

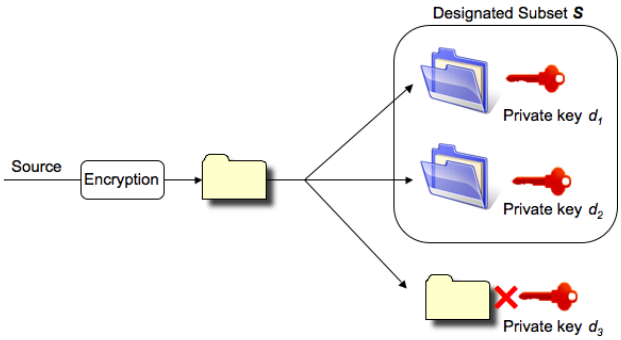

Fig. 3 Broadcast encryption.

ded Block Coding with Optimal Truncation (EBCOT) algorithm.

The EBCOT algorithm realizes various levels of scalability. There are four basic scalability dimensions in a JPEG2000 code stream: resolution (R), quality layer (L), precinct (spatial location) (P) and component (C). Different scalability levels are achieved by ordering packets within the code stream. Figure 2 shows an example of RLCP order structure.

2.2 JPSEC

The JPSEC standard is an open framework, and can be extended to support additional security services and security tools as needed. It focuses on the following media security services: 1) confidentiality, 2) integrity verification 3) source authentication, 4) conditional access, 5) secure scalable streaming and secure transcoding, and 6) registered content identification. In order to secure an image, it applies one or more JPSEC protection tools (e.g. encryption, digital signature). The resulting JPSEC code stream is generated by inserting the corresponding JPSEC syntax in the stream.

Many secure methods [8]–[11] have been proposed that support JPSEC. However, most of them suit only unicast transmission.

3. Layered Multicast Encryption for Flexible Access Control

Multicast transmission is essential since the number of receivers will dramatically increase. Unfortunately, multicast transmission makes all destinations receive the same content at a uniform rate (i.e. uniform quality). Broadband encryption suits the multicasting of such encrypted content [12]. Here, we consider a layered multicast encryption scheme that can distribute the same content to multiple multicast groups where there are as many groups as there are layers.

3.1 Broadcast Encryption

Broadcast encryption [1]–[5] has several applications including pay-TV systems and DVD content protection. It involves a broadcaster and n receivers. Each receiver is given a unique private key. The broadcaster has a broadcaster key. The broadcaster wishes to broadcast message M to a designated set $S \subseteq \{1, \ldots, n\}$ of receivers. All receivers in S should be able to decrypt the broadcast message using only its private key while receivers outside S should not be able to do so even if they collude. The concept of broadcast encryption is shown in Fig. 3.

Broadcast encryption schemes were first formally studied by Fiat and Naor [1]. In addition, a broadcast encryption code with practical cipher text size was proposed by Boneh, Gentry, and Waters in 2005 [2]. We call this encryption scheme BGW. BGW yields compact ciphertexts even if the number of receivers is high. Previous broadcast encryption methods made ciphertext size proportional to the number of receivers. Since the emergence of BGW, broadcast encryption has been attracting a lot of attention for practical use.

The procedure of the BGW method is shown below:

1. **Setup**(n): Broadcast encryption generates n private keys $d_1, \ldots, d_n$ and a public key PK.
2. **Encryption**(S, PK): Takes as input a subset $S \subseteq \{1, \ldots, n\}$, and a public key PK. Outputs the pair {Hdr, K}. Where $K \in \mathcal{K}$ is the message encryption key chosen from finite key set $\mathcal{K}$ (in detail, see Appendix A) and Hdr is called the header. Common key CK is a message that can be deciphered only by the receivers in S. Let CM be the encryption of CK under message encryption key K. The broadcast consists of {S, Hdr, CM}.
3. **Decryption**(S, i, d_i, Hdr, PK): Takes as input a subset $S \subseteq \{1, \ldots, n\}$, user id $i \in \{1, \ldots, n\}$ and private key d_i for user i, header Hdr, and the public key PK. If $i \in S$, the algorithm outputs message encryption key $K \in \mathcal{K}$. Intuitively, user i can then use K to decrypt CM and obtain common key CK.

3.2 Layered Multicast Encryption

In multicast transmission, code streams generated in real-time are transmitted at a uniform rate to all receivers in the network. Broadband encryption suits the multicasting of encrypted video with uniform quality/resolution.

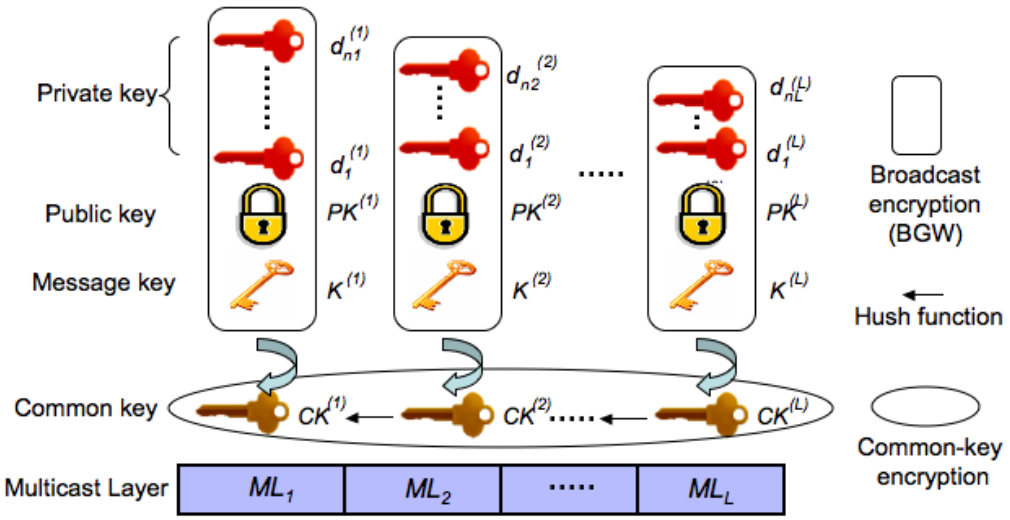

Fig. 5 Proposed layered multicast encryption.

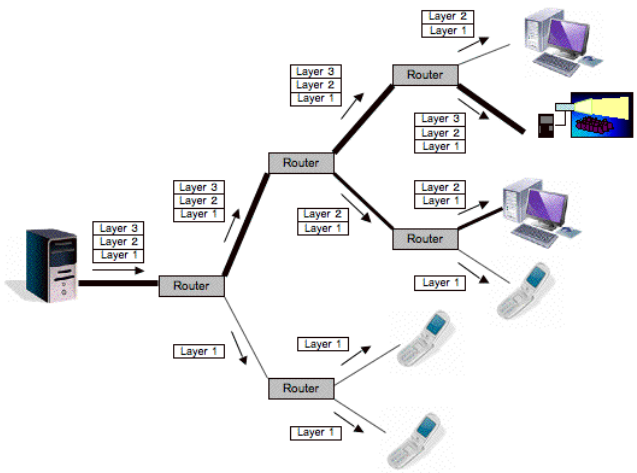

Fig. 4 Layered multicast transmission.

Here, we consider a layered multicast transmission scheme that can distribute different quality videos (same original content) to multiple multicast groups, as shown in Fig. 4. A simple way to achieve layered multicast encryption is to apply multicast encryption to each layer. However, this demands several private keys to control video quality. To access the layered code streams, the user must interact with an on-line key server to obtain the many private keys needed. Therefore, key management becomes complicated.

In order to overcome this problem, we develop a layered encryption scheme. The proposed scheme needs just one private key, which is distributed to the user, to control video quality. We also discuss the security and computational complexity of the proposed method as compared against the conventional non-layered encryption scheme.

3.3 Proposed Layered Multicast Encryption

Our layered multicast encryption scheme needs only one private key to control video quality. Figure 5 illustrates the principle of the proposed layered multicast encryption scheme. It is a hybrid of a common key scheme and a broadcast encryption scheme. We use the BGW method as the broadcast encryption scheme for distributing the common key for JPEG2000 code streams. We assume that JPEG2000 code streams that offer L multicast layers ML_j $(j = 1, \ldots, L)$. The total number of receivers $n = n_1 + n_2 + \ldots + n_L$, where n_j is the number of receivers that can access the multicast layers $ML_1, \ldots, ML_j$. $S^{(j)} \subseteq \{1^{(j)}, \ldots, n^{(j)}\}$ is the subset of receivers permitted to access the multicast layers $ML_1, \ldots, ML_j$.

The progression order of the JPEG2000 code streams is decided in advance. Different scalability levels are achieved by ordering packets and can be assigned to different multicast groups. For example, when the progression order is set to RLCP as shown in Fig. 2, $ML_1 = LL_2$, $ML_2 = \{HL_2, LH_2, HH_2\}$, $ML_3 = \{HL_1, LH_1, HH_1\}$.

The procedure of the proposed layered multicast encryption scheme are shown below:

1. **[Key server] Setup of common key**: Common key encryption, such as AES [13], is used to encrypt the code stream of multicast layer ML_j using common key $CK^{(j)}$. The common key of each layer is calculated from master key $CK^{(L)}$ using the following equation:

$$CK^{(j-1)} = H(CK^{(j)}), \quad j = L, \ldots, 2$$

   where $H(\cdot)$ is a one way hash function (e.g. [14]).

2. **[Key server] Setup of broadcast encryption**: Broadcast encryption generates n_j private keys $d_1^{(j)}, \ldots, d_{n_j}^{(j)}$ and a public key $PK^{(j)}$ for $j = 1, \ldots, L$. Independent private key sets and corresponding public keys are generated for each layer j (for details, see Sect. 4.1).

3. **[Key server] Common key encryption**: Takes as input subset $S^{(j)} \subseteq \{1^{(j)}, \ldots, n^{(j)}\}$, and a public key $PK^{(j)}$. Outputs the pair $\{\mathrm{Hdr}^{(j)}, K^{(j)}\}$, where $K^{(j)} \in \mathcal{K}^{(j)}$ is the message encryption key chosen from finite key set

$\mathcal{K}^{(j)}$ (by setting different random sets (g, t, α), independent key sets $\mathcal{K}^{(j)}$ can be generated, for details, see Appendix A) and $\mathrm{Hdr}^{(j)}$ is called the header. Common key $CK^{(j)}$ is a message that can be deciphered only by the receivers in $S^{(j)}$. Let $CM^{(j)}$ be the encryption of $CK^{(j)}$ under message encryption key $K^{(j)}$. The broadcast consists of $\{S^{(j)}, \mathrm{Hdr}^{(j)}, CM^{(j)}\}$. Step 3 is carried out for $j = 1, \ldots, L$.

4. **[Receiver] Common key decryption**: Takes as input a subset $S^{(j)} \subseteq \{1^{(j)}, \ldots, n^{(j)}\}$, user id $i \in \{1^{(j)}, \ldots, n^{(j)}\}$ and private key $d_i^{(j)}$ for user i, header $\mathrm{Hdr}^{(j)}$, and the public key $PK^{(j)}$. If $i \in S^{(j)}$, the algorithm outputs message encryption key $K^{(j)} \in \mathcal{K}^{(j)}$. Intuitively, user i can then use $K^{(j)}$ to decrypt $CM^{(j)}$ and obtain common key $CK^{(j)}$. Next, user i calculates $CK^{(1)}, \ldots, CK^{(j-1)}$ by using the one way hash function $H(\cdot)$ and can access multicast layers $ML_1, \ldots, ML_j$.
5. **[Content server] Layered multicast transmission:** Encrypts JPEG2000 code stream$^{(j)}$ by using $CK^{(j)}$ and makes each encrypted JPEG2000 code stream$^{(j)}$ for multicast layer ML_j for $j = 1, \ldots, L$. Multicasts code streams to all receivers.
6. **[Receiver] Decoding of code stream:** Receives encrypted JPEG2000 code streams. Decrypts each code stream by using obtained $CK^{(j)}$. Decodes original JPEG2000 code streams.

The proposed layered multicast encryption scheme can be implemented within the JPSEC framework.

In order to clarity the difference from the conventional schemes, the procedure of the proposed method and the conventional schemes are summarized in Tables A·1, A·2 and A·3 (shown in Appendix C).

4. Security and Complexity Considerations

In this section, we point out the issues raised by the hybrid scheme and offer solutions. Specifically, we discuss 1) the security issues of the hybrid scheme, and 2) its complexity in terms of key size and header size.

4.1 Security

We discuss here the security of the proposed method from two points of view. One is the degree of the security reduction created by using multiple (i.e. the number of L which is the number of multicast layers) BGW methods. The other is the probability of private key congruence.

A. Multiple BGW methods

We define $p(n_i)$ as the risk probability when the broadcast encryption scheme is applied to n_i receivers at multicast layer i. According to the BGW method [2], individual risk probability $p(n_i)$ can be assumed to be almost zero:

$$p(n_i) \simeq 0 \tag{1}$$

This is argued on the complexity assumption called the bilinear ℓ-Diffie-Hellman Exponent assumption (ℓ-BDHE) (for details, see Appendix B).

Conversely, $1 - p(n_i)$ is defined as the safe probability. The total safe probability $1 - P$ of the proposed method is the product of each safety probability $1 - p(n_i)$. Thus the total risk probability, P, is given by the following equation:

$$\begin{aligned} P &= 1 - \{1 - p(n_1)\}\{1 - p(n_2)\}, \cdots, \{1 - p(n_L)\} \\ &= 1 - \prod_{i=1}^{L} \{1 - p(n_i)\} \end{aligned} \tag{2}$$

From Eqs. (1)–(2), the total risk probability P of the proposed method can be assumed to remain almost the same (almost zero) regardless of the number of receivers belonging to each multicast layer.

$$P = 1 - \prod_{i=1}^{L} \{1 - p(n_i)\} \simeq 0 \tag{3}$$

B. Private Key Congruence

Here we discuss the a probability that private key $d_s^{(i)}$ $(s = 1, \cdots, n_i)$ of layer i becomes congruent with private key $d_t^{(j)}$ $(t = 1, \cdots, n_j)$ of different layer j, where $i, j = 1, \cdots, L, i \neq j$. If this situation occurs, a non-authenticated user can decrypt the message of a prohibited layer. For example, if $d_1^{(1)} = d_1^{(L)}$, a user that has private key $d_1^{(1)}$ can decrypt messages in all layers. A user that has private key $d_1^{(1)}$ should be able to decrypt only the messages of layer 1. In this situation, access control fails.

In order to prevent private key congruence, independent key sets should be generated as shown in Fig. 6. The private key is set to $d_i = g_i^{\gamma}$ (the subscript $^{(j)}$ is omitted for simplicity). Where, g_i is a random generator defined in Appendix A and $\gamma \in \mathbb{Z}_p$ is a random value. The independent key sets can be generated by setting different random sets (g, α, γ). As a result, we can guarantee the following relation:

$$\boldsymbol{d^{(1)} \cap d^{(2)} \cap \cdots \cap d^{(L)} = \emptyset} \tag{4}$$

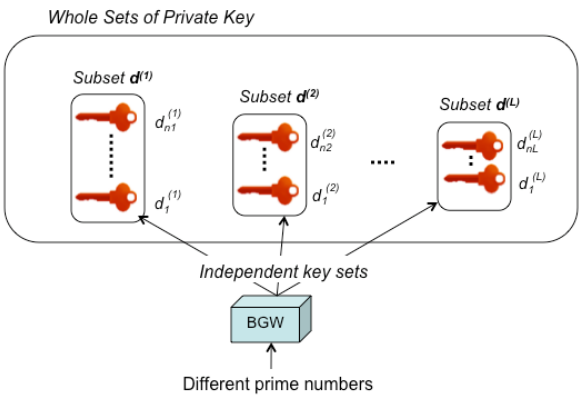

Fig. 6 Independent private key sets generation of the proposed LME method.

Table 1 Comparison of three encryption schemes, ME, MEE and LME, in terms of key size, full header size and layered transmission functionality.

	ME	MEE	LME
Private key d_i size	$n \times P_{rs}$	$\left(\sum_{j=1}^{L} j * n_j\right) \times P_{rs}$	$n \times P_{rs}$
Public key PK size	$(2n+1) \times g_s$	$\left(\sum_{j=1}^{L} j * 2n_j + L\right) \times g_s$	$(2n+1) \times g_s$
Common key CK size	C_s	$L \times C_s$	$L \times C_s$
Full header (S, Hdr) size	H_s	$L \times H_s$	$L \times H_s$
Layered function	×	○	○

P_{rs}: Private key (d_i) size of each receiver
g_s: Public key PK element size
C_s: Common key CK size per layer
H_s: Full header (S, Hdr) size per layer

where $\boldsymbol{d}^{(i)} = \{d_1^{(i)}, \cdots, d_{ni}^{(i)}\}$. Therefore, we can realize secure access control.

4.2 Complexity

Table 1 compares three encryption schemes; 1) Multicast Encryption (ME) [12], 2) Multicast Encryption to Each layer (MEE), 3) the proposed Layered Multicast Encryption (LME), in terms of private key d_i size, public key PK size, common key CK size, full header (input subset S, header Hdr) size and layered transmission functionality. The sizes are the total sizes of keys and full header prepared at the key server.

The private key d_i size and the public key PK size of LME are proportional to the number of receivers, n, the same as for ME. While those of MEE are proportional to the number of receivers, n, and multicast layers L, respectively. With regard to the common key CK size and the full header (S, Hdr) key size, those of the proposed LME are proportional to the number of multicast layers L. This overhead is practical because the number of multicast layers is limited. Computational complexity is evaluated in the next section.

5. Experiment and Results

We conducted an experiment to evaluate the effectiveness of the proposed method. The wavelet decomposition level was set to 3 and the progression order of JPEG2000 was set to RLCP. The number of multicast layers was 3. As shown in Fig. 7, the layers were set to $ML_1 = LL_3$, $ML_2 = \{HL_3, LH_3, HH_3\}$, $ML_3 = \{HL_2, LH_2, HH_2, HL_1, LH_1, HH_1\}$. The total numbers of receivers were

1) $n = 1500$ ($n_1 = 500$, $n_2 = 500$, $n_3 = 500$),
2) $n = 2000$ ($n_1 = 1000$, $n_2 = 500$, $n_3 = 500$),
3) $n = 2500$ ($n_1 = 1500$, $n_2 = 500$, $n_3 = 500$).

We processed one frame taken from the 4 K digital cinema Standard Evaluation Material (StEM) (4096×1716 [pixels]) [16], see Fig. 8.

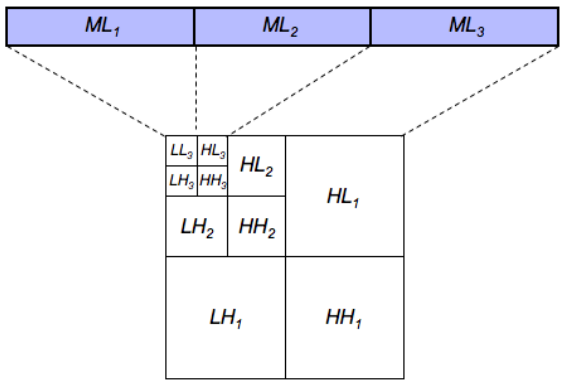

Fig. 7 Set of multicast layers used in the experiment.

Fig. 8 The frame of the StEM sequence used in the experiment.

5.1 Complexity

We evaluated complexity in terms of key size, header size and computational complexity of the broadcast encryption (BGW). η_T pairing [15] was used for the BGW method. The η_t pairing is bilinear pairing defined on elliptic curves, which is known to operate at high speed. The finite-body $\mathbb{F}_{3^{97}}$ of characteristic 3 with degree 97 was used. In the case of characteristic 3, the η_t pairing is calculated on the supersingular elliptic curve defined by $y^2 = x^3 - x + b$ with $b \in \{1, -1\}$. All supersingular curves are isomorphic to this curve. $\mathbb{F}_{3^{97}}$ is an extension field over the $\mathbb{F}_3$ (the finite field of characteristic 3) with extension degree 97.

Table 2 Key and full header size [Bytes] of three encryption schemes.

(a) n = 1500 (n_1 = 500, n_2 = 500, n_3 = 500)

	ME	MEE	LME
Private key d_i size	30,000	60,000	30,000
Public key PK size	60,020	120,060	60,020
Common key CK size	20	60	60
Full header (S, Hdr) size	208	624	624

(b) n = 2000 (n_1 = 1000, n_2 = 500, n_3 = 500)

	ME	MEE	LME
Private key d_i size	40,000	70,000	40,000
Public key PK size	80,020	140,060	80,020
Common key CK size	20	60	60
Full header (S, Hdr) size	270	810	810

(c) n = 2500 (n_1 = 1500, n_2 = 500, n_3 = 500)

	ME	MEE	LME
Private key d_i size	50,000	80,000	50,000
Public key PK size	100,020	160,060	100,020
Common key CK size	20	60	60
Full header (S, Hdr) size	333	999	999

P_{rs} = 160 [bits], g_s = 160 [bits], C_s = 160 [bits], H_{dr} = 160 [bits]

Table 3 PC and software specification.

Category		Specification
PC	Processor	Athlon X2 3800+
	Main memory	2 G byte
	OS	Windows Vista (32 bits)
Software	Language	C
	Compiler and optimization	Gcc 3.4.4 O2-fomit-frame-pointer
	Library for increased digit of executions	GMP 4.2.2

Table 2 lists the key and full header sizes of the three encryption schemes. It shows that private key d_i size and public key PK size are relatively large compared with common key CK size and full header (S, Hdr) size. Private key d_i size and public key PK size of LME are the same as those of ME. LME suppresses the increase in complexity to the maximum extent.

Specifications of the PC and software used are described in Table 3. Table 4 shows time to generate public key and private keys of the three encryption schemes. Table 5 shows encryption time of common key CK and decryption time of code message CM for a designated set S of receivers. Regarding the total number of receivers, n is set to 2000 for these Tables 4–5. The detail execution time of each layer is also shown. Figure 9 shows total execution time for the number of receivers $n = \{1500, 2000, 2500\}$. These results show that LME has low complexity comparable to ME. The following points are understood from these evaluation results.

1. Execution time of generating public and private keys for 2000 receivers is less than 10 seconds. This time is short enough as a preliminary step for contents delivery.
2. Encryption execution time of CK for 2000 receivers is less than 0.1 seconds. This value represents an additional transport delay, but is not significant.
3. Decryption execution time of CM for 2000 receivers is less than 0.1 seconds. This value represents an additional transport delay, but is not significant.
4. From Fig. 9, encryption execution of generation, encryption and decryption is linear against the number of receivers. These times are short enough for practical use.

Table 4 Execution time [sec.] of generating public and private keys of three encryption schemes. The total number of receivers, n, is 2000 (n_1 = 1000, n_2 = 500, n_3 = 500).

Layer	ME	MEE	LME
Layer 1	-	9.522	4.783
Layer 2	-	4.783	2.377
Layer 3	-	2.377	2.377
Total	9.522	16.682	9.537

Table 5 Encryption execution time [sec.] of CK and decryption execution time of CM for a designated set S of receivers. The total number of receivers, n, is 2000 (n_1 = 1000, n_2 = 500, n_3 = 500).

(a) Encryption

Layer	ME	MEE	LME
Layer 1	-	0.082	0.044
Layer 2	-	0.044	0.025
Layer 3	-	0.025	0.025
Total	0.082	0.151	0.094

(b) Decryption

Layer	ME	MEE	LME
Layer 1	-	0.080	0.042
Layer 2	-	0.042	0.023
Layer 3	-	0.023	0.023
Total	0.080	0.145	0.088

5.2 Decoded Images

We evaluated two approaches for common key encryption that perform scrambling on either the signs of the wavelet coefficients or directly on the JPEG2000 code-stream.

A. Pseudorandom Generator

Scrambling is performed by using an existing pseudorandom generator. The pseudorandom generator is a deterministic procedure that produces a pseudorandom distribution from a short uniform input, known as a random seed. The signs of the wavelet coefficients in each code-block are inverted pseudo-randomly. Note that this method modifies only the most significant bit-plane of the coefficients and can be performed on-the-fly during entropy coding. The sign flipping takes place as follows. For each coefficient, a new pseudo-random value is generated and compared with a density threshold. If the pseudo-random value is greater than the threshold, the sign is inverted; otherwise the sign is left unchanged. This approach adds a known noise to the quantized wavelet coefficients.

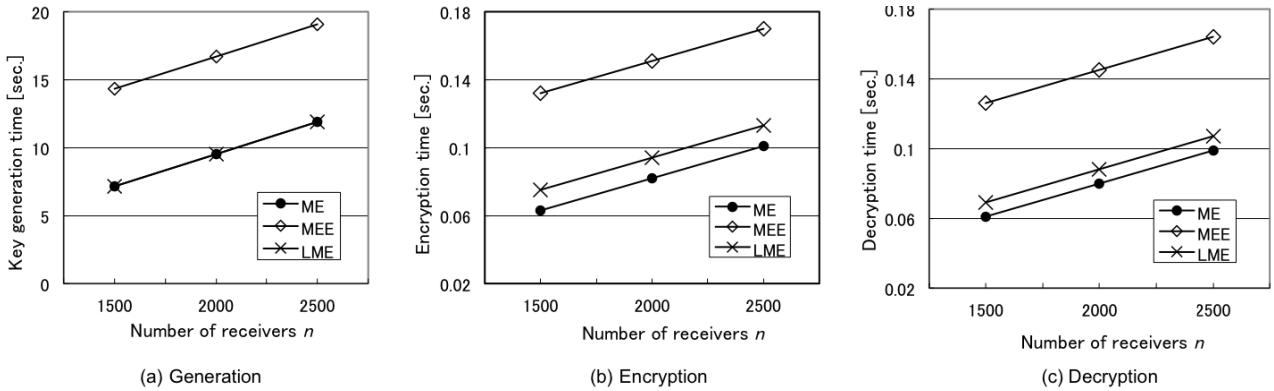

Fig. 9 (a) Time to generate public and private keys, (b) encryption execution time and (c) decryption execution time, of three encryption schemes for the total number of receivers, n, is 1500, 2000 and 2500.

Fig. 10 Decrypted images (double size) by authenticated users for "layers 1" and "layers 1, 2".

Fig. 11 Decoded images of all layers by non-authenticated users (encryption used is sign scrambling of wavelet coefficients).

B. Advanced Encryption Standard (AES)

Scrambling is performed directly on the code-stream by AES [13]. AES is a block cipher adopted as an encryption standard by the U.S. government. It has been analyzed extensively and is now used worldwide. Digital Cinema Initiatives (DCI) [17], a joint venture consisting of Hollywood seven major studios established to create digital cinema specifications, recommends that content must be encrypted by AES.

Fig. 12 Decoded images of all layers by non-authenticated users (encryption used is AES).

Figure 10 shows images decoded by authenticated users. Figure 10 (a) is a result by authenticated users for multicast layer 1. Figure 10 (b) is a result by authenticated users for multicast layers 1 and 2. As we can see, the proposed layered multicast encryption scheme can select each hierarchy as targets from the encrypted code-streams corresponding to each user's authorized image quality.

Figures 11–12 show the results of decoded images of all layers by non-authenticated users by applying sign scrambling of wavelet coefficients and AES, respectively. These are the results of decoding all code-streams. The layer number means the decrypted layers. For example, "Decrypted layer: 1, 2" means that layers "1, 2" were decrypted while layer "3" was not decrypted. From Figs. 11–12, we can see that the non-authenticated users can't view the content.

6. Conclusions

In this paper, we proposed the hierarchical encryption of Motion JPEG2000 code streams for layered multicast transmission. The layered multicast encryption proposal allows a sender to multicast hierarchical encrypted JPEG2000 code streams such that only designated groups of users can decrypt the code streams. While keeping full layering functionality, the proposed method can control video quality with just one private key per user. We clarified the security issues raised and showed that the proposed encryption scheme is secure. It was shown that this method can deliver contents to up to 2000 destinations with practical processing times.

The encrypted bitstreams produced by the proposed method fully comply with JPSEC, so that a standard JPEG2000 decoder can decode the encrypted images and so the useful functionalities of JPEG2000 are retained.

Acknowledgments

The authors would like to thank Prof. Tsuyoshi Takagi of Kyusyu University for his helpful discussions and suggestions.

References

[1] A. Fiat and M. Naor, "Broadcast encryption," CRYPTO 1993, LNCS 0773, pp.480–491.
[2] D. Boneh, C. Gentry, and B. Waters, "Collusion resistant broadcast encryption with short ciphertexts and private keys," CRYPTO '05, pp.258–275, 2005.
[3] C. Delerabl'ee, P. Paillier, and D. Pointcheval, "Fully collusion secure dynamic broadcast encryption with constant-size ciphertexts or decryption keys," Pairing 2007, LNCS 4575, pp.39–59, 2007.
[4] R. Sakai and J. Furukawa, "Identity-based broadcast encryption," Cryptology ePrint Archive Report, 2007/217, 2007.
[5] C. Delerabl'ee, "Identity-based broadcast encryption with constant size ciphertexts and private keys," ASIACRYPT 2007, LNCS 4833, pp.200–215, 2007.
[6] ISO/IEC 15444-1, JPEG2000 Part I: Core coding system, ISO/IEC JTC1/SC29 WG1.
[7] ISO/IEC 15444-8, JPEG2000 Part 8: JPSEC: secure JPEG 2000, ISO/IEC JTC1/SC29 WG1.
[8] L. Apostolopoulos, S. Wee, F. Dufaux, T. Ebrahimi, S. Qibin, and Z. Zhishou, "The emerging JPEG-2000 security (JPSEC) standard," IEEE ISCAS2006, pp.3882–3885, Sept. 2006.
[9] Y. Wu, D. Ma, and R.H. Deng, "Flexible access control to JPEG 2000 image code-streams," IEEE Trans. Multimed., vol.9, no.6, pp.1314–1324, Oct. 2007.
[10] S. Wee and J. Apostolopoulos, "Secure scalable streaming and secure transcoding with JPEG-2000," IEEE ICIP2003, pp.729–732, 2003.
[11] D. Engel, T. Stütz, and A. Uhl, "Format-compliant JPEG2000 encryption in JPSEC: security, applicability, and the impact of compression parameters," EURASIP Journal on Information Security, vol.2007, Article ID 94565, 2007.
[12] H. Uematsu, K. Toyoshima, T. Inoue, H. Takahashi, S. Nishina, T. Takagi, and S. Minato, "Contents multicast method enabling leakage destination tracing and exclusion," APCC2008, pp.1–5, Oct. 2008.
[13] National Institute of Standards and Technology: Data Encryption Standard (DES), http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf, (2001).
[14] Secure Hash Standard, FIPS PUB 180-1, April 1995, http://www.itl.nist.gov/fipspubs/fip180-1.htm.
[15] M. Sirase, Y. Kawahara, T. Takagi, and E. Okamoto, "Universal η_T algorithm over arbitrary extension degree," WISA 2007, LNCS 4867, pp.1–15, 2007.
[16] American Society of Cinematographers/Digital Cinema Initiatives Standard Evaluation Material (StEM), http://www.dcimovies.com/StEM/
[17] Digital Cinema Initiatives, LLC Technology Committee, "Digital Cinema Systems Specification," 2005.

Appendix A: Generation of Message Encryption Key K [2]

1. **Setup**(n): Let $\mathbb{G}$ be a bilinear group of prime order p. The algorithm first picks a random generator, $g \in \mathbb{G}$ and random $\alpha \in \mathbb{Z}_p$. It computes $g_i = g^{(\alpha^i)} \in \mathbb{G}$ for $i = 1, 2, \ldots, n, n+2, \ldots, 2n$.
2. **Set**(K): Pick random t in $\mathbb{Z}_p$ and set

$$K = e(g_{n+1}, g)^t \in \mathbb{G}. \quad (\text{A}\cdot 1)$$

Where, function $e(\cdot, \cdot)$ is a bilinear map with the following properties:

a. For all $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}$, we have $e(u^a, v^b) = e(u, v)^{ab}$
b. The map is not degenerate, i.e., $e(g, g) \neq 1$.

Different random sets (g, t, α) generate finite key set $\mathcal{K}$. Since each $K \in \mathcal{K}$ is generated independently, we can use any K.

Appendix B: Complexity Assumptions of BGW [2]

Security of the BGW method is based on the l-bilinear Diffe-Hellman Exponent assumption (ℓ-BDHE). Let $\mathbb{G}$ be a bilinear group of prime order p. The ℓ-BDHE problem in $\mathbb{G}$ is stated as follows: given a vector of $2l + 1$ elements

$$(h, g, g^{\alpha}, g^{(\alpha^2)}, \cdots, g^{(\alpha^l)}, g^{(\alpha^{l+2})} \cdots, g^{(\alpha^{2l})}) \in \mathbb{G}^{2l+1} \quad (\text{A}\cdot 2)$$

as input, output $e(g, h)^{(\alpha^{l+1})} \in \mathbb{G}$. Note that the input vector is missing the term $g^{(\alpha^{l+1})}$ so that the bilinear map seems to be of little help in computing the required $e(g, h)^{(\alpha^{l+1})}$.

For simplicity of notation, we use g_i to denote $g_i = \mathrm{g}^{(\alpha)^i}$ in $\mathbb{G}$. Algorithm $\mathcal{A}$ has advantage in solving ℓ-BDHE in G if

$$Pr[\mathcal{A}(h, g, g_1, \cdots, g_l, g_{l+2}, \cdots, g_{2l}) = e(g_{l+1}, h)] \geq \epsilon \quad (\text{A}\cdot 3)$$

where the probability is over the random choice of generator g in $\mathbb{G}$, the random choice of h in $\mathbb{G}$, the random choice of α in $\mathbb{Z}_p$, and the random bits used by $\mathcal{A}$.

Security of the BGW method based on the ℓ-BDHE assumption is proved in ref. [2].

Appendix C: Procedure of ME, MEE and LME

Procedure of ME, MEE and LME algorithms are shown in Tables A· 1, A· 2 and A· 3.

Table A· 1 Procedure of ME.

ME (Multicast Encryption/Decryption)			
Key/contents server processing	Receiver $i^{(j)}$ processing	Input	Output
1. Setup of common key			
・Set *CK* value.			*CK*
2. Setup of Broadcast Encryption			
・Generates private keys d_n, …, d_1 and a public key *PK*.		*n* (number of all receivers)	・d_n, …, d_1, *PK*
・Sends *PK* and d_i to each receiver *i*. ($i = n, \ldots, 1$)	・Receiver *i* receives *PK* and d_i.		
3. Encryption of Common key			
・Broadcast Encryption		*S* (a designated subset for target receivers), *PK*.	Hdr (header), *K* (message encryption key of Broadcast encryption).
・Encryption of Common key		*K*, *CK*	*CM*
・Multicasts *S*, Hdr, *CM* to all receivers.	・Receivers receive *S*, Hdr, *CM*.		
	4. Decryption of Common key		
	・Broadcast decryption	*S*, Hdr, *PK*, *i* (user ID), d_i	*K*
	・Decryption of common key	*CM*, *K*	*CK*
5. Layered multicast of codestreams			
・Codestreams encryption		Codestream, *CK*	*Enc*(Codestream)
・Multicasts encrypted a series of codestream to all receivers.	・Receivers receive a series of *Enc*(Codestream).		
6. Decryption of codestreams			
	・Codestreams decryption	*Enc*(Codestream), *CK*	Codestream
	・Restoration of a video stream	Codestreams	A video stream

Table A· 2 Procedure of MEE.

MEE (Multicast Encryption/Decryption to Each layer)			
Key/contents server processing	Receiver $i^{(j)}$ processing	Input	Output
1. Setup of Common key			
▪Set $CK^{(L)}$, …, $CK^{(1)}$ values.			$CK^{(L)}$, …, $CK^{(1)}$
2. Setup of Broadcast Encryption			
▪Generates private keys $d_{(nL+\ldots+nj)}{}^{(j)}$, $d_{(nL+\ldots+nj)-1}{}^{(j)}$, …, $d_1{}^{(j)}$ and a public key $PK^{(j)}$ for each multicast layer ML_j. ($j = L$, …, 1)		n_L(number of receivers at ML_L)	$d_{nL}{}^{(L)}$, …, $d_1{}^{(L)}$, $PK^{(L)}$
		n_L+n_{L-1}	$d_{(nL+\ nL-1)}{}^{(L-1)}$, $d_{(nL+\ nL-1)-1}{}^{(L-1)}$, …, $d_1{}^{(L-1)}$, $PK^{(L-1)}$
		…	…
		n ($= n_L+n_{L-1}+\ldots+n_1$)	$d_n{}^{(1)}$, …, $d_1{}^{(1)}$, $PK^{(1)}$
▪Sends $PK^{(j)}$, …, $PK^{(1)}$ and $d_i{}^{(j)}$, …, $d_i{}^{(1)}$ to receiver i belongs to ML_j. ($i = n$, …, 1)	▪Receiver i belongs to ML_j receives $PK^{(j)}$, …, $PK^{(1)}$ and $d_i{}^{(j)}$, …, $d_i{}^{(1)}$.		
3. Encryption of Common key			
▪Broadcast Encryption		$S^{(L)} \subseteq \{n^{(L)}, \ldots, 1^{(L)}\}$ (a designated subset for ML_L), $PK^{(L)}$	$Hdr^{(L)}$ (header), $K^{(L)}$ (message encryption key of Broadcast encryption for ML_L)
		$S^{(L-1)} \subseteq \{n^{(L)}, \ldots, 1^{(L)}, n^{(L-1)}, \ldots, 1^{(L-1)}\}$, $PK^{(L-1)}$	$Hdr^{(L-1)}$, $K^{(L-1)}$
		…	…
		$S^{(1)} \subseteq \{n^{(L)}, \ldots, 1^{(L)}, n^{(L-1)}, \ldots, 1^{(L-1)}, \ldots, n^{(1)}, \ldots, 1^{(1)}\}$, $PK^{(1)}$	$Hdr^{(1)}$, $K^{(1)}$
▪Encryption of Common key		$K^{(L)}$, $CK^{(L)}$	$CM^{(L)}$
		…	…
		$K^{(1)}$, $CK^{(1)}$	$CM^{(1)}$
▪Multicasts $S^{(L)}$, $Hdr^{(L)}$, $CM^{(L)}$, …, $S^{(1)}$, $Hdr^{(1)}$, $CM^{(1)}$ to all receivers	▪Receivers belong to ML_j pick up $S^{(j)}$, $Hdr^{(j)}$, $CM^{(j)}$, …, $S^{(1)}$, $Hdr^{(1)}$, $CM^{(1)}$.		
4. Decryption of Common key			
	▪Broadcast decryption (Receiver i belongs to ML_j)	$S^{(j)}$, $Hdr^{(j)}$, $PK^{(j)}$, i (user ID), $d_i{}^{(j)}$	$K^{(j)}$
		…	…
		$S^{(1)}$, $Hdr^{(1)}$, $PK^{(1)}$, i , $d_i{}^{(1)}$	$K^{(1)}$
	▪Decryption of common key	$CM^{(j)}$, $K^{(j)}$	$CK^{(j)}$
		…	…
		$CM^{(1)}$, $K^{(1)}$	$CK^{(1)}$
5. Layered multicast of codestreams			
▪Codestreams encryption		$Codestream^{(L)}$, $CK^{(L)}$	$Enc^{(L)}(Codestream^{(L)})$
		…	…
		$Codestream^{(1)}$, $CK^{(1)}$	$Enc^{(1)}(Codestream^{(1)})$
▪Multicasts L series of encrypted codestream of all multicast layers ($Enc^{(L)}(Codestream^{(L)})$, …, $Enc^{(1)}(Codestream^{(1)})$) to all receivers.	▪Receivers belong to ML_j pick up j series of $CK^{(j)}(Codestream^{(j)})$, $Enc^{(j-1)}(Codestream^{(j-1)})$, …, $Enc^{(1)}(Codestream^{(1)})$.		
6. Decryption of codestreams			
	▪Codestreams decryption (Receiver i belongs to ML_j)	$Enc^{(j)}(Codestream)$, …, $CK^{(j)}$	$Codestream^{(j)}$
		$Enc^{(j-1)}(Codestream)$, …, $CK^{(j-1)}$	$Codestream^{(j-1)}$
		…	…
		$Enc^{(1)}(Codestream)$, …, $CK^{(1)}$	$Codestream^{(1)}$
	▪Decoding of codestreams	$Codestream^{(j)}$, $Codestream^{(j-1)}$, …, $Codestream^{(1)}$.	$Codestream^{(j, \ldots, 1)}$
	▪Restoration of a video stream	$Codestreams^{(j, \ldots, 1)}$	A video $stream^{(j, \ldots, 1)}$

shows procedure different from that of the proposed LME method.

Table A· 3 Procedure of LME.

LME (Proposed Layered Multicast Encryption/Decryption)			
Key/content server processing	Receiver $i^{(j)}$ processing	Input	Output
1. Setup of Common key			
▪Set $CK^{(L)}$ value.			$CK^{(L)}$
▪Calculates $CK^{(j-1)}$ from $CK^{(j)}$ by using Hash Function $H(\blacksquare)$. ($j = L, \ldots, 2$)		$CK^{(L)}$	$CK^{(L-1)}$
		$CK^{(L-1)}$	$CK^{(L-2)}$
		…	…
		$CK^{(2)}$	$CK^{(1)}$
2. Setup of Broadcast Encryption			
▪Generates private keys $d_{nj}^{(j)}, \ldots, d_1^{(j)}$ and a public key $PK^{(j)}$ for each multicast layer ML_j ($j = L, \ldots, 1$)		n_L(number of receivers at ML_L)	$d_{nL}^{(L)}, \ldots, d_1^{(L)}, PK^{(L)}$
		n_{L-1}	$d_{nL-1}^{(L-1)}, \ldots, d_1^{(L-1)}, PK^{(L-1)}$
		…	…
		n_1	$d_{n1}^{(1)}, \ldots, d_1^{(1)}, PK^{(1)}$
▪Sends $PK^{(j)}$ and $d_i^{(j)}$ to receiver i belongs to ML_j. ($i = n, \ldots, 1$)	▪Receiver i belongs to ML_j receives $PK^{(j)}$ and $d_i^{(j)}$.		
3. Encryption of Common key			
▪Broadcast Encryption		$S^{(L)} \subseteq \{n^{(L)}, \ldots, 1^{(L)}\}$ (a designated subset for ML_L), $PK^{(L)}$	$Hdr^{(L)}$ (header), $K^{(L)}$ (message encryption key of Broadcast encryption for ML_L)
		$S^{(L-1)} \subseteq \{n^{(L-1)}, \ldots, 1^{(L-1)}\}$, $PK^{(L-1)}$	$Hdr^{(L-1)}$, $K^{(L-1)}$
		…	…
		$S^{(1)} \subseteq \{n^{(1)}, \ldots, 1^{(1)}\}$, $PK^{(1)}$	$Hdr^{(1)}$, $K^{(1)}$
▪Encryption of Common key		$K^{(L)}$, $CK^{(L)}$	$CM^{(L)}$
		…	…
		$K^{(1)}$, $CK^{(1)}$	$CM^{(1)}$
▪Multicasts $S^{(L)}$, $Hdr^{(L)}$, $CM^{(L)}$, …, $S^{(1)}$, $Hdr^{(1)}$, $CM^{(1)}$ to all receivers	▪Receivers belong to ML_j pick up $S^{(j)}$, $Hdr^{(j)}$ and $CM^{(j)}$.		
4. Decryption of Common key			
	▪Broadcast decryption (Receiver i belongs to ML_j)	$S^{(j)}$, $Hdr^{(j)}$, $PK^{(j)}$, i (user ID), $d_i^{(j)}$	$K^{(j)}$
	▪Decryption of common key	$CM^{(j)}$, $K^{(j)}$	$CK^{(j)}$
	▪Calculates $CK^{(j-1)}$ from $CK^{(j)}$ by using Hash Function $H(\blacksquare)$.	$CK^{(j)}$	$CK^{(j-1)}$
	…	…	…
	Calculates $CK^{(1)}$ from $CK^{(2)}$.	$CK^{(2)}$	$CK^{(1)}$
5. Layered multicast of codestreams			
▪Codestreams encryption		Codestream$^{(L)}$, $CK^{(L)}$	$Enc^{(L)}$(Codestream$^{(L)}$)
		…	…
		Codestream$^{(1)}$, $CK^{(1)}$	$Enc^{(1)}$(Codestream$^{(1)}$)
▪Multicasts L series of encrypted codestream of all multicast layers ($Enc^{(L)}$(Codestream$^{(L)}$), …, $Enc^{(1)}$(Codestream$^{(1)}$)) to all receivers.	▪Receivers belong to ML_j pick up j series of $CK^{(j)}$(Codestream$^{(j)}$), $Enc^{(j-1)}$(Codestream$^{(j-1)}$), …, $Enc^{(1)}$(Codestream$^{(1)}$).		
6. Decryption of codestreams			
	▪Codestreams decryption (Receiver i belongs to ML_j)	$Enc^{(j)}$(Codestream), …, $CK^{(j)}$	Codestream$^{(j)}$
		$Enc^{(j-1)}$(Codestream), …, $CK^{(j-1)}$	Codestream$^{(j-1)}$
		…	…
		$Enc^{(1)}$(Codestream), …, $CK^{(1)}$	Codestream$^{(1)}$
	▪Decoding of codestreams	Codestream$^{(j)}$, Codestream$^{(j-1)}$, …, Codestream$^{(1)}$.	Codestream$^{(j, \ldots, 1)}$
	▪Restoration of a video stream	Codestreams$^{(j, \ldots, 1)}$	A video stream$^{(j, \ldots, 1)}$

shows procedure different from that of the MEE method.

Takayuki Nakachi received his Ph.D. degree in electrical engineering from Keio University, Japan, in 1997. Since joining NTT Laboratories in 1997, he has been researching Super High Definition (SHD) image coding, especially in the area of lossless and scalable video coding. His current research interests include distributed source coding, super resolution and secure media processing. From 2006 to 2007, he was a visiting scientist at Stanford University, USA. He is currently a senior research engineer of the media processing systems research group in NTT Network Innovation Laboratories. He is a member of IEEE.

Kan Toyoshima received the B.E. and M.E. degrees in Electrical Engineering from Fukui University, Fukui, Japan in 1982 and 1984 respectively. He joined Electrical Communications Laboratories of Nippon Telegraph and Telephone Public Corporation in 1984. He has been engaged in research and development on ATM (Asynchronous Transfer Mode) transmission system, ATM exchange system, IP over ATM systems, unicast based stream multicast systems (Flexcast) and secure multicast systems. He is currently a Senior Research Engineer in NTT Network Innovation Laboratories. He received the Young Researchers' Award of IEICE of Japan in 1988. He is a member of IEEE.

Yoshihide Tonomura received his B.S. and M.S. degrees in electronics engineering from Nagaoka University of Technology, and a PhD from Tokyo Metropolitan University in 2002, 2004, and 2010, respectively. He joined NTT Network Innovation Laboratories in 2004. His research is focused on image processing theories and applications. He is a member of IEEE.

Tatsuya Fujii received his B.S., M.S. and Ph. D. degrees, all in electrical engineering from the University of Tokyo, Tokyo, Japan, in 1986, 1988, and 1991, respectively. He joined NTT, Japan, in 1991. He has been researching parallel image processing and super high-definition image communication networks. In 1996, he was a visiting researcher of Washington University in St. Louis. He is currently a group leader of the media processing systems research group in NTT Network Innovation Laboratories. He is a member of ITE of Japan and IEEE.