Scalable Protection for MPEG-4 Fine Granularity Scalability

Bin B. Zhu, *Senior Member, IEEE*, Chun Yuan, Yidong Wang, and Shipeng Li, *Member, IEEE*

***Abstract*—The newly adopted MPEG-4 fine granularity scalability (FGS) video coding standard offers easy and flexible adaptation to varying network bandwidths and different application needs. Encryption for FGS should preserve such adaptation capabilities and enable intermediate stages to process encrypted data directly without decryption. In this paper, we propose two novel encryption algorithms for MPEG-4 FGS that meet these requirements. The first algorithm encrypts an FGS stream (containing both the base and the enhancement layers) into a single access layer and preserves the original fine granularity scalability and error resilience performance in an encrypted stream. The second algorithm encrypts an FGS stream into multiple quality layers divided according to either peak signal-to-noise ratio (PSNR) or bit rates, with lower quality layers being accessible and reusable by a higher quality layer of the same type, but not vice versa. Both PSNR and bit-rate layers are supported simultaneously so a layer of either type can be selected on the fly without decryption. The base layer for the second algorithm may be unencrypted to allow free view of the content at low-quality or content-based search of a video database without decryption. Both algorithms are fast, error-resilient, and have negligible compression overhead. The same approach can be applied to other scalable multimedia formats.**

***Index Terms*—Digital rights management, FGS, fine granularity scalability (FGS), layered access control, MPEG-4, multimedia protection, scalable protection, scalable video encryption, selective encryption, video encryption.**

I. Introduction

SCALABLE coding has been attracting increasing interest in both the industry and academia due to its flexibility and easy adaptation to a wide range of applications such as multimedia streaming. The Moving Picture Experts Group (MPEG) has recently adopted a new scalable video coding format called *fine granularity scalability* (FGS) to its MPEG-4 standard [1]. In MPEG-4 FGS (or simply FGS in the following), a video sequence is compressed into a single stream with two layers: a base layer and an enhancement layer. The base layer is a nonscalable coding of a video sequence at the lower bound of a bit-rate range. The enhancement layer encodes the difference between the original sequence and the reconstructed sequence from the base layer in a scalable manner to offer a range of bit rates for the sequence. Fine grain scalability in FGS enables one-compression-to-meet-the-needs-of-all applications, which is very desirable in multimedia streaming and other applications. Rate reduction and other rate-shaping operations can be performed directly on a compressed stream without decompression.

Multimedia digital rights management (DRM) manages all rights for multimedia from creation to consumption [2], [3]. MPEG has been actively developing a DRM framework, the Intellectual Property Management and Protection (IPMP), for the MPEG-4 standard [4], [5]. There are also several commercial DRM products available on the market. A typical one is the Windows Media Rights Manager from Microsoft [6]. Digital video encryption plays a critical role in a digital video DRM system. Early encryption algorithms were developed for nonscalable formats. With the introduction of scalable video coding, it is natural to extend video encryption to this new format. Scalability offered by scalable coding poses new challenges to the encryption system design, and also enables new services that cannot be offered by nonscalable formats. In designing an encryption system for multimedia in general and scalable multimedia in particular, the following issues need to be considered.

1) **Security**. This is an essential requirement for any encryption system. Video encryption has several unique features that differ from conventional data encryption: video data has a much higher data rate, and partial content leakage may be acceptable. Security for multimedia encryption includes two aspects: encryption perceptual effects (i.e., how much perceptual content leaks out) and encryption security (i.e., how difficult to break a security system). Video encryption should be robust to known-plaintext attacks in particular since many commercial video sequences start with well-known short clips, for example, a company logo clip. Many proposed video encryption algorithms are vulnerable to known-plaintext attacks.
2) **Complexity**. Encryption or decryption incurs computational overhead. Many applications require real-time video decryption on inexpensive consumer devices where low decryption complexity is essential. Complexity and security are typically mutually competitive. A tradeoff is often needed in designing a video encryption system. In some applications, such as early digital TV broadcasting, security is partially sacrificed for low complexity [7].
3) **Compression overhead**. Compression overhead due to encryption manifests in several ways: coding efficiency may be directly reduced with modified coding

Manuscript received December 31, 2002; revised December 8, 2003. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Hong Heather Yu.

B. B. Zhu and S. Li are with Microsoft Research Asia, 3F Sigma, Haidian, Beijing 100080, China (e-mail: binzhu@microsoft.com; spli@microsoft.com).

C. Yuan is with the Department of Computer Science, Tsinghua University, Beijing 100084, China (e-mail: Chun.Yuan@inria.fr).

Y. Wang is with the Department of Computer Science and Technology, Beijing University, Beijing 100871, China (e-mail: wangyidong@icst.pku.edu.cn).

Digital Object Identifier 10.1109/TMM.2005.843340

parameters or input data statistical properties; additional bits may be added to a compressed stream for decryption parameters, boundary indicators of encrypted segments, etc. This compression overhead should be minimized.

4) **Error resilience**. Errors do occur in multimedia storage and transmission. Wireless networks are notorious for transmission errors. Network packets may be lost in transmission due to congestion, buffer overflow, or other network imperfection. Encryption, with a block cipher for example, may expand a single bit error in a ciphertext to many bit errors in the decrypted plaintext. A well-designed encryption system should confine the encryption-incurred error propagation, and enable quick recovery from bit errors and fast resynchronization from packet losses. Many proposed multimedia encryption algorithms were designed under a perfect transmission environment. These algorithms may suffer a great perceptual degradation for an extensive period should bit errors or packet losses occur during multimedia streaming.
5) **Rate shaping/transcoding on ciphertext**. Multimedia may be processed by many intermediate stages such as transcoders from creation to consumption. This is particularly true for a scalable stream where rate shaping operations can be directly applied without decompression. When multimedia is encrypted, it is desirable that these operations can also be applied by intermediate stages directly on an encrypted stream without decryption. Otherwise, decryption and re-encryption have to be performed by an intermediate stage in order to process the multimedia data, which incurs computational overhead, and, more importantly, lowers system security since both decryption and encryption keys have to be shared with the intermediate stage in order to perform an operation.
6) **Encryption granularity**. Consumers are accustomed to random and reverse play for audiovisual data. To meet this requirement, it is desirable to have small encryption granularity, i.e., data units to be independently encrypted are small in size. Small encryption granularity also helps contain encryption-incurred error propagation and offer fine grain scalability in an encrypted stream. On the other hand, small encryption units may incur higher compression overhead since each encryption unit may need additional bits to decrypt the unit. Small encryption units also make a brute force attack easier. Encryption granularity should be designed to balance the conflicting requirements.

Many algorithms have been proposed for video encryption. The most straightforward approach is the *naive algorithm*, a name borrowed from [8], which encrypts a compressed video stream with a conventional cipher such as the data encryption standard (DES) [9] in the same way as encrypting text. A naive algorithm usually has a large computational overhead and the worst error resilience performance. Rate shaping operations cannot be performed directly on a ciphertext generated with a naive algorithm. Another approach is the *selective algorithm*, which exploits compression characteristics and encrypts only important part of compressed video data with conventional ciphers [10], [11]. The partial data to be encrypted can be I-frames, I-frames plus all I-blocks in P- and B-frames, or zero-frequency (dc) coefficients and lower nonzero-frequency (ac) coefficients of I-blocks. Encryption of I-frames alone does not provide sufficient security due to exposed I-blocks in P- and B-frames and inter-frame correlation [8]. A scheme to reduce the amount of data to be encrypted is described in [12] and [13], where half of data, i.e., the even-indexed subsequence, is encrypted with DES and the rest is replaced by the XORing result of odd-indexed and even-indexed subsequences. Another selective algorithm is to randomly flip sign bits of all discrete cosine transform (DCT) coefficients [14] or sign bits of differential values of dc coefficients in I-blocks and sign bits of differential values of motion vectors (MVs) [15]. The third approach is the *scrambling algorithm* which, instead of applying conventional ciphers directly to encrypt video data as in the other two approaches, hides compression parameters or applies permutation to the compression process or its output to prevent unauthorized users from correct decompression. A simple scheme is to replace the zigzag order with a random order in mapping a two-dimensional block of DCT coefficients to a one-dimensional vector in the encoding process [16]. MVs and selected DCT coefficients can be permuted before entropy coding [17], [18]. Variable-length coding (VLC) codes can also be permuted in a format-compliant way [19], [20]. These algorithms modify the underlying data's statistical properties, and therefore lower compression efficiency. An algorithm which permutes codewords in a compressed bitstream without incurring any bit overhead is proposed in [21]. Coding tables in entropy coding can also be permuted [22].

While some aforementioned algorithms, for example most scrambling algorithms, are equally applicable to MPEG-4 FGS, algorithms designed specifically for scalable formats have also been reported recently. An algorithm called *secure scalable streaming* (SSS), which enables bit-rate reduction without decryption, is described in [23]. For MPEG-4 FGS, the approach partitions video data in both the base and the enhancement layers into packets. All data except header fields in each packet is encrypted with DES in the cipher block chaining (CBC) mode. Hints for rate-distortion (RD)-optimal cutoff points are inserted into unencrypted header fields to allow RD-optimal truncations. A simple layered access control algorithm for wavelet image coding is proposed in [24], where signs of wavelet coefficients in high band layers are randomly inverted. Selective encryption that encrypts only important data in quadtree and wavelet image coding is described in [25].

In this paper, we propose two novel encryption algorithms for MPEG-4 FGS. The first algorithm is called *scalable single-layer FGS encryption* (SSLFE), which encrypts a scalable stream into a single access layer. The original FGS and error resilience performance are fully preserved in an encrypted stream. The second algorithm is called *scalable multilayer FGS encryption* (SMLFE) which encrypts a single scalable stream into multiple quality layers, with lower layers being accessible and reusable by a higher layer, but not vice versa. Layers can be partitioned according to PSNR or bit rates. Both PSNR

and bit-rate layers are supported simultaneously. Block-level bit-rate reduction can be performed directly on encrypted data. The base layer in the second algorithm may be unencrypted to allow free view of the content at low quality. In both algorithms, encryption is applied after entropy coding so there is no adverse impact on a codec's coding efficiency. The number of bits added to an output stream for correct decryption is also negligible. Both SSLFE and SMLFE are fast and error-resilient. Preliminary versions of the two algorithms are reported in [26] and [27], respectively.

This paper is organized as follows. In Section II MPEG-4 FGS and the encryption algorithm used for the proposed algorithms are briefly described. SSLFE is described in Section III and SMLFE in Section IV. Security of the proposed algorithms is discussed in Section V. Implementation details and experimental results are reported in Section VI. We conclude the paper in Section VII.

II. Background

A. MPEG-4 FGS

This subsection gives a very brief introduction to MPEG-4 FGS. More details can be found in [1]. In MPEG-4, the *video object* (VO) corresponds to entities in the bitstream that can be accessed and manipulated. An instance of VO at a given time is called the *video object plane* (VOP) [28]. The basic idea in MPEG-4 FGS is to encode a video sequence into a nonscalable base layer and a scalable enhancement layer. The MPEG-4 Advanced Simple Profile (ASP) provides a subset of nonscalable video coding tools to achieve high coding efficiency for the base layer. The bit rate of the base layer is the lower bound of a bit-rate range that FGS supports. The base layer is typically encoded at a very low bit rate. The FGS profile is used to obtain the enhancement layer to achieve optimized video quality with a single stream for a wide range of bit rates. More precisely, each frame's residue, i.e., the difference between the original frame and the corresponding frame reconstructed from the base layer is encoded for the enhancement layer in a scalable manner: DCT coefficients of the residue are compressed bit-plane wise from the most significant bit to the least significant bit. For a temporal enhancement frame which does not have a corresponding frame in the base layer, the bit-plane coding is applied to the entire DCT coefficients of the frame. This is called *FGS temporal scalability* (FGST). FGST can be encoded using either forward or bi-directional prediction from the base layer. MPEG-4 FGS provides very fine grain scalability to allow near RD-optimal bit-rate reduction.

To simplify description in this paper, a VOP in the base layer is called a *base VOP*. A VOP in the enhancement layer is called an *enhancement VOP* which can be either an *FGS VOP* or an *FGST VOP*. Frames and VOPs are interchangeably used. MPGE-4 FGS (or simply FGS[1]) is used frequently in this paper to include both MPEG-4 ASP and the MPEG-4 FGS profile, i.e., the coding that generates both the base layer and the enhancement layer. An *FGS stream* (or an *MPEG-4 FGS stream*) refers to a stream that contains both of these two layers.

In MPEG-4 FGS, video data is grouped into video packets, which are separated by the resynchronization marker. The bit-plane start code, *fgs_bp_start_code*, in the enhancement layer also serves as a resynchronization marker for error resilience purposes. For the sake of simple description in this paper, both the resynchronization marker and *fgs_bp_start_code* are referred to as *vp_marker*, and the data separated by a *vp_marker* is called a *video packet*. Video packets are aligned with macroblocks. In MPEG-4 FGS, video packets are determined at the time of compression, but can be changed later by modifying resynchronization marker positions.

Due to the different roles they play, the base layer and the enhancement layer are unequally protected against network imperfection in transmission in real applications. The base layer is typically well protected against bit errors or packet losses, and is virtually lossless in transmission. The enhancement layer, on the other hand, is lightly or not protected against network imperfection. In this paper, we assume that the base layer is lossless transmitted.

B. Chain & Sum (C&S) Encryption Algorithm

In this paper an *encryption cell*, or simply *cell*, is defined as a chuck of data that is independently encrypted. An encryption cell in both SSLFE and SMLFE is encrypted with the C&S encryption algorithm proposed in [29] with minor modifications. In the C&S encryption, a cell is first processed by a CBC-like primitive in which the block cipher is replaced by a pair of invertible universal hash functions that are applied alternatively. The output blocks (i.e., "words" in [29]) are summed up and written in place of the next-to-last block. The resulting last two blocks, called *pre-MAC* (message authentication code), are encrypted by a block cipher with a key, i.e., the video encryption key K_v of the video sequence in our proposed algorithms. This encrypted pre-MAC is implicitly a MAC value for the cell. The pre-MAC combined with K_v is input to a stream cipher to encrypt the remaining blocks. Although we do not mention it explicitly in this paper, it should be understood that the parameters of the invertible universal hash functions used in the C&S encryption are also part of the keys in encryption of a video sequence. The whole process can be reversed to recover the plaintext. When a cell contains a trailing partial block, pre-MAC is calculated with the aforementioned procedure as if the partial block does not exist. The partial block is then encrypted by the stream cipher along with other non-pre-MAC blocks. Details for the C&S encryption can be found in [29].

Since a *keyed* hash value (i.e., pre-MAC) of a cell is used as part of the key to a cipher to encrypt the cell itself, a single bit difference in a plaintext results in an uncorrelated ciphertext. This effect can also be achieved with other encryption algorithms, for example, a block cipher in the CBC mode where different initial vectors (IVs) are used for different cells. These IVs have to be inserted into the output in video encryption, which incurs compression overhead. The C&S encryption, on the other hand, does not incur any compression overhead since the keyed hash value replaces part of the underlying data to be encrypted.

[1]This should not be confused with FGS VOP used in this paper. An FGS VOP in this paper always means the normal, i.e., the non-FGST, VOP in the enhancement layer.

This high sensitivity of ciphertext to plaintext without any compression overhead is very desirable in encrypting scalable video since it allows smaller encryption cells thus finer grain scalability without worrying about compression overhead, and makes the encryption robust to known plain-text attacks (see the discussion in Section V).

The C&S encryption can be considered as an enhanced stream cipher where part of the input key is a hash value of the data to be encrypted. Since all the data except the pre-MAC is encrypted with a stream cipher, the C&S encryption is very fast. It is reported in [29] that the pre-MAC calculation speed was approximately 350 Mbps ("bps" means bits per second) on a Pentium 266-MHZ system when the operation was implemented in assembly language on the field $Z(2^{31}-1)$.

These advantages of the C&S encryption are gained at the cost of others. Compared with a block cipher in the CBC mode, the C&S encryption has a disadvantage that one bit error, especially when it occurs in the encrypted pre-MAC, may affect many bits or even the whole cell. The CBC encryption, on the other hand, can contain a bit error within the current block that the wrong bit lies and the next block. This is not a significant disadvantage in video encryption since the CBC decrypted data in this case is unlikely to be decodable by the codec following the decryption module. A careful alignment of encryption cells with the underlying compression structure can alleviate this error propagation problem. Another disadvantage is that identical inputs produce identical ciphertext outputs in the C&S encryption but different outputs in the CBC encryption, thanks to different IVs being used in encrypting each input in the CBC encryption. This identical-input-identical-output may be a serious vulnerability in cryptography but is acceptable in video encryption, as we will explain in Section V.

After a careful comparison of different encryption algorithms, we choose the C&S encryption as the encryption algorithm in our proposed algorithms. It is worth noting that other encryption algorithms can also be used in our algorithms, with possibly minor changes.

III. Scalable Single-Layer FGS Encryption (SSLFE)

A. Base-Layer Encryption is Not Enough

The goal for SSLFE is to provide a lightweight encryption that preserves the full fine grain scalability and the error resilience performance of MPEG-4 FGS after encryption. An intuitive approach is to extend selective encryption algorithms proposed in [25] to FGS where the base layer is encrypted but the enhancement layer is not encrypted. Since the enhancement layer uses base VOPs as references, it seems that such an approach does not leak much visual information of the content. This conjecture is valid if we are only concerned about each individual frame. For a video sequence, this intuitive approach leaks important visual information. We have performed the following experiment to find out how much visual information this approach leaks out: a QCIF video sequence is compressed into a base layer at around 50 kbps and an enhancement layer at 1.0 Mbps. During decompression, the pixel values in the base layer are set to 0 (or other fixed values) to mimic unavailability of the encrypted base layer. The enhancement layer is then used to reconstruct the video sequence. The QCIF sequences listed in Table I of Section VI were tested. We had the following interesting observations for the reconstructed sequences: The visual effect for each frame was about the same as that of SSLFE in the full encryption mode (see Fig. 2). No meaningful visual information could be extracted from each individual frame. When a reconstructed sequence was played, however, the outlines and trajectories of moving objects were readily visible. Moreover, we could easily identify what these objects were and what they were doing. This phenomenon can be explained by the strong correlation among neighboring frames in a video sequence. When reference base VOPs are strongly correlated, a series of enhancement VOPs reveal some content information. Such content leakage may be unacceptable in many applications. We conclude that encrypting the base layer alone is not enough in general to protect FGS streams. SSLFE described next removes this vulnerability with a lightweight encryption on the enhancement layer, too.

B. Details of SSLFE

SSLFE exploits the FGS features mentioned in Section II-A, and applies different encryption schemes to the base layer and the enhancement layer. The base layer is encrypted in either a selective or a full encryption mode. This is similar to conventional encryption algorithms for nonscalable coding. A lightweight selective encryption is applied to the enhancement layer to make the encryption transparent to intermediate processing stages. In this way, the FGS is fully preserved in an encrypted FGS stream.

1) Base-Layer Encryption: The base layer can be encrypted in either a selective or a full encryption mode. The C&S encryption is used to encrypt each cell. In the *selective encryption mode* (SEM), the dc values with known number of bits (i.e. *intra_dc_coefficient* and *dct_dc_differential*), the sign bits of DCT coefficients, and the MV sign bits (i.e., the sign bits of *horizontal_mv_data* and *vertical_mv_data*), as well as the MV residues, *horizontal_mv_residual* and *vertical_mv_residual*, are extracted from each base VOP to form an encryption cell. After encryption, the corresponding bits in the ciphertext are placed back to write over the original bits. In the *full encryption mode* (FEM), the entropy-coded video data except the VOP header forms an encryption cell for each base VOP. The VOP start code, *vop_start_code*, serves as the separator header for each encryption cell. When resynchronization markers are used (i.e., the flag *resync_marker_disabled* is set to 0), an alternative FEM scheme is that the video data in each video packet forms an encryption cell. In this alternative scheme, the encryption cell separator is the resynchronization marker, and a base VOP may contain several encryption cells. There will be no more discussion on this alternative scheme since we have assumed lossless transmission for the base layer in our design.

In FEM, there is a slim chance that ciphertext emulates the encryption cell separator. To deal with this case, a one-bit flag called *emulation flag* is inserted into the header of each base VOP. If emulation occurs in a base VOP, the corresponding emulation flag is set to 1, followed by a custom header, *encryption_block_size*, which indicates the size of the VOP. If

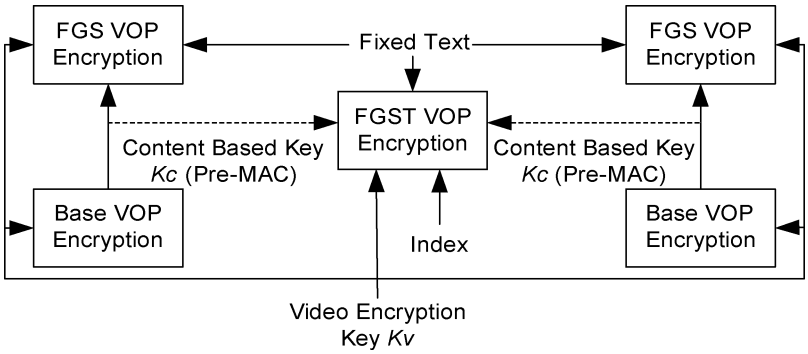


Fig. 1. Block diagram for the keys used in SSLFE.

there is no emulation, the emulation flag is set to 0 and *encryption_block_size* is not inserted. Both the flag and *encryption_block_size* are unencrypted.

2) Enhancement Layer Encryption: Encryption for the enhancement layer in SSLFE is designed to be a lightweight, format-compliant, selective encryption to preserve the fine granularity scalability of MPEG-4 FGS. Fig. 1 shows the keys used to encrypt FGS and FGST VOPs in the enhancement layer. In the FGS VOP encryption, the pre-MAC from the encryption of the reference base VOP is used as the content-based key K_c. This K_c is combined with the video encryption key K_v, as well as a fixed text such as "Enhancement Layer", as the input to a stream cipher to generate a random binary matrix of the same size as the VOP. The sign bit of a DCT coefficient in the FGS VOP, should it appear, would XOR the random bit at the same position in the random binary matrix. In this way, if there is any packet loss, received packets can still be correctly aligned with the random bits. A fixed text is needed as part of the key to the stream cipher since K_v and K_c (i.e., the pre-MAC) are also the input to the stream cipher of the C&S encryption in encrypting the reference base VOP. The fixed text ensures that uncorrelated random sequences are used when the same stream cipher is used to encrypt both base VOPs and enhancement VOPs.

For FGST VOPs, the sign bits of DCT coefficients are encrypted as in the FGS VOP encryption. In addition, the MV sign bits and the MV residues are also encrypted by a random sequence in the same way as encrypting the DCT sign bits. The content-based key K_c in encrypting an FGST VOP is determined according to how the prediction is used in encoding the FGST VOP. All the pre-MACs from encryption of the base VOPs that the prediction is based on are combined as the content-based key K_c for the FGST VOP.

It is possible that predictions in several FGST VOPs use the same reference base VOPs. To prevent correlated random sequences from being used in encrypting these FGS VOPs, an FGST VOP index is also used as part of the key to the stream cipher, as shown in Fig. 1. This index is inserted as a custom header into the FGST VOP header. The index also serves as an FGST VOP identifier in case some FGST VOPs are lost. FGST VOP indexes can be reused in different groups of FGST VOPs since different groups use content-based keys from different base VOPs, and we are only interested in the FGST VOP order within the same group. Two to three bits are enough for the FGST VOP index in most applications. This inserted index is preferred over a time stamp in the FGST VOP since some typical operations performed by an intermediate stage may change the time stamp, which may render a wrong random sequence to be used at decryption.

C. SSLFE Performance Discussion

SSLFE generates fully format-compliant streams in SEM but only the enhancement layer stream is format-compliant[2] in FEM. In either SEM or FEM, encryption is applied after entropy coding, and has no adverse impact to codec's coding efficiency. There is no single bit added to the output in encrypting the base layer in SEM. Therefore, the base-layer encryption in SEM incurs no compression overhead. The base-layer encryption in FEM, on the other hand, adds one bit, i.e., the one-bit emulation flag, to each base VOP. This is an overhead of 30 bps if there are 30 base VOPs per second, or 0.06% of the base-layer bit rate if the base layer is compressed at 50 kbps. Another potential overhead for the base-layer encryption in FEM is the custom header, *encryption_block_size*, which is used to indicate the base VOP size when ciphertext emulates the encryption cell separator, *vop_start_code*, which is 32 bits [28]. The probability that emulation occurs is $O(2^{-32})$. Therefore the compression overhead caused by inserting *encryption_block_size* is statistically negligible. For the enhancement layer, FGS VOP encryption does not incur any compression overhead but FGST VOP encryption adds up to 3 bits per FGST VOP, which is typically about 70 bps or less overhead (depending on number of FGST VOPs per second). This overhead is very small as compared to a typical bit rate of 1 Mbps or higher for the enhancement layer. The overhead increases if the enhancement layer is truncated but it is still very small. We conclude that SSLFE has negligible compression overhead.

As we mentioned in Section II-A, the base layer is highly protected against network imperfection. We can assume that the base-layer transmission is lossless in analyzing the performance of SSLFE. The enhancement layer is encrypted by a stream cipher so encryption does not cause any error propagation. Each encrypted field (i.e., the sign bit of a DCT coefficient, etc.) is decrypted by XORing with a field at fixed position in a random sequence so any received data can be decrypted.[3] In conclusion,

[2]If FGST is used, an index is added to each FGST VOP as a custom header. This should not affect the format-compliance.

[3]If the FGST VOP index is lost, a received FGST VOP cannot be correctly decrypted. Since the index is packetized with the VOP header in transmission, loss of the header packet makes it impossible to decompress the received VOP either, even without any encryption.

an FGS stream encrypted with SSLFE has exactly the same error resilience performance as FGS when bit errors or packet losses occur (to the enhancement layer). In other words, SSLFE is error resilient.

In MPEG-4 FGS, it is the enhancement layer which offers fine grain scalability. Since the output of the enhancement encryption in SSLFE is format-compliant, and each encrypted field is encrypted in situ, any scalable operations could be performed directly on the encrypted enhancement layer as if the layer were not encrypted. SSLFE enables the full FGS in encrypted streams.

SSLFE itself is a selective encryption algorithm where the major computational overhead is on the base-layer processing. In MPEG-4 FGS, the base layer is encoded at a much lower bit rate than the enhancement layer. Only a small portion of data needs to be processed in SSLFE. Since the C&S encryption is very fast, SSLFE is efficient and fast, too. Experimental data on processing speeds of SSLFE will be reported in Section VI.

IV. Scalable Multilayer FGS Encryption (SMLFE)

SSLFE described in the previous section encrypts the entire MPEG-4 FGS stream into a single access layer with a single access key. This access mode is exactly the same as the non-scalable video encryption. MPEG-4 FGS is scalable at a fine grain scale. A single FGS stream can be encrypted into scalable multiple quality layers to provide multiple accesses to different qualities. This is exactly the goal for SMLFE to be described in this section. SMLFE encrypts FGS video data (FGS headers are excluded) into different quality layers. Each quality layer has its own access control. Quality layers can be divided according to PSNR (the *PSNR layers*) or to bit rates (the *bit-rate layers*). These two types of quality layers serve different application scenarios. The PSNR layer is a natural choice if we separate quality layers according to different visual qualities targeted at local play or other applications, although it is well known that PSNR is not a good measure of perceptual quality. If a video sequence is targeted for streaming over a network, the bit-rate layer is a natural choice. A single type of quality layer does not work well for both scenarios since each frame may have variable number of coded bit planes or bits. SMLFE supports both layer types simultaneously. A media server can select directly from an encrypted stream a quality layer of either type to send. SMLFE allows a higher quality layer to access and reuse the data of lower quality layers of the same type, but not vice versa. The protection of the two different layer types is orthogonal, i.e., a right to access a layer of one type does not make the layers of the other type also accessible, or vice versa.

The base layer may be exposed without encryption in SMLFE to provide a free view of the content at a low quality. This is desirable in many applications. For example, a potential consumer can skim video content before purchase. Another example is that a content-based search in a video database consisting of encrypted video sequences can simply work on the unencrypted base layer without decrypting any video sequences. In the following description of SMLFE, we assume that the base layer is not encrypted. It is trivial to encrypt the base layer as another quality layer if necessary.

A. Design Assumptions

The following two assumptions on multimedia transmission have been made in the design of SMLFE:

- a transport packet should contain complete video packets so a video packet is either received as a whole or completely lost when packet loss occurs;
- a transport packet should contain the information to derive the indexes of the video packets it contains so a received video packet can be decoded to the right position in transmission with packet losses;

To meet the first assumption, a format-aware packetizer is needed for transport packetization. The above two assumptions are valid for most multimedia networks in real applications. Furthermore, we also assume that video packets do not change after compression. In MPEG-4 FGS, video packets are determined at the time of compression, but they can be changed after compression by moving around, removing, or inserting resynchronization markers. This restriction on the video packet may have adverse impact in some applications, but should be acceptable in most applications.

In MPEG-4, the size of a video packet is typically kept below a threshold. If the number of bits contained in the current video packet exceeds a predetermined threshold, a new video packet is created at the start of the next macroblock [28]. The size of a video packet may still vary greatly because the most significant bit-plane may have much less number of bits (recall from Section II-A that the bit-plane start code also servers as a video packet separator), and also because the last video packet in a bit plane may be much smaller than others. The latter case can be avoided by adjusting the sizes of the previous video packets of the same bit plane. We can assume that the latter case does not occur in the design of SMLFE.

B. Details of SMLFE

In SMLFE, the video data in each video packet forms an encryption cell which is independently encrypted with the C&S encryption. The existing separator *vp_marker* is used as the separator for each encryption cell. When a ciphertext emulates the separator *vp_marker*, a technique similar to the one described in Section III-B1 can be used. More precisely, a one-bit emulation flag is inserted to the enhancement VOP header. The flag is off by default. If emulation occurs in any video packet of an enhancement VOP, the flag is set to on, followed by a custom header, *emulation_info*, which lists the indexes and sizes of the video packets in which emulation occurs. If the emulation flag is off, *emulation_info* is not inserted.

If the number of bits in the most significant bit-plane is too small, several video packets are combined together to form a large enough encryption cell. A custom marker *merged_vp* is inserted into the enhancement VOP header to indicate such a case (see below). In our experiments, only the most significant bit-plane may need to merge into the next video packet, so one bit for *merged_vp* is adequate.

A PSNR layer is a group of adjacent bit planes in each enhancement VOP. A bit-rate layer is a group of adjacent video packets. Each layer of either type is therefore aligned with the video packets. A content owner can specify where to separate

a PSNR or bit-rate layer according to a video's characteristics and business needs, which is beyond the scope of this paper. Suppose that for each enhancement VOP all the bit-planes are partitioned into T adjacent groups to form T PSNR layers, and that all the video packets are partitioned into M adjacent groups to form M bit-rate layers, the data of the enhancement VOP is then partitioned into $T \times M$ different segments $\{S_{t,m}\}$, where $t = 1, \cdots, T$ and $m = 1, \cdots, M$. There exists some correlation between PSNR layers and bit-rate layers. For example, a low PSNR layer is likely to share data with a low bit-rate layer but unlikely to share with a high bit-rate layer. This means that some segments out of the total $T \times M$ segments per enhancement VOP are likely to be empty (i.e., of length 0).

In SMLFE a set of $T \times M$ different keys $\{K_{t,m}\}$ are independently and randomly generated for each video sequence. The key $K_{t,m}$ is used to encrypt the corresponding segment $S_{t,m}$ for each enhancement VOP, where $t = 1, \cdots, T$ and $m = 1, \cdots, M$. Instead of a single video encryption key K_v in SSLFE described previously, SMLFE has $T \times M$ segment encryption keys for a video sequence.[4] These keys are reused in encryption of each enhancement VOP. A nonempty segment $S_{t,m}$ is first partitioned into video packets, and the video data in each video packet form an encryption cell to be encrypted by the C&S encryption with the segment key $K_{t,m}$.

When a user gets the right to access to a certain quality layer, all the keys for that and lower quality layers of the same type are sent to the user. For example, if the layer a user has the permission to access is the PSNR layer $t = 2$, then the $2M$ keys $\{K_{t,m}\}$, where $t \leq 2$ and $m = 1, \cdots, M$, are sent to the user. In this way, a right to access a quality layer can also access lower quality layers of the same type. Higher quality layers of the same type and quality layers of a different type are not accessible. Note that in a DRM system, these access keys are packed as part of a license sent to a user.

There is no need to use a marker to separate PSNR layers since the separation points are the same for all the VOPs. The existing MPEG-4 FGS bit-plane start code serves such a purpose. Separation points for bit-rate layers, on the other hand, vary from one VOP to another. In SMLFE no marker is actually inserted into a bitstream to separate bit-rate layers. Instead, a custom header *smlfe_vop_info* is inserted into the header of each enhancement VOP to indicate how many video packets an enhancement VOP has at the encryption time. A few additional bits are also added to register possible minor variations from the general grouping rule for video packets to allow a fine-tuned division of bit-rate layers for each VOP. This approach is feasible since MPEG-4 tries to maintain video packets at a constant size [28]. The previous mentioned marker *merged_vp* and the one-bit emulation flag are also placed into *smlfe_vop_info*. From our experiments, 24 bits are adequate for most applications. Because of the second assumption in Section IV-A, bit-rate layer separation points can be derived from this 24 bit custom header even under the circumstance that some video packets are lost in transmission. This guarantees that the right key is used in decrypting each received video packet. In applications where the 2nd assumption is invalid, a bit-rate layer index header has to be inserted into the unencrypted part for each video packet to indicate which bit-rate layer the video packet belongs to. Eight different bit-rate layers, i.e., a bit-rate layer index header of three bits, should be enough for most applications.

Note that in streaming applications, it may be advantageous for a streaming server to derive bit-rate layer separation points so that no bandwidth is wasted in sending extra data that an end user cannot access. Another method is to use an assistant file which contains all break points for bit-rate layers. This assistant file is used by a streaming server without sending to end users so it does not consume any end user's bandwidth.

C. SMLFE Performance Discussion

Like SSLME, encryption in SMLFE is applied after compression so the underlying codec's coding efficiency is not affected. As described above, *smlfe_vop_info* adds 24 bit to each enhancement VOP. This causes 720 bps overhead for a 30 frames per second video sequence. Compared to a 1 Mbps or above bit rate that the enhancement layer is typically compressed to, this overhead is negligible. In the case that a ciphertext emulates the video packet separator *vp_marker* in a VOP, *emulation_info* is added to the enhancement VOP. In MPEG-4, the resynchronization marker is at least 17 bits and *fgs_bp_start_code* has 27 bits for identification [28]. This means *vp_marker* has at least 17 bits in length for identifying the header, which implies that the probability that *vp_marker* is emulated is $O(2^{-17})$ or less. In other words, the compression overhead caused by inserting *emulation_info* when emulation of *vp_marker* occurs is statistically negligible. In conclusion, SMLFE has negligible compression overhead.

Let us look at SMLFE's error resilience performance. Under the two assumptions described in Section IV-A, a received video packet can be correctly decrypted regardless of loss of other packets, as long as the aforementioned custom header, *smlfe_vop_info*, is received. *smlfe_vop_info* is transmitted with the VOP header. When *smlfe_vop_info* is lost, the VOP header is lost, and the whole VOP is not decodable regardless of whether the VOP is encrypted or not. Therefore SMLFE incurs no adverse effect to the error resilience performance of MPEG-4 FGS when packet losses occur. In other words, the algorithm is robust to packet losses.

In MPEG-4, if bit errors occur and the Reversible Variable Length Codes (RVLC) is not used, the video packet that contains bit errors is typically discarded. Using RVLC reduces coding efficiency. In SMLFE, a bit error in an encryption cell extends to many bits inside the cell after decryption, but never extends to other cells, thanks to independent encryption of each cell. Only the affected cell is discarded in this case. Since an encryption cell is aligned with a video packet (except possible video packets from the most significant bit-plane), the algorithm has no adverse impact to the final result when bit errors occur. In conclusion, SMLFE is robust to both bit errors and packet losses.

In SMLFE, video data is encrypted in a full encryption mode. This prevents an intermediate stage from being able to perform RD-optimal truncation or other fine grain scalable operations directly on an encrypted cell. Instead, a coarser, video-packet-level truncation is supported in the SMLFE encrypted stream.

[4]For security reasons, these keys can be updated in the middle of a sequence, esp. when the sequence is long. In this case, a video sequence has more than $T \times M$ video encryption keys.

Under the first assumption in Section IV-A, video-packet-level truncation is enough for transmission, even though such a block-level truncation may not necessarily be RD-optimal for an arbitrary desired bit rate. This coarse scalability in the encrypted stream should not have a significant adverse impact to the applications that SMLFE is designed for where scalable multiple access layers are desirable.

The full encryption in SMLFE also has some but limited adverse impact to the processing speed. In SMLFE, a large amount of data has to be encrypted. Encryption is generally not an issue since, like the MPEG-4 FGS compression, it is applied only once. Intermediate stages can perform video-packet-level scalable operations directly on an encrypted stream without decryption or re-encryption. The major potential computational impact is on the client side where decryption is performed. Since the C&S encryption is simple and fast. Decrypting a large amount of data is not a significant overhead in most applications.

We conclude this section by pointing out that SSLFE and SMLFE are designed for different application scenarios with different features. We do not believe that one algorithm can be replaced by the other.

V. Discussion on Security of SSLFE and SMLFE

As we mentioned in Section I, security for video encryption involves two different aspects. One is the visual effect, i.e., how much visual information leaks after encryption. The other is the system security which is the robustness of the system against cryptanalysis. Tolerance to visual content leakage and judgment of the success of an attack depend on applications. For example, in military and other high security applications, any unauthorized disclosure of visual content, due to either the system design or an attack, may be considered failure for the system. On the other extreme, some video applications may consider partial content leakage acceptable as long as the visual quality for an unauthorized access is much lower than the quality associated with authorized access. In addition to the attacks well studied in cryptanalysis, strong correlation in video may be exploited by some signal processing algorithms such as error concealment techniques in an attack. With this in mind, we would like to have a somewhat intuitive discussion on the security of the proposed algorithms. Rigorous cryptanalysis is much more challenging and is beyond the scope of this paper. The visual effects for SSLFE will be reported and discussed in Section VI.

Both SSLFE and SMLFE use the C&S encryption as the underlying cipher. The security of the C&S encryption plays a critical role in the security of the proposed algorithms. The C&S encryption can be considered as an enhanced stream cipher. Its security depends on the stream and block ciphers used in the algorithm, and the pre-MAC as a keyed hash value. If the field $Z(2^{31}-1)$ is used, it has been proved in [29] that the pre-MAC calculation is an almost 2-universal hash function with a collision probability of 2^{-62}. The 62 bit pre-MAC is encrypted by a block cipher. Break of the block cipher encryption reveals the pre-MAC and the video encryption key in SSLFE or the segment encryption keys in SMLFE which can be used to further reverse the stream cipher encryption. The security of the C&S encryption is therefore at the same level of a 62 bit block cipher encryption in the Electronic Codebook (ECB) mode. In the proposed algorithms, most of the data to be encrypted are processed by the stream cipher in the C&S encryption. The block cipher encrypts much less data.

It is important that a video encryption algorithm is robust to known plain-text attacks since many commercial video sequences start with a well known short sequence. Both SSLFE and SMLFE are robust to known-plaintext attacks, thanks to the pre-MAC, a keyed hash value of the data to be encrypted, used as part of the key to a stream cipher to encrypt the transformed data obtained in the process of the pre-MAC calculation. Secret parameters are used in the calculation. In the proposed algorithms, the C&S encryption is applied to the compressed stream. Compression removes redundancy in a video sequence, and, if we ignore the certain structure in a video format, generates rather random output. Even two identical frames in a video sequence may generate different compressed bit sequences (due to using different references or encoding to different frame types, etc). In the C&S encryption, a single bit difference in a plaintext generates a very different, uncorrelated ciphertext. This content-sensitive encryption on the compressed stream makes known-plaintext attacks difficult. Knowing a plaintext and its corresponding ciphertext does not help much to break other encrypted data unless the known plaintext is repeated, or the encryption key is deduced. To increase security, we recommend that each video sequence be encrypted with a different encryption key in SSLFE or keys in SMLFE, and different secret parameters in calculating pre-MAC. For long video sequences, the keys and secret parameters should be updated on a regular basis.

For identical inputs, the C&S encryption generates identical pre-MACs and outputs. This may be a serious vulnerability in cryptography but is acceptable in typical video applications. In most video applications, a local security compromise, for example, the encryption of a few frames is broken, is acceptable as long as the compromise does not expand extensively. Since redundancy in video sequences is removed by the compression before encryption, and the number of bits in an encryption cell is in the hundreds or more, the chance that identical encryption cells occur is very slim.

Security of SSLFE in SEM is worth further discussion since limited data is actually encrypted. In SEM, the dc values of a known number of bits, the ac coefficient sign bits, the MV sign bits, and the MV residues in the base layer are encrypted. It is shown in [16] that encryption of dc coefficients alone leaves edges of an encrypted frame still comprehensible. Encryption of the ac coefficient sign bits can be applied to only nonzero ac coefficients. In an experiment to count nonzero ac coefficients for the base layer compressed at typical bit rates, we found that when the ac coefficient prediction was turned on, there was on average 1 nonzero ac coefficient per 8×8 block for the QCIF video clip "Miss America" compressed at 30 kbps, and 4.3 nonzero ac coefficients for the QCIF "Coast Guard" at 100 kbps. A brute force attack on the encryption of ac coefficient sign bits requires two trials on average for "Miss America" and about 16 trials for the "Coast Guard" to break an 8×8 block. The enhancement layer, on the other hand, has much more nonzero DCT coefficients and therefore much more states, thanks to its much higher bit rate compression. In the base layer,

TABLE I
PROCESSING SPEED OF SSLFE FOR ENCRYPTING THE BASE LAYER ALONE ("BASE ONLY") OR BOTH THE BASE AND THE ENHANCEMENT LAYERS ("ALL"). THE BASE LAYER IS ENCRYPTED EITHER IN THE SEM OR THE FEM. THE SECOND COLUMN LISTS THE BASE LAYER BITRATE FOR EACH SEQUENCE. THE ENHANCEMENT LAYER IS COMPRESSED TO 2.5 MBPS FOR ALL THE SEQUENCES

Video Sequences	Base Layer Bitrate (Kbps)	SEM (Mbps)		FEM (Mbps)	
		Base Only	All	Base Only	All
Akyio	7.65	25	7905	55	17391
Carphone	24.2	29	3122	82	8828
Coastguard	27.2	31	2978	86	8261
Foreman	32.2	30	2400	90	7200
MissAm	8.62	31	9257	55	16423
Salesman	10.4	29	7201	59	14650

an MV residue is typically small (under 2 bits on average in our experiments). The number of MVs per frame is much less than the number of DCT coefficients. MV sign bits offer limited states, too. Encryption on the MV sign bits and residues does not make a brute force attack much more difficult.

In addition to the limited number of states offered in the base-layer encryption with SEM, the strong correlation in video can also be exploited in an attack. Security for error-concealment-based attacks for encryption of different fields such as MVs, Intra-DC, DCT sign bits, etc has been studied in detail in [20] which concludes that format-compliant selective video encryption has to be combined with a permutation algorithm proposed in the paper to improve robustness against error-concealment-based attacks. This permutation technique can be used in SSLFE to improve the security for the base-layer encryption with SEM, and also the enhancement layer encryption, if higher security is needed. Although the security of SSLFE in SEM is not very high, it generates format-compliant outputs, and can still be used in many applications.

VI. EXPERIMENTAL RESULTS

We have implemented in C++ a demonstration system on top of the MPEG-4 FGS reference code from MPEG that matches the version described in [28]. Our modules were integrated into the compression and decompression process. The C&S encryption was implemented on the filed $Z(2^{31}-1)$ where RC4 [9] was used as the stream cipher and RC5 [9] as the block cipher. RC4 was also used as the stream cipher to encrypt the enhancement layer in SSLFE. For SMLFE and the base-layer encryption in SSLME with FEM, compressed data was passed to our encryption module frame by frame with indicators of the start and the end for each encryption cell. For the base-layer encryption with SSLFE in SEM, bits and location of each field to be encrypted were extracted, and the bits were placed into an encryption buffer during the compression. At the end of a VOP compression, the C&S encryption was applied to the encryption buffer, and fields in the resulting ciphertext were placed back to their original positions. The enhancement layer encryption for SSLFE was carried out in situ during the compression. The processing speeds reported below are the *actual* computational overhead of the proposed algorithms, which accounts for all the additional operations (except operations that are difficult for timing) added to the original compression and decompression operations. This computational overhead includes encryption overhead as well as the overhead to preprocess and postprocess data to be encrypted. For example, the processing speeds reported below for SSLFE in SEM include processing time spent on extracting the fields to be encrypted, placing back these fields after encryption, the C&S encryption, etc. We believe that such a speed is more accurate than the speed of encrypting data alone in measuring computational overhead of an encryption system, and fairer in comparing speeds of different video encryption algorithms. All tests were done on a P3 667-MHz Dell PC with 512-MB memory. Quite a few QCIF color video sequences from MPEG were used in our experiments. The base layer was encoded to the frame rate a third of the original frame rate and at a nominal bit rate around 30 kbps. The actual base-layer bit rate varied greatly from one sequence to another. The enhancement layer was encoded to the bit rate of 2.5 Mbps for all the sequences.

We tested the processing speed of our implemented C&S encryption on data in memory with a typical size of an encryption cell in SSLFE and SMLFE. The average speed, including the stream and block cipher encryption, was about 90 Mbps for either encryption or decryption. This speed seems much slower than the results reported in [29]. We believe that the speed discrepancy can be explained by our C++ implementation of the C&S encryption versus assembly code implementation in [29] and the additional time spent on RC4 and RC5 encryption in our tests.

Table I shows the average processing speeds for the base-layer encryption alone and all (i.e. both base and enhancement layer encryption) encryption for SSLFE in both SEM and FEM. The base-layer bit rates for the sequences are also listed in the table. The number of bits in the base layer processed in SEM range from 12.67% to 15.72% of the total bits in the base layer. An interesting phenomenon we can see from Table I is that the base-layer processing speed in SEM is always slower than that in FEM. This seems to contradict the common belief that selective encryption should always be faster than a full encryption. This can be explained by the fact that our speed data includes the time spent on preparing data for the encryption process. The base-layer encryption in SEM requires additional operations to extract and place back all the fields to be encrypted that encryption in FEM does not have. All these fields are small, typically one to several bits per field. These preparing operations may cost

Fig. 2. Encryption visual effects for SSLFE. Left: original: *Akyio* (top) and *Foreman* (bottom). Middle: selective encryption mode. Right: full encryption mode.

more in time than the saving of bits to be encrypted by a cipher, esp. when a faster cipher like the C&S encryption is used. An in situ encryption by a stream cipher like that used in the enhancement layer encryption in SSLFE can dramatically reduce this preparing overhead, and make a selective encryption faster than a full encryption. In fact, if we exclude the time spent on preprocessing and postprocessing in SEM, the encryption speed is close to the C&S encryption speed reported above, and the base-layer encryption in SEM is indeed much faster than that in FEM, thanks to much less data needing to be encrypted.

The visual effect of *Akyio* and *Foreman* encrypted with SSLFE in both SEM and FEM are shown in Fig. 2. As we can see from *Akyio* in Fig. 2, some of the outline of the speaker in SEM is still partially visible. Much less visual information can be seen for *Foreman* in SEM due to higher motions in the sequence. The frames encrypted in FEM appear very random. Playing these encrypted video sequences showed that the visual content leakage phenomenon described in Section III-A when only the base layer is encrypted disappeared in both SEM and FEM. The sequences encrypted in SSLFE with SEM still revealed some visible structures in the encrypted video. In particular, it was very easily to tell which parts were still and which parts were moving in a video sequence with low motion, such as Akyio.

In the SMLFE experiments, the enhancement layer was divided into four PSNR layers and four bit-rate layers. Sixteen segment keys are used for each tested sequence. The PSNR layers were determined for each video sequence in such a way that each PSNR layer showed perceptible improvement over the next lower layer. The bit-rate layers were almost equally separated in number of bits. The visual effects for *Foreman* at the four PSNR layers are shown in Fig. 3.

We have also tested the error resilience of the proposed SSLFE and SMLFE against MPEG-4 FGS without encryption

Fig. 3. SMLFE visual effects for 4 PSNR layers for *Foreman*. (PSNR layer number, PSNR value) for the frame ordered from top to bottom and left to right is: (1, 27 dB), (2, 31 dB), (3, 36 dB), (4, 45 dB).

by a blind test in which both encrypted (after decryption) and unencrypted video of the same sequence were played side by side in a random order and an observer was asked to select the unencrypted. The original sequence was played before the test of each sequence. Bit errors and packet losses in the enhancement layer were simulated by randomly flipping video data bits or dropping video packets at the same positions (exactly the same positions if the added bits in the encryption were removed). The base layer was not touched. Four people with image processing experiences including one author were involved in the test. Each sequence was tested ten times with each

TABLE II
BLIND TEST RESULTS FOR ERROR RESILIENCE OF SSLFE AND SMLFE. COLUMNS 2 AND 4 SHOW THE AVERAGE OF THE TESTING RESULTS AND COLUMNS 3 AND 5 SHOW THE CORRESPONDING STANDARD DEVIATIONS

Video Sequences	SSLFE		SMLFE	
	Favoring encrypted (%)	Standard deviation (%)	Favoring encrypted (%)	Standard deviation (%)
Akyio	52.86	9.39	51.50	8.78
Carphone	49.61	6.96	53.08	10.50
Coastguard	47.50	12.23	49.72	9.55
Foreman	50.47	7.58	46.02	11.10

person, and each time with independent random bit errors or packet losses. Table II shows the testing results. Note that since the base layer is not touched, SSLFE in either SEM or FEM can be used in the blind test for SSLFE. As we can see from the table, the encrypted and unencrypted are favored almost equally, and the ideal no difference case (50%) is within the sampling error. We conclude that there is no visual degradation for either SSLFE or SMLFE under bit errors and packet losses. This result is expected since the MPEG-4 FGS decoder we used in the tests did not use any extra measures to protect against bit errors. It is possible to see some visual difference for SMLFE if a decoder uses advanced technologies to protect against bit errors.

VII. CONCLUSION

We have presented two novel encryption algorithms for MPEG-4 FGS. The first algorithm, SSLFE, encrypts an FGS stream to a single layer in either the selective or the full encryption mode. The original FGS and error resilience performance of MPEG-4 FGS are fully preserved in the encrypted stream. The second algorithm, SMLFE, encrypts a single FGS stream into multiple quality layers partitioned according to either PSNR or bit rates. The lower quality layers are accessible and reusable by a higher quality layer of the same type, but not vice versa. Both layer types are supported simultaneously. Both SSLFE and SMLFE are fast and have negligible compression overhead. They can be extended to other scalable multimedia formats.

ACKNOWLEDGMENT

The authors would thank Dr. F. Wu, Dr. G. Shen, Dr. X. Sun, M. Su, and Dr. X. Wang for valuable comments, suggestions, and fruitful discussions. They would also like to thank Dr. M. D. Swanson for numerous suggestions and modifications to improve the presentation and readability of this paper.

REFERENCES

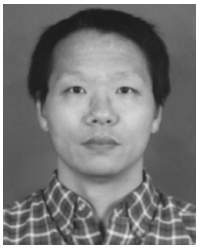
[1] W. Li, "Overview of fine granularity scalability in MPEG-4 video standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 3, pp. 301–317, Mar. 2001.
[2] R. Iannella, "Digital Rights Management (DRM) architectures," *D-Lib Mag.*, vol. 7, no. 6, Jun. 2001.
[3] A. M. Eskicioglu, J. Town, and E. J. Delp, "Security of digital entertainment content from creation to consumption," *Signal Process.: Image Commun.*, vol. 18, no. 4, pp. 237–262, April 2003.
[4] *MPEG-4 IPMP Extension Committee Draft*, ISO/IEC 14496-1:2001/AMD3.
[5] *Study of FPDAM ISO/IEC 14 496-1:2001/AMD3*, ISO/IEC JTC 1/SC 29/WG11 N5068.
[6] Microsoft. Architecture of Windows Media Rights Manager. [Online] Available: http://www.microsoft.com/windows/windows-media/wm7/drm/architecture.aspx
[7] B. M. Macq and J. Quisquater, "Cryptology for digital TV broadcasting," *Proc. IEEE*, vol. 83, no. 6, pp. 944–957, Jun. 1995.
[8] I. Agi and L. Gong, "An empirical study of secure MPEG video transmissions," in *Proc. Internet Society Symp. Network & Distributed System Security*, San Diego, CA, Feb. 1996, pp. 137–144.
[9] B. Schneier, *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, 2nd ed: Wiley, 1996.
[10] T. B. Maples and G. A. Spanos, "Performance study of a selective encryption scheme for the security of networked, real-time video," in *Proc. 4th Int. Conf. Computer Communications & Networks*, Las Vegas, NV, Sept. 1995.
[11] J. Meyer and F. Gadegast. (1995) Security Mechanisms for Multimedia Data With the Example MPEG-1 Video. [Online] Available http://www.gadegast.de/frank/doc/secmeng.pdf
[12] L. Qiao and K. Nahrstedt, "A new algorithm for MPEG video encryption," in *Proc. 1st Int. Conf. Imaging Science, Systems & Tech.*, Las Vegas, NV, Jun. 1997, pp. 21–29.
[13] ——, "Comparison of MPEG encryption algorithms," *Int. J. Comput. & Graph.*, vol. 22, no. 3, 1998.
[14] C. Shi and B. Bhargava, "A fast MPEG video encryption algorithm," in *Proc. ACM Int. Conf. Multimedia*, Bristol, U.K., Sep. 1998, pp. 81–88.
[15] ——, "An efficient MPEG video encryption algorithm," in *Proc. 17th IEEE Symp. Reliable Distributed Systems*, West Lafayette, IN, Oct. 1998, pp. 381–386.
[16] L. Tang, "Methods for encrypting and decrypting MPEG video data efficiently," in *Proc. ACM Int. Conf. Multimedia*, Boston, MA, Nov. 1996, pp. 219–229.
[17] W. Zeng and S. Lei, "Efficient frequency domain video scrambling for content access control," in *Proc. ACM Int. Conf. Multimedia*, Orlando, FL, Oct. 1999, pp. 285–294.
[18] ——, "Efficient frequency domain selective scrambling of digital video," *IEEE Trans. Multimedia*, vol. 5, no. 1, pp. 118–129, Mar. 2003.
[19] J. Wen, M. Severa, W. Zeng, M. H. Luttrell, and W. Jin, "A format-compliant configurable encryption framework for access control of multimedia," in *IEEE Workshop Multimedia Signal Processing*, Cannes, France, Oct. 2001, pp. 435–440.
[20] ——, "A format-compliant configurable encryption framework for access control of video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 6, pp. 545–557, Jun. 2002.
[21] W. Zeng, J. Wen, and M. Severa, "Fast self-synchronous content scrambling by spatially shuffling codewords of compressed bitstreams," in *Proc. IEEE Int. Conf. Image Processing*, vol. 3, Rochester, NY, Sep. 2002, pp. 169–172.
[22] C.-P. Wu and C.-C. J. Kuo, "Efficient multimedia encryption via entropy codec design," in *Proc. SPIE Security and Watermarking of Multimedia Contents III*, vol. 4314, San Jose, CA, Jan. 2001.
[23] S. J. Wee and J. G. Apostolopoulos, "Secure scalable streaming enabling transcoding without decryption," in *Proc. IEEE Int. Conf. Image Processing*, vol. 1, Thessaloniki, Greece, Oct. 2001, pp. 437–440.
[24] R. Grosbois, P. Gerbelot, and T. Ebrahimi, "Authentication and access control in the JPEG 2000 compressed domain," in *Proc. SPIE 46th Annu. Meeting, Applications of Digital Image Processing XXIV*, San Diego, CA, 2001.
[25] H. Cheng and X. Li, "Partial encryption of compressed images and videos," *IEEE Trans. Signal Process.*, vol. 48, no. 8, pp. 2439–2451, Aug. 2000.

[26] C. Yuan, B. B. Zhu, Y. Wang, S. Li, and Y. Zhong, "Efficient and fully scalable encryption for MPEG-4 FGS," in *Proc. IEEE Int. Symp. Circuits and Systems*, vol. 2, Bangkok, Thailand, May 2003, pp. 620–623.

[27] C. Yuan, B. B. Zhu, M. Su, X. Wang, S. Li, and Y. Zhong, "Scalable access control enabling rate shaping without decryption for MPEG-4 FGS video," in *Proc. IEEE Int. Conf. Image Processing*, vol. 1, Barcelona, Spain, Sep. 2003, pp. 517–520.

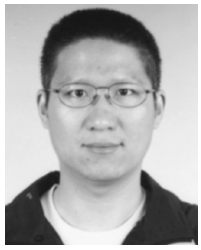
[28] ISO/IEC JTC1/SC29/WG11 N3515, MPEG-4 Video Verification Model Version 17.0, Beijing, China, Jul. 2000.

[29] M. H. Jakubowski and R. Venkatesan, "The chain & sum primitive and its applications to MAC's and stream ciphers," in *Proc. EUROCRYPT'98*, 1998, pp. 281–293.



Bin B. Zhu (S'95–M'97–SM'05) received the B.S. degree in physics from the University of Science and Technology of China in 1986, and the M.S. and Ph.D. degrees in electrical engineering from the University of Minnesota, Minneapolis, in September 1993 and December 1998, respectively.

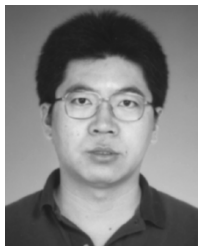
From 1997 to 2001, he was a Lead Scientist at Cognicity, Inc., an entertainment marketing software tools publisher he co-founded. He has been with Microsoft Research Asia, Beijing, China, since December 2001. He has published two book chapters and over 30 peer-reviewed papers in leading journals and conferences in the areas of multimedia encryption, digital rights management, multimedia watermarking and authentication, and data and image compression. He has six issued and ten pending U.S. patents. His current research interests include digital rights management and security, distributed multimedia networks, wireless communications, and multimedia authentication, watermarking, and compression.



Chun Yuan received the M.S. and Ph.D. degrees from the the Computer Science Department, Tsinghua University, Beijing, China, in 2000 and 2003, respectively.

In 2002, he worked in the Internet Multimedia Group of Microsoft Research Asia, Beijing, China, as an intern. He is currently a Postdoctorate Researcher with the SMIS Project of INRIA-Rocquencout, France. His research interests include multimedia security and digital rights management in distributed network, database security and access control, and cryptography algorithms.

Yidong Wang received the B.S. degree from Peking University, Beijing, China, in 2001. He is currently pursuing the M.S. degree in computer science and working on digital rights management of streaming media at Peking University, Beijing, China.



Shipeng Li received the B.S. and M.S. degrees from the University of Science and Technology of China (USTC), Hefei, in 1988 and 1991, respectively, and the Ph.D. degree from Lehigh University, Bethlehem, PA, in 1996, all in electrical engineering.

He was with the Electrical Engineering Department, USTC, during 1991–1992. He was a Member of Technical Staff at Sarnoff Corporation, Princeton, NJ, during 1996–1999. He has been a Researcher with Microsoft Research Asia, Beijing, China, since May 1999, and is now Manager of the Internet Media Group. His research interests include image/video compression and communications, digital television, multimedia, wireless communication, and digital rights management and security. He has contributed several technologies to the MPEG-4 and H.264 international standard.

Dr. Li is a member of the Visual Signal Processing and Communications Technical Committee of the IEEE Circuits and Systems (CAS) Society.