A Framework of Scalable Layered Access Control for Multimedia

Bin B. Zhu, Shipeng Li
Microsoft Research Asia, Beijing 100080, China
{binzhu, spli}@microsoft.com

Min Feng
Dept. of Math, Beijing Univ., Beijing, 100871, China
fengmin@math.pku.edu.cn

***Abstract*—Scalable coders offer various scalabilities with a single codestream. In this paper, we extend our previous work to propose a general framework to enable a single encrypted scalable multimedia stream to support different access types and different access layers for each access type. Access types are protected orthogonally such that access to one type does not deduce the secret to access to other types. Layers are protected one-directionally such that a higher layer accesses and reuses the data of all the lower layers of the same type but not vice versa. An efficient key scheme is proposed to minimize complexity of key management for the framework. With an appropriate encryption scheme is used, an encrypted stream generated by our proposed framework enables fine granularity of scalability to achieve "encrypting once and decrypting multi-ways", which is very desired in many DRM applications. The proposed framework works with all scalable coders. This paper describes the detail to work with JPEG 2000 and MPEG-4 FGS.**

1. INTRODUCTION

A modern scalable codec supports multiple access types and fine granularity scalability (FGS) with a single scalable codestream so that different representations can be extracted from the codestream to fit different application scenarios. Many scalable coders have been proposed, and some have been adopted as standards. JPEG 2000 (J2K) [1] is a wavelet-based scalable image coding standard that supports resolution, quality, color-component, and other scalabilities with a single codestream. MPEG-4 FGS [2] is a fine-grain scalable video coding standard that supports both temporal and quality scalabilities. A recently proposed Universal Scalable Video Coding (USVC) [3] supports temporal, spatial, and quality scalabilities. The Embedded Audio Coding (EAC) [4] supports quality, sampling-rate, and channel scalabilities. All these scalable coders support a large range of qualities. A scalable codestream is organized on some fundamental building blocks, denoted as Scalable Building Blocks (SBBs) in this paper. For example, J2K partitions a codestream according to tiles, components, resolution levels, precincts, and layers (which are not necessarily the same as access layers of our to-be-proposed framework). The J2K SBB is a packet which can be uniquely identified by the aforesaid scalable parameters. Based on supported scalabilities, SBBs can be extracted from a codestream to form a best-fit representation for an application. An SBB may also be truncated or rate-shaped. This encoding-once-decoding-many-ways is very desirable in many applications.

Multimedia Digital Rights Management (DRM) provides persistent protection for digital multimedia from creation to consumption. MPEG-4 has recently adopted a DRM framework, eXtensions to the Intellectual Property Management and Protection (IPMP-X) [5]. The Open Mobile Alliance (OMA) has also adopted a DRM standard for mobile environments [6]. Several commercial DRM products are also available on the market. A typical one is the Microsoft Windows Media Rights Manager (WMRM) [7]. A DRM system such as WMRM encrypts and packages digital media into a digital media file to be distributed in superdistribution. The decryption key is uploaded to a license server along with a specification of rights to use the content desired by the publisher. To play protected content, a user first acquires a license from the license server which contains the decryption key and a specification of the user's access rights. A license is individualized, typically encrypted with a key bound to the user's hardware to prevent unauthorized sharing with others.

In a DRM application, multimedia is typically encrypted without knowing distribution-channel's or playing-device's characteristics. A unique requirement of scalable media encryption for DRM applications is to preserve important, if not all, scalabilities in a cipherstream. This requirement enables any nodes, possibly untrusted, in a distribution route to choose a best-fit representation without compromising the end-to-end security of a DRM system. Several encryption schemes designed for scalable media have been reported recently. Interested readers are referred to the review paper [8]. Among them, Eskicioglu et al. [9] partitions a video stream into a few layers, and each layer is encrypted independently with the same key or a different key. Grosbois et al. [10] proposes encryption schemes for JPEG 2000 to support resolution and quality access, but the two access types cannot be supported simultaneously with a single cipherstream. We have proposed an encryption scheme for MPEG-4 FGS to support both PSNR layers and bitrate layers simultaneously [11][12]. An efficient key scheme for that encryption scheme is proposed in [13].

In this paper, we extend our previous work to propose a general framework of Scalable Layered Access Control (SLAC) which supports any number of access types and any number of access layers per type with a *single* encrypted stream. These access types are protected orthogonally such that access to a layer of one type can not deduce the secrets to access any layers of other types. Layers in SLAC are protected one-directionally such that a higher layer accesses and reuses the data of all the lower layers of the same type but not vice versa. Reusing lower layer data enables easy switching from a low quality play to a high quality play: a simple download of a new license (with possibly incremental protected data, esp. in streaming case) is enough. This is a great advantage over a traditional DRM system which has to download the whole protected content again. Lower layers in SLAC may be unencrypted to enable content-based retrieval in a multimedia database and free preview before purchase. If an appropriate encryption scheme such as the one proposed in [14] is used with SLAC, the encrypted codestream offers fine-grain transcoding without decryption. Such a cipherstream can be adapted to fit a large range of applications without sacrificing the end-to-end security of a multimedia protection system.

0-7803-8834-8/05/$20.00 ©2005 IEEE.

A great challenge for SLAC is to design an efficient key scheme to lower the complexity in key management and distribution. We propose in this paper a novel and efficient key scheme to support arbitrary number of layers and types and all potential organizations of data in a scalable codestream. Like non-scalable case, only one key is needed to be maintained for each SLAC-protected stream by a license server. This "master key" is used with a cryptographic hash function and a Group Diffie-Hellman (GDH) key agreement to generate type keys, layer keys and encryption keys. A virtual layer is introduced in the key generation to guarantee the security of each encrypted block. The detail for SLAC to work with JPEG 2000, MPEG-4 FGS, and USVC are also briefly described.

This paper is organized as follows: In the next section, the proposed framework SLAC is described in detail. Security of SLAC is also discussed in this section. SLAC systems with typical scalable coders such as J2K, the MPEG-4 FGS, and USVC are described in Section 3. We conclude the paper in Section 4. We summarize the major innovations in this paper before we move to the next section: we propose a general multimedia encryption framework to support arbitrary number of access types and access layers with a single cipherstream and a novel key scheme for efficient key management and distribution for the framework.

2. SCALABLE LAYERED ACCESS CONTROL (SLAC) FRAMEWORK

2.1. Notation

The following notation is used throughout the paper:

n	number of different access types
n_j	number of access layers for the j^{th} access type
$L_{i,j}$	i^{th} access layer of j^{th} access type. A virtual layer, denoted as $L_{n_j+1,j}$, can be considered as the highest layer for j^{th} access type for key generation purpose.
K	"master key" for a scalable stream
$K_{i,j}$	key, called *layer key*, for the layer $L_{i,j}$
K_j	key for j^{th} access type, called *type key*. It is also the virtual layer key of j^{th} type, $K_{n_j+1,j} := K_j$.
G_q	A cyclic group
q	the order of the cyclic group G_q
α	a generator of G_q
$\Gamma_{i_1,\cdots,i_n}$	the set consists of the layers $L_{i_1,1},\cdots,L_{i_n,n}$ that each type contributes one layer, where $1 \le i_j \le n_j + 1,\ 1 \le j \le n$. It may also represent the SABB (see Section 2.3) it uniquely identifies
SK_Γ	the encryption key for the SABB specified by Γ
H	a cryptographic hash function
$H^m(x)$	result after H is applied m times to x, $H^0(x) := x$

2.2. Type Keys and Layer Keys

For access layers of the same type in SLAC, higher index denotes higher privilege. A higher layer can access lower layers of the same type, but not vice versa. For example, a layer $L_{b,j}$ can derive all the keys for the lower layers $\{L_{i,j} \mid i < b\}$ of j-th type. The type key K_j for j-th type is generated with the equation $K_j = H(K // j)$ where K is the "master key" for a scalable content, and "//" means concatenation operation. This type key is assigned to the virtual layer $L_{n_j+1,j}$ of the same type: $K_{n_j+1,j} := K_j$. Note that a virtual layer is indexed as the highest actual layer n_j plus 1, so a virtual layer can be considered as a layer next but higher than the highest actual layer for key generation purpose. Keys for all actual layers are derived from the keys of virtual layers:

$$K_{i,j} = H(K_{i+1,j}) = H^{n_j+1-i}(K_{n_j+1,j}),\quad 1 \le i \le n_j \qquad (1)$$

2.3. Encryption Keys and Scalable Media Encryption

As we mentioned in Section 1, SBB is the most fundamental building block in a scalable codestream. Similarly, the most fundamental building block in SLAC is called *Scalable Access Building Block (SABB)* which is uniquely identified by the set of layers $\Gamma_{i_1,\cdots,i_n}$. If a SABB does not contain any data from one access type, the virtual layer of the access type is used to represent the SABB. The key of the virtual layer is also used in Eq. (2) to be described below in SABB key generation. Due to this one-to-one correspondence, the set $\Gamma_{i_1,\cdots,i_n}$ is also used to represent the SABB that the layers in the set uniquely identify when there is no misunderstanding. Note that SABB can be the same as or different from SBB, depending on how access types are specified by the publisher.

Each SABB Γ is encrypted by the corresponding SABB key SK_Γ which is generated from the layer keys of the layers in Γ:

$$SK_\Gamma = \alpha^{\prod K_{i,j}} \mid K_{i,j} \in \Gamma. \qquad (2)$$

Symmetric encryption is used in SLAC to encrypt each SABB so the decryption key for the SABB Γ is also given by Eq. (2).

Depending on the type of multimedia, security requirement, and the scalable coder that SLAC works with, many proposed encryption schemes can be used to encrypt SABBs. Depending on which scheme is used, the resulting cipherstream can be format-compliant or non-format-compliant, with the same as or coarser than the granularity of scalability of the underlying scalable coder. A SABB may be partitioned into smaller data units and each unit may be encrypted independently. A particular useful encryption scheme for SLAC is the one proposed in [14] which preserves the original scalability of the underlying scalable coder in the cipherstream. This enables a DRM-protected content to be transcoded at a fine-grain scalability by any nodes, trusted or not, to fit into any particular applications.

Virtual layers are never accessed. A user can only access actual layers. Virtual layers are used only in generating SABB encryption keys, since otherwise the encryption key of a SABB

which does not contain any data from one or more access types may be deduced by unauthorized users.

2.4. Key Management

Key management plays an important role in a DRM system. It is desirable that a license server stores and manages as few keys as possible for each protected multimedia file. SLAC requires a license server to store only the "master key" for each protected multimedia file (assuming rekeying is not used). This is the same as the traditional single access case. All the other keys can be regenerated from the master key. Of course, a license server may choose to store more "keys" such as the immediate results of Eq. (2) to reduce calculations that may otherwise need in delivering a license to a user.

Keys sent to a user Alice varies, depending on the access types Alice acquires. A user generally acquires fewer access types than the access types a SLAC stream supports, typically one access type such as resolution or quality for JPEG 2000. For example for a SLAC J2K system, Alice may acquire a right to access a J2K image at a certain resolution $L_{b,resolution}$. We denote the access types Alice acquires as AT_{Alice}, and the rest access types as do-not-care types NT_{Alice}. For the given J2K example, $AT_{Alice} = \{L_{b,resolution}\}$, and NT_{Alice} contains all the access types that the publisher selects for a SLAC stream except the resolution type. We note that AT_{Alice} contains a specific layer index for each access type in AT_{Alice}. In fact, AT_{Alice} plays dual roles. It indicates the access types as well as the specific layers of those access types that Alice has acquired.

The license sent to Alice contains the layer keys of the layers specified by AT_{Alice}:

$$SK_{Alice} = \{K_{i,j} \mid L_{i,j} \in AT_{Alice}\}, \tag{3}$$

as well as the exponentials from all the valid combinations of the layers of the access types in NT_{Alice}:

$$PK_{Alice} = \{\alpha^{\prod K_{i,j}} \mid j \in NT_{Alice}, 1 \le i \le n_j + 1\}. \tag{4}$$

The DRM module at Alice side extracts the layer keys in SK_{Alice} and the partial results in PK_{Alice} from the license to calculate the decryption keys according to Eq. (2) for the SABBs that Alice can access.

For some combinations of access types that a publisher allows users to acquire, it is possible to package all possible "keys" in Eq. (4) for various users with the protected content rather than sending a set of "keys" to each individual user inside a license. Protected content is distributed in superdistribution which is a much more efficient distribution mechanism than the license distribution. A license to Alice needs to contain only the layer keys in SK_{Alice}, thus greatly reduces the size of a user license and the workload for a license server (note that hash operation Eq. (1) may still have to be executed, but a hash operation is very fast in general). For example, if a user is allowed to acquire only one access type, then the following "keys"

$$PK = \{\alpha^{\prod_{1 \le j \le n,\ j \ne k} K_{i,j}} \mid 1 \le i \le n_j + 1, 1 \le k \le n\} \tag{5}$$

are packaged with the protected content. Our previous schemes proposed in [11][12][13] belong to this case.

If a single access type is used in a SLAC cipherstream, A SABB key is in fact a layer key. Eqs. (2, 4, 5) are not needed in that case. So the traditional single access protection is included in SLAC as a special case.

2.5. SLAC Security and Implementation

The one-directional access rule in SLAC that a higher layer can access lower layers of the same type but not vice versa is guaranteed by the one-way cryptographic hash function H used in Eq. (1). Since different types use independent random type keys, and to solve a generalized discrete logarithm problem is a hard problem [15], knowledge of one layer key does not deduce layer keys of other types.

The generation of an encryption key from layer keys, i.e., Eq. (2), is in fact the Group Diffie-Hellman key agreement proposed in [16] whose security is proved in [17] based on the Combinational Diffie-Hellman assumption and the Decisional Diffie-Hellman assumption.

In an actual implementation, the discrete logarithm calculations in Eqs. (2, 4, 5) may be replaced by their elliptic curve counterparts.

3. SLAC SYSTEMS

The proposed SLAC framework can work with many proposed scalable coders. Due to the length limitation, we shall briefly describe SLAC with J2K, the MPEG FGS, and USVC in this section. More details will be reported in a separate lengthy paper.

3.1. JPEG 2000 (J2K)

A J2K image codestream is organized in a hierarchical manner with the structure elements: tiles, components, resolution levels, precincts, layers, and packets. A packet is the fundamental building block which can be uniquely identified by the first five aforementioned structure elements. Typical J2K requests are resolution requests, J2K layer, i.e. quality, requests, and sometimes component requests. Suppose that a publisher has decided to support resolution requests and layer requests in a SLAC encrypted J2K codestream, i.e. $n = 2$. Let $j = 1$ denote the resolution access type, and $j = 2$ the quality access type. Assume a codestream is partitioned into n_1 resolution layers and n_2 quality layers. Each resolution layer consists of one or more neighboring J2K resolution levels, and each quality layer consists of one or more neighboring J2K layers. An unencrypted header indicates how a stream is partitioned into different layers. A codestream is therefore partitioned into $n_1 \times n_2$ SABBs. A SABB can be empty. Each non-empty SABB consists of one or more J2K packets. In this case, Eq. (2) reduces to the basic Diffie-Hellman key agreement [15].

If no user is allowed to have multi-parameter requests (i.e., an access by specifying more than one parameter), for example, Alice acquires a right to access to m^{th} resolution layer, then the layer key $SK_{Alice} = \{K_{m,1}\}$ from Eq. (3) and $PK_{Alice} = \{\alpha^{K_{1,2}}, \cdots, \alpha^{K_{n_2,2}}, \alpha^{K_{n_2+1,2}}\}$ from Eq. (4) are packaged in the license sent to Alice. An alternative way of key management for this case is to packetize

$PK = \{\alpha^{K_{1,1}}, \cdots, \alpha^{K_{n_1+1,1}}; \alpha^{K_{1,2}}, \cdots, \alpha^{K_{n_2+1,2}}\}$ from Eq. (5) with the protected content along with G_q parameters q, α, etc. In this latter approach, only $SK_{Alice} = \{K_{m,1}\}$ is contained in the license sent to Alice.

If multi-parameter requests are allowed, for example, Alice acquires a right to access m_1^{th} resolution layer and to m_2^{th} quality layer, then the layers keys $SK_{Alice} = \{K_{m_1,1}, K_{m_2,2}\}$ from Eq. (3) and $PK_{Alice} = \phi$ are sent to Alice in a license.

3.2. MPEG-4 FGS and USVC

For the MPEG-4 FGS, the scheme proposed in [11][12], denoted as SMLFE, is a special case of SLAC. Two mutual-exclusive access types, PSNR layers and bitrate layers, are supported in a cipherstream. The "keys" PK from Eq. (5) are packaged with the protected content along with G_q parameters q, α, etc., as proposed in [13]. MPEG-4 FGS also supports temporal scalability, and USVC supports both the temporal and spatial scalabilities. SLAC can support up to three access types, PSNR, bitrate, and temporal, for MPEG-4 FGS; and up to four access types, PSNR, bitrate, temporal, and spatial, for USVC. Each SLAC access layer consists of one or more neighboring MPEG-4 FGS or USVC layers.

Suppose three access types $n = 3$, PSNR type ($j = 1$), bitrate type ($j = 2$), and temporal type ($j = 3$) are supported in a SLAC + MPEG-4 FGS system. As in SMLFE, the PSNR type and the bitrate type are mutual exclusive, i.e., a user is not allowed to get a right to access a representation specified by both a PSNR layer and a bitrate layer, since these two types are designed to address different needs. In this case, the exponentials $PK = \{\alpha^{K_{1,1}}, \cdots, \alpha^{K_{n_1+1,1}}; \alpha^{K_{1,2}}, \cdots, \alpha^{K_{n_2+1,2}}\}$ from Eq. (5) can be packetized with the protected content. If Alice acquires a right to access a representation specified by two access type parameters, say a PSNR layer b and a temporal layer c, then $SK_{Alice} = \{K_{b,1}, K_{c,3}\}$ from Eq. (3) and $PK_{Alice} = \phi$ are sent in a license to Alice. Note that PK_{Alice} is empty since the exponentials from Eq. (4) are packaged in the protected content and have already been delivered to Alice.

4. CONCLUSION

We have presented in this paper a framework of scalable layered access control for multimedia, SLAC, which exploits the unique features of scalable coders, and supports arbitrary number of access types and layers, limited only by the underlying scalable coder, with a single cipherstream. A higher layer accesses and reuses the data of all the lower layers of the same type but not vice versa. An efficient key scheme for SLAC has also been described which minimizes the complexity of key management and distribution. Low-quality representations can be unencrypted to enable content-based retrieval and free preview. SLAC works with all proposed scalable multimedia coders. Detail to work with J2K, MPEG-4 FGS, and USVC have been described.

REFERENCES

[1] M. Rabbani and R. Joshi, "An Overview of the JPEG 2000 Still Image Compression Standard," *Signal Processing: Image Communications*, 2002, vol. 17, no. 1, pp. 3-48.

[2] W. Li, "Overview of Fine Granularity Scalability in MPEG-4 Video Standard," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 11, no. 3, March 2001, pp. 301 – 317.

[3] F. Wu, S. Li, R. Yan, X. Sun, and Y. Zhang, "Efficient and Universal Scalable Video Coding," *IEEE Int. Conf. Image Processing*, Sept. 2002, vol. 2, pp. 37 – 40.

[4] J. Li, "Embedded Audio Coding (EAC) with Implicit Auditory Masking," *Proc. 10th ACM Int. Conf. Multimedia*, Dec. 2002, pp. 592 – 601.

[5] ISO/IEC JTC1/SC29/WG11 14496-13:2004(E), *Information Technology – Coding of Audio-Visual Object – Part 13: Intellectual Property Management and Protection (IPMP) Extensions*, 2004.

[6] Open Mobile Alliance, *OMA DRM Specification Draft Version 2.0*, March 2004. http://www.openmobilealliance.org

[7] *Architecture of Windows Media Rights Manager*, http://www.microsoft.com/windows/windowsmedia/wm7/drm/architecture.aspx.

[8] B. B. Zhu, M. D. Swanson, and S. Li, "Encryption and Authentication for Scalable Multimedia: Current State of the Art and Challenges," *Proc. of SPIE Internet Multimedia Management Systems V,* vol. 5601, pp. 157-170, Philadelphia PA, Oct. 2004, (invited paper).

[9] A. M. Eskicioglu and E. J. Delp, "An Integrated Approach to Encrypting Scalable Video," *IEEE. Int. Conf. on Multimedia and Expo*, 2002, vol. 1, pp. 573 – 576.

[10] R. Grosbois, P. Gerbelot, and T. Ebrahimi, "Authentication and Access Control in the JPEG 2000 Compressed Domain". *Proc. SPIE 46th Annual Meeting, Applications of Digital Image Processing XXIV*, San Diego, California, 2001, vol. 4472, pp. 95 – 104.

[11] C. Yuan, B. B. Zhu, M. Su, X. Wang, S. Li, and Y. Zhong, "Layered Access Control for MPEG-4 FGS Video," *IEEE Int. Conf. Image Processing*, Barcelona, Spain, Sept. 2003, vol. 1, pp. 517 – 520.

[12] B. B. Zhu, C. Yuan, Y. Wang, S. Li, "Scalable Protection for MPEG-4 Fine Granularity Scalability," to be published in *IEEE Trans Multimedia*.

[13] B. B. Zhu, M. Feng, and S. Li, "An Efficient Key Scheme for Layered Access Control of MPEG-4 FGS Video," *IEEE Int. Conf. Multimedia & Expo,* June 2004, Taiwan.

[14] B. B. Zhu, Y. Yang, and S. Li, "JPEG 2000 Encryption Enabling Fine Granularity Scalability without Decryption," submitted to *ISCAS 2005*.

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

[16] M. Steiner, G. Tsudik, and M. Waidner, "Diffie-Hellman Key Distribution Extended to Group Communication," *Proc. ACM Conf. Computer & Comm. Security*, New Delhi, India, March 1996, pp. 31-37.

[17] E. Bresson, O. Chevassut, and D. Pointcheval, "The Group Diffie-Hellman Problems," *Proc. Advances in Cryptology - ASIACRYPT 2002*. Queenstown, New Zealand, December 2002, LNCS 2501, pp. 497-514.