A Joint Signal Processing and Cryptographic Approach to Multimedia Encryption

Yinian Mao, *Student Member, IEEE,* and Min Wu, *Member, IEEE*

***Abstract*—In recent years, there has been an increasing trend for multimedia applications to use delegate service providers for content distribution, archiving, search, and retrieval. These delegate services have brought new challenges to the protection of multimedia content confidentiality. This paper discusses the importance and feasibility of applying a joint signal processing and cryptographic approach to multimedia encryption, in order to address the access control issues unique to multimedia applications. We propose two atomic encryption operations that can preserve standard compliance and are friendly to delegate processing. Quantitative analysis for these operations is presented to demonstrate that a good tradeoff can be made between security and bitrate overhead. In assisting the design and evaluation of media security systems, we also propose a set of multimedia-oriented security scores to quantify the security against approximation attacks and to complement the existing notion of generic data security. Using video as an example, we present a systematic study on how to strategically integrate different atomic operations to build a video encryption system. The resulting system can provide superior performance over both generic encryption and its simple adaptation to video in terms of a joint consideration of security, bitrate overhead, and friendliness to delegate processing.**

***Index Terms*—Delegate processing, multimedia encryption, overhead analysis, security score.**

I. Introduction

THE past decade has witnessed significant advancement in coding and communication technologies for digital multimedia, paving ways to many opportunities for people around the world to acquire, utilize, and share multimedia content [1]. To allow for wider availability of multimedia information and successful commercialization of many multimedia related services, assuring that multimedia information is used only by authorized users for authorized purposes has become essential. This paper focuses on protecting the confidentiality and achieving access control of multimedia information, which is one of the crucial security elements for many applications.

In a typical use of multimedia illustrated in Fig. 1, the owner of the multimedia content wants to distribute the content through networks or archive it for future use. With the sophistication of heterogeneous networks and the growing amount of information being generated, it is becoming less efficient for the content owners to manage the distribution or archiving process all by themselves. As a result, third-party service providers, equipped with specialized servers, huge disk space, and abundant bandwidth resources, will serve as *delegates* for content owners to perform content distribution, archiving, search and retrieval. On one hand, the delegate service providers often need to process the received media data, such as adapting the rate of the media data according to the available bandwidth. On the other hand, the owner may not want to reveal the media content to these delegates because of security and privacy concerns. One such example is the privacy-preserving data retrieval using untrusted server [2].

A common way to achieve content confidentiality is to encrypt the entire multimedia sequence using a cipher, such as DES, AES, or RSA [3], [4]. However, many types of processing, such as rate adaptation for multimedia transmission in heterogeneous networks [5] and DC-image extraction for multimedia content searching [6], cannot be applied directly in the bitstream encrypted by these generic encryption tools or their simple variations. This implies that the processing would still require the delegates to hold the decryption keys to decrypt the content, process the data, and then re-encrypt the content. Since revealing decryption keys to potentially untrustworthy delegates is often not in line with the security requirements of many applications, generic encryption alone is inadequate in the delegate service scenario.

Another unique issue with multimedia encryption is the relation between value, quality, and timeliness of the data. Unlike the "all-or-nothing" protection for generic data, the value of multimedia and in turn its security protection are closely tied with the perceptual quality and the timeliness of the content. A popular sports game, for example, requires paramount protection at the show time to ensure revenue from the viewership, but the demands of security reduce quickly or even vanish over the following days. Additionally, high-quality content is often priced at premium rate and has restricted access, while its low-resolution, low-quality versions are moderately priced or given out for free to reach a wide audience [7].

The first step toward addressing the aforementioned issues is to design flexible multimedia encryption schemes that can handle delegate processing and achieve access control by content and quality [8]. In this paper, we focus on achieving content confidentiality during multimedia distribution, archiving, and other delegate processing. Our goal is to design encryption tools for multimedia that can *encrypt once and securely*

Manuscript received July 13, 2004; revised July 13, 2005. This work was supported in part by the U.S. National Science Foundation ung Grant CCR-0133704 (CAREER) and in part by the U.S. Army Research Office under Grant #DAAD19-01-1-0494 (URI). Preliminary results of this paper were presented at the IEEE Multimedia Signal Processing Workshop (MMSP'02) and the International Conference on Image Processing (ICIP'04). The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Gaurav Sharma.

The authors are with the Department of Electrical and Computer Engineering, University of Maryland, College Park, MD 20742 USA (e-mail: ymao@eng.umd.edu; minwu@eng.umd.edu).

Digital Object Identifier 10.1109/TIP.2006.873426

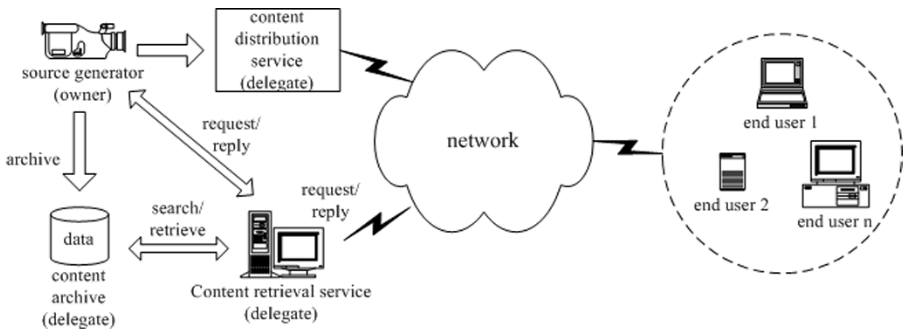


Fig. 1. Typical usage of multimedia content.

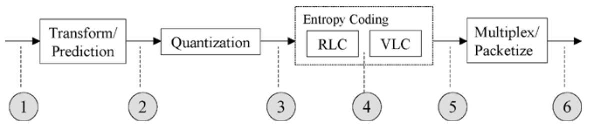


Fig. 2. Candidate domains to apply encryption to multimedia.

process in many ways using the existing multimedia signal processing techniques. To achieve this goal, we jointly consider signal processing and cryptography in our exploration of multimedia encryption. In particular, we investigate the possible domains in which encryptions can be applied, including the sample domain, the quantized transform domain, the intermediate bitplanes, and the bitstream domain. We propose and analyze two atomic encryption operations tailored for multimedia signals, namely, a generalized index mapping encryption tool with controlled overhead and an intrabitplane encryption tool compatible with fine granularity scalable coding. A video encryption system, which incorporates the proposed operations and other relatively straightforward extensions of generic encryption, is then studied. The resulting system takes into consideration the structure and syntax of multimedia sources and protects the content confidentiality during delegate processing.

The rest of this paper is organized as follows. Section II introduces the possible domains of encryption and reviews prior works. We then propose and analyze two atomic operations, namely, generalized index mapping with controlled overhead in Section III, and constrained shuffling for bitplanes in Section IV. In Section V, we discuss the security evaluation for multimedia encryption. Section VI presents experimental results of an encryption system that combines the proposed operations with other extensions of generic encryption, and provides performance comparison of different encryption configurations. Finally, the conclusions are drawn in Section VII.

II. Background and Preliminaries

We examine in this section the possible domains in which encryption can be applied to multimedia, along with a review of prior work. Using a widely adopted multimedia coding framework, we illustrate the candidate domains for applying encryption to multimedia in Fig. 2.

A. Encryption Before and After Coding

According to Fig. 2, there are two straightforward places to apply generic encryption to multimedia. The first possibility is to encrypt multimedia samples before any compression (i.e., Stage 1 in Fig. 2). The main problem with this approach is that the encryption often significantly changes the statistical characteristics of the original multimedia source, resulting in much reduced compressibility. It is worth noting a novel approach has been recently proposed to efficiently compress encrypted data [12]. By employing distributed source coding theory, this new method achieves the same compression gain as compressing the unencrypted data in the case of ideal Gaussian source. The compression gain, however, would be reduced for more general source that is common in practice, and it cannot easily support many other forms of delegate processing.

The second possibility is to apply generic encryption to the encoded bitstream after compression (i.e., Stages 5 and 6 in Fig. 2) [13]. This approach introduces little overhead, but may destroy the structures and syntax readily available in the unencrypted bitstream. Such structures, often indicated by special header/marker patterns, would enable many kinds of processing in delegate service providers and intermediate network links, such as bandwidth adaptation, unequal error protection, and random access [14]–[17].

As headers and markers are special bit patterns in a compressed bitstream for a variety of purposes [16], a simple way to realize syntax-aware encryption is only to encrypt the content-carrying fields of the compressed multimedia bitstream, such as the fields of motion vectors and DCT coefficients in MPEG video, and keep the structure and headers/markers of the bitstream unchanged [19], [20]. However, this method can only preserve limited syntax from unencrypted media to facilitate a fixed set of processing. Other advanced features in multimedia signal processing, such as the fine granular scalable (FGS) coding [5], [9], [10], cannot be easily preserved using this approach. Bitrate overhead also occurs in the range of 1%–8% due to block padding and the added side information. Furthermore, each tailored encryption technique may also require different handling by delegate processing units. This is not always realistic, because the processing units are likely coming from different vendors and the barrier to standardization seems insurmountable due to market considerations [11].

After applying generic encryption, some parts of the encrypted media data could become identical to certain headers/markers. This emulation problem can bring potentially serious troubles to delegate service providers and intermediate processing modules [17] when the multimedia data go through certain network protocols, transcoding, and error recovery.

One possible remedy to header emulation is bit-stuffing [18], a technique that is widely adopted in the packetization stage of network communications [21].

B. Encryption at Intermediate Stages of Coding

Recently, there have been interests in studying how to encrypt multimedia data in such a way that the encrypted data can still be represented in a meaningful, standard-compliant format [22]. They are particularly useful for secure delegate services and multimedia communications that prefer handling media streams compliant to certain multimedia coding standard, such as JPEG or MPEG-1/2/4 standard [17], [23]. The encryption is performed in the intermediate stages illustrated in Fig. 2. For example, at Stage 2, the motion vectors in video can be encrypted by applying DES to their codeword indices [17]. At Stage 3, DC and selected AC coefficients in each block of a JPEG image or an MPEG video frame can be shuffled within the block [24], or across blocks but within the same frequency band [25]. At Stage 4, the entropy codeword can be spatially shuffled within the compressed bitstream [23]; the Huffman codewords of coefficients and/or motion vectors can be encrypted by alternating between several Huffman codebooks in a cryptographically secure fashion [27]. At Stage 5, only intra-coded frames and blocks of an MPEG video are selected and encrypted using a classic DES cipher [26] or its variations [13]. Some of these schemes are also known as *selective encryption* [23], [26], [27], i.e., they encrypt only portions of multimedia data stream that carry rich content, in hope of alleviating the problem of high computational complexity and the potential bitrate overhead. However, we believe that the computational complexity in encryption is not a major concern given the fast-growing computation power and the efficient implementation of established generic encryption tools. In contrast, the protection of content confidentiality under different processing scenarios, such as delegate services and communications, is of paramount urgency. Unfortunately, few existing work has thoroughly considered these scenarios.

In the next two sections, we propose two general atomic encryption operations using index mapping and constrained shuffling to achieve confidentiality protection in delegate services and other processing scenarios. The rationale is to ensure that the encrypted bitstream still complies with the state-of-the-art multimedia coding techniques. We are particularly interested in how much price in terms of security and compressibility these techniques have to pay to achieve standard-compliant encryption, and we answer this question through analysis and simulations.

III. Generalized Index Mapping With Controlled Overhead

Unlike generic data encryption where the encryption output can take values over the entire data space, joint signal processing and cryptographic encryption requires that the encrypted output should satisfy additional constraints. These constraints are essential to preserve the structure, syntax, and standard compliance that enable delegate processing and leads to communication friendliness. A format-compliant encryption scheme was proposed in [17] by assigning a fixed-length index to each variable length codeword (VLC), encrypting the concatenated indices, and then mapping the encrypted indices back to codeword domain to form an encrypted bit-stream. This prior approach would work well with such codes as the Huffman codes and the Golomb–Rice codes, which associate each symbol coming from a finite set with a unique codeword of integer length, but it cannot be directly applied to VLCs that allow fractional codeword length per symbol, such as the arithmetic codes. In addition, this prior encryption work incurs a substantial amount of bitrate overhead, and analytic study has not been provided regarding the overhead. In this section, we construct and analyze an encryption tool that can overcome these two problems.

A. Generalized Index Mapping

We extend the index encryption idea to apply encryption directly to symbols that take values from a finite set before getting into VLC codeword domain. Examples include working with quantized coefficients and quantized prediction residues (Stage 3 in Fig. 2), as well as run-length coding symbols (Stage 4 in Fig. 2). The encryption process to produce a ciphertext symbol $X^{(\mathrm{enc})}$ from a clear-text symbol X is shown as follows:

$$X^{(\mathrm{enc})} = Enc(X) \triangleq T^{-1}\left[\mathcal{E}\left(T(X)\right)\right] \tag{1}$$

where $\mathcal{E}(\cdot)$ is a core encryption primitive such as AES or one-time pad [3], and $T(\cdot)$ represents a codebook that establishes a bijective mapping between all possible symbol values and indices represented by binary strings. The goal of this bijection is to produce fixed-length indices that will be passed to subsequent encryption or decryption. The decryption process has a similar structure

$$X = Dec\left(X^{(\mathrm{enc})}\right) \triangleq T^{-1}\left[\mathcal{D}\left(T\left(X^{(\mathrm{enc})}\right)\right)\right] \tag{2}$$

where $\mathcal{D}(\cdot)$ is a core decryption primitive corresponding to $\mathcal{E}(\cdot)$.

As a simple example, we consider encrypting a string of symbols coming from a finite set $\{\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D}\}$. The symbol sequence to be encrypted is "ABBDC." We first assign a fixed-length index to each symbol

$$\begin{array}{llll} A \rightarrow [00], & B \rightarrow [01] \\ C \rightarrow [10], & D \rightarrow [11]. \end{array}$$

We then convert the symbol sequence to an index sequence "00 01 01 11 10" and encrypt the index sequence using an appropriate encryption primitive such as a stream cipher (the one-time pad) with a random bit-stream [0100 1011 1001 ...]. Finally, we convert the encrypted index sequence "01 01 11 00 00" back to symbol sequence "BBDAA." After encryption, any appropriate VLC coding can be applied to the encrypted symbol sequence. It is worth noting that in such an encryption one input symbol can be mapped to different encrypted cipher-text. For instance, in the previous example, the symbol B has appeared in the clear-text sequence twice, the first time it was mapped to B and the second time to D.

When processing a large sequence of symbols, the encryption method by index mapping tends to make the encrypted symbols uniformly distributed, which is good in terms of security

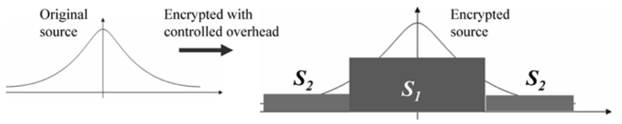


Fig. 3. Index mapping within subsets gives piecewise constant approximation of the distribution.

[3]. However, the entropy of the encrypted symbols is increased from the unencrypted ones. Since the compressibility of a sequence of symbols using entropy coding depends on the entropy of the source symbols [28], the index-mapping encryption would bring a bit-rate overhead in compression. Next, we discuss how to quantify and control this overhead.

B. Analysis and Control of Overhead

In the following analysis, we investigate the impact of the index encryption on the compressibility of the source symbols, which can be quantified by the changes in average code length before and after encrypting a sequence of symbols.

1) Case 1: We consider compressing the source symbols using a default entropy codebook as provided by many multimedia standards. The default codebook is obtained from a set of representative training samples and is used most of the time for the simplicity of implementation. We denote the probability mass function of the symbols prior to encryption by $\{p_i\}$, that of the symbols after encryption by $\{q_i\}$, and the code length designed for distribution $\{p_i\}$ by $\{l_i\}$. If encryption is performed on an index drawn from the full range of symbol values, the distribution of ciphertext symbols, q, will be uniform over the entire range. Alternatively, if we partition the range of symbol values into mutually exclusive subsets $\{S_j\}$ and restrict the outcome of the encryption of a symbol $x \in S_j$ to be within the subset S_j, i.e., $Enc(x) \in S_j$, the distribution q will be a piecewise uniform approximation of p, as illustrated in Fig. 3.

Consider $\{q_i\}$ to be a piecewise uniform approximation of $\{p_i\}$, i.e., for each subset S_j from a nonoverlapping partition of the symbols' range, we have $\sum_{i\in S_j} p_i = \sum_{i\in S_j} q_i = |S_j|q^{(S_j)}$, where $q^{(S_j)} \triangleq q_i$ for all $i \in S_j$, and $|\cdot|$ denotes the cardinality of a set. Assuming that encryption changes the symbol distribution from $\{p_i\}$ to the above $\{q_i\}$ and the same codebook of code length $\{l_i\}$ is used both before and after encryption, we can show that the changes of the expected code length δL is

$$\delta L = \sum_i (q_i - p_i) l_i = D(p\|q) + D(q\|r) - D(p\|r) \qquad (3)$$

where $D(\cdot\|\cdot)$ represents the Kullback-Leibler divergence, and r represents a probability distribution of $\{r_i \triangleq P(R=i) = 2^{-l_i}/\sum_k 2^{-l_k}\}$. The derivation is presented in Appendix A.

If we partition the symbol range S into more than one subset and restrict the encryption output to be in the same subset as the input symbol, the complexity of a brute-force attack for each symbol is reduced[1] from $2^{|S|}$ to $2^{|S_j|}$, where S_j is the subset to which the symbol belongs. On the other hand, the overhead is also reduced because in the Kullback-Leibler divergence sense the distance from the original distribution p to the piecewise uniform distribution q is closer than that to a completely uniform distribution. Thus by controlling the set partitioning, we can adjust the tradeoff between the security and the overhead. In addition, (3) suggests that the optimality of the codelength $\{l_i\}$ designed for probability distribution $\{p_i\}$ affects the changes of the expected code length after encryption. If the code is optimal for $\{p_i\}$, i.e., $l_i = -\log p_i$, then $D(p\|r) = 0$ and (3) is reduced to $\delta L = D(p\|q) + D(q\|p)$.

[1]This is equivalent to encrypting fewer bits of the indices. As will be discussed later in this section, symmetric set partitioning can protect the sign bit, which is important to resisting attacks.

2) Case 2: We consider compressing the source symbols using adaptive or universal entropy coding that adjusts itself to the source's distribution. Arithmetic coding and Lempel–Ziv coding are two such examples. Assuming that the adaptive entropy coding can achieve the entropy bound for any source distribution, we exploit the piecewise constant property of $\{q_i\}$ and show in Appendix A that the change of the average codeword length is

$$\delta L = H_{\{q_i\}} - H_{\{p_i\}} = D(p\|q) \qquad (4)$$

where $H_{\{x_i\}}$ is the entropy of a discrete random variable following the distribution $\{x_i\}$. Similar to the first case, this result also indicates that if we partition the symbol range S into more than one subset and restrict the encryption output to be in the same subset as the input symbol, the distribution of encrypted source $\{q_i\}$ can better resemble that of the original source $\{p_i\}$, leading to reduced overhead in compression.

The final result of the relative bitrate overhead (η) also depends on the ratio of the size of the content to be encrypted (B_1) to the overall size of the stream (B). That is

$$\eta = \frac{B_1^{(e)} - B_1}{B} \times 100\% = \eta_e \frac{B_1}{B} \times 100\% \qquad (5)$$

where $B_1^{(e)}$ denotes the size of the encrypted part and η_e is the relative overhead for the part being encrypted. Even if η_e is large as in the case of prior work [17], the overall overhead can be constrained if only a relatively small part of the stream is encrypted. With our proposed technique of set partitioning, the overall overhead can be controlled both through reducing η_e and through maintaining a low B_1/B ratio by selectively encrypting only a portion of the stream (such as the perceptually significant coefficients).

C. Set-Partitioning in Index Mapping

As we have seen, the set partitioning technique can control the bitrate overhead introduced by the index mapping encryption, trading off the resistance against brute-force attacks. The choice of partition also affects the security against estimation attack. Since the encryption flattens the distribution within each subset, no particular clue of the exact unencrypted source value can be inferred from its encrypted value. The best estimate of an unencrypted value X in terms of the mean square error has to resort to its conditional statistical distribution in the subset S_j. It can be shown that, by observing $X^{(\mathrm{enc})} \in S_j$, the minimum mean-square error (MMSE) estimate $\hat{X}$ for X is

$$\hat{X} = \arg\min_t E\left(|X-t|^2 | X^{(\mathrm{enc})}\right) = E(X|S_j) \qquad (6)$$

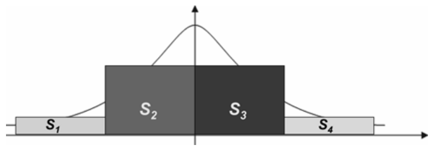


Fig. 4. Set partition with subsets asymmetric to zero.

and the corresponding mean square error is

$$E\left(|\hat{X} - X|^2 | S_j\right) = Var(X|S_j). \tag{7}$$

It has been known that many variables in efficient multimedia representation, including the quantized transform coefficients and prediction residues, follow a symmetric zero-mean distribution such as a Gaussian distribution or a Laplacian distribution. For these variables, if the subset partition is symmetric around zero, as shown in the example of Fig. 3, the MMSE estimate of the original value given the encrypted one will always be zero, regardless of the subset. This means that the sign of the original value is not inferrable. Since the sign, or more generally, the phase, is known to carry important information of a signal [37], an attacker can hardly get any useful information from such MMSE estimation. However, if the partition is not symmetric, for example, as in Fig. 4, the MMSE estimates still preserve the sign and the approximate energy of the original value for each subset. Such a choice of partition can leak a substantial amount of information to an attacker after estimation, and therefore should be avoided.

For several popular entropy coding techniques such as those in JPEG and MPEG, the choice of set partition can be tailored to eliminate the overhead. For example, the JPEG standard employs run-length coding [45], where the run of zero coefficients are coded before the encoding of a nonzero coefficient. If we only encrypt the values of nonzero coefficients and group all coefficient values with the same code-length into the same subset, the code-length of the encrypted value will be identical to that of the unencrypted one. No bitrate overhead will be introduced in this particular case.

D. Examples and Discussions

As an example, we encrypt the DC prediction residues of the JPEG representation of the 512×512 *Lena* image using the index mapping approach. In JPEG, the DC coefficient in each block collectively captures the coarse information of an image and is differentially encoded to reduce the redundancy. For natural images, DC differential residues are approximately Laplacianly distributed with very small probability outside the range of $[-63, 64]$. We encode both the unencrypted and the encrypted prediction residues with the default Huffman table in the JPEG standard. Without encryption, the average code length for encoding DCs is 5.78 bits. In the first encryption experiment, we apply the proposed generalized index encryption to the DC differential residues within $[-63, 64]$ without set partitioning. The index encryption is realized via XORing with a one-time pad, resulting in an average code length of 8.60 bits, or an overhead of 2.82 bits. In the second encryption experiment, we partition the symbol range of $[-63, 64]$ into two subsets $[-31, 32]$ and $[-63, -32] \cup [33, 64]$, and restrict the input and output of index

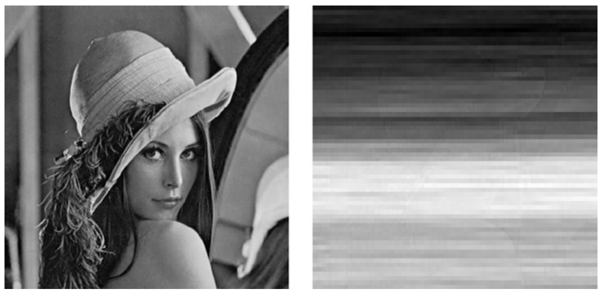


Fig. 5. Encryption results on the Lenna image based on generalized index mapping of DC differential residues: (left) original and (right) encrypted.

encryption to be in the same subset. Fig. 5 shows the encryption result of the Lenna image.[2] With set partitioning, the overhead in average code length is reduced from 2.82 to 1.53 bits.

IV. Constrained Shuffling

Random permutation or shuffling is a common cryptographic primitive operation. The temporal characteristic of audio and video data as well as the spatial characteristic of visual data make permutation a natural way to scramble the semantic meaning of multimedia signals. Even before the arrival of digital technology, an early work by Cox *et al.* builds an analog voice privacy system on a subband representation framework and permutes time segments of subband signals across both time and frequency [40]. More sophisticated coding techniques have been employed by modern digital coding systems. Thus, to control the bit-rate overhead and allow for delegate processing, random permutation should be performed in a constrained way and in appropriate domains. In this section, we use the encryption of scalable video with fine granularity as an example to illustrate the proposed constrained shuffling technique, which is known as the *intra bitplane shuffling*. This encryption technique is compatible with fine granularity scalable coding and provides a tool for access control of multimedia content at different quality levels.

A. Intra Bitplane Shuffling (IBS)

Fine granularity scalability (FGS) is desirable in multimedia communications to provide a near-continuous tradeoff between bitrate and quality. FGS is commonly achieved by bitplane coding, as used in the embedded zero-tree wavelet (EZW) coder [10] and the MPEG-4 FGS coder [9]. We shall use the MPEG-4 FGS to illustrate the concept and the approach can be extended to other FGS coders. As surveyed in [5], MPEG-4 FGS is a functionality provided by the MPEG-4 streaming video profile. A video is first encoded into two layers, namely, a base layer that provides a basic quality level at a low bit rate and an enhancement layer that provides successive refinement. The enhancement layer is encoded bitplane by bitplane from the most significant bitplane to the least significant one to achieve fine granularity scalability. Each bitplane within an image block

[2]Encrypting DC alone is not secure enough as an attacker can still get the edge information by setting the DCs to constant and observing the resulting image. We only encrypt DC in this experiment for the purpose of demonstrating the proposed approach as one potential building block. We will show in Section VI that a complete encryption system should encrypt both DCs and other information.

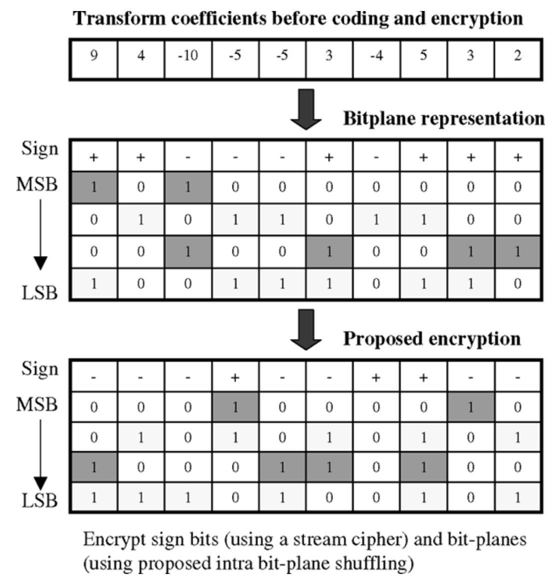

Transform coefficients before coding and encryption
9 4 -10 -5 -5 3 -4 5 3 2
Bitplane representation
Sign
MSB
LSB
Proposed encryption
Encrypt sign bits (using a stream cipher) and bit-planes (using proposed intra bit-plane shuffling)

Fig. 6. Illustration of intra bitplane shuffling.

is represented by (R_i, EOP_i) symbols, where R_i is the run of zeros before the ith "1," and EOP_i is an end-of-plane flag indicating whether the current "1" is the last bit with value 1 in the current bitplane. The run-EOP symbols are encoded using variable-length codes and interleaved with sign bits.

To provide access control to the FGS encoded enhancement layers, the index-based encryption discussed in Section III can be applied to each run-EOP symbol, and the overhead can be analyzed using (3) or (4). We now present an alternative encryption by shuffling each bitplane according to a set of cryptographically secure shuffle tables.

Fig. 6 illustrates the proposed intrabitplane shuffling. We perform random shuffling on each bitplane of n bits and the shuffled bitplane will then be encoded using the run-EOP approach. For example, the first unencrypted bitplane[3] in Fig. 6 "1 0 1 0 0 0 0 0 0 0" has $n_1 = 2$ bits of value "1" out of a total of $n = 10$ bits, which will lead to $\binom{n}{n_1} = 45$ different permutated patterns. In addition to bit plane shuffling, the sign bit s_i of each coefficient is randomly flipped according to a pseudo-random bit b_i from a one-time pad, i.e., the sign remains the same when $b_i = 0$ and changes when $b_i = 1$.

B. *Analysis of Overhead*

An important property of shuffling is that the set of elements before and after shuffling are identical. For intrabitplane shuffling, this implies that the number of "1s" is preserved. Thus the number of run-EOP symbols representing the encrypted bitplane is unchanged, ensuring no overhead coming from the increase in the number of run-EOP symbols. We denote by N_d the number of occurrences that the run of zeros before a "1" equals to d after shuffling. Assuming each of the $\binom{n}{n_1}$ shuffles is equally likely for a specific n-bit bitplane with a total of n_1 bits of "1," we show in the Appendix B that N_d has expected value

$$E(N_d|n_1) = \left(\prod_{k=0}^{d-1}\left(1-\frac{n_1}{n-k}\right)\right)\frac{n_1^2}{n-d}. \tag{8}$$

[3]We use "the first bitplane" to denote the MSB bitplane throughout this paper.

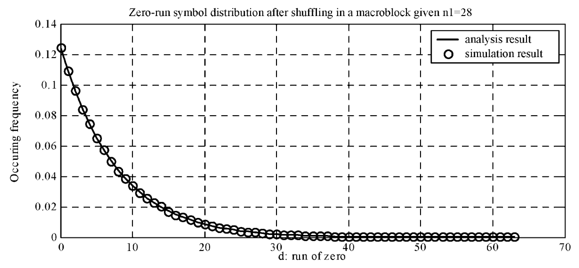


Fig. 7. Expected histograms of zero-run lengths after intrabitplane shuffling: Solid line indicates the analytic result and circles indicate the experimental result; the number of "1s" per macroblock (n_1) is 28 bits, which is the average from the 2nd MSB bitplane of the "Foreman."

We, therefore, arrive at an expected histogram $\{E(N_d|n_1)\}$ versus d, which suggests the likelihood of getting each possible run length.

Shuffling can also be done in a block larger than the block for run-EOP encoding. For example, we can shuffle a macroblock of $n = 256$ bits and perform run-EOP encoding in a smaller block of $n_B = 64$ bits. In this case, the expected zero-run histogram within an encoding block will become

$$E(N_d|n_1) = \prod_{k=0}^{d-1}\left(1-\frac{n_1}{n-k}\right)\cdot\frac{n_1}{n-d} \cdot\left[n_1 - \frac{(n-n_B)(n_1-1)}{n-d-1}\right]$$

which will be reduced to (8) if $n = n_B$. From the histogram before and after encryption we can arrive at the expected overhead per symbol.

As a proof-of-concept, we experiment on the FGS encoded enhancement layer of two video sequences in QCIF format (176×144 pixels per frame). One is ten frames from the "*Foreman*" sequence, and the other is 100 frames the "*Carphone*" sequence. We use intrabitplane shuffling to encrypt each bitplane of the enhancement bitstream, where shuffling is performed on a macroblock ($n = 256$) followed by encoding block by block ($n_B = 64$). The expected histograms $\{E(N_d|n_1)\}$ is presented in Fig. 7, which also shows that the experimental results match the above analytic results very well.

To compare different encryption approaches, we use the "*Foreman*" sequence and encrypt the first three most significant bitplanes, which provides sufficient visual scrambling on the enhancement layer. We have found that for the "*Foreman*" sequence, the intra bitplane shuffling approach gives an overhead of 7.0%, while the overhead by the index-mapping approach in Section III is 14.3%. In general, the bitrate overhead for each bitplane by the proposed shuffling approach depends on the overhead of each run-length symbol and the number of symbols (n_1), as reflected in the plot of the overhead contributed by each bitplane in Fig. 8. The arc shape is a result of an increase in symbol number from MSBs to LSBs and a decrease in overhead per symbol. From Fig. 8 we can also see that some videos, such as "*Foreman*," have very few number of "1s" in the MSB. Therefore the overhead from encrypting the MSB is very small. This also suggests that the MSB alone does not contribute to the perceptual quality significantly in these sequences, and more bitplanes should be protected.

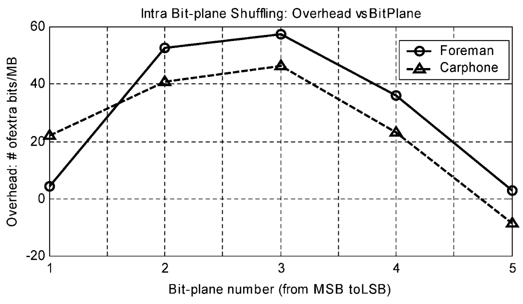


Fig. 8. Overhead of each bitplane by the proposed intrabitplane shuffling algorithm.

C. The Security of IBS

The security of intra bitplane shuffling is affected by how shuffle tables are generated and used. Shuffle tables can be generated from a cryptographically strong pseudorandom sequence using a classical linear-complexity algorithm [43]. Different shuffle tables should be used for different bitplanes, which forces attackers to resort to a brute-force search to simultaneously guess the permutation table used in multiple bitplanes. Similarly, the shuffle tables should be updated frequently without reuse. The amount of brute-force trials for finding the exact clear-text of an enhancement frame is proportional to

$$\prod_{i=1}^{N_{blk}} \prod_{j=1}^{N_{bp}} \binom{n}{n_1^{(i,j)}}$$

where $n_1^{(i,j)}$ is the number of "1s" in the jth bitplane of the ith block, and N_{blk} and N_{bp} represent the number of blocks and bitplanes, respectively. As an example, we consider recovering exactly the second bit plane in the "*Foreman*" sequence. This bit plane is important because adding it to the base-plus-MSB video improves the PSNR from 29 dB to 33.4 dB. In the second bit plane, the average number of "1s" in each shuffled 8×8 block is about 7 in the "*Foreman*" sequence. Suppose each shuffled block has 7 bit of "1s" in a bit plane with QCIF size, the number of brute-force trials an attacker has to perform is proportional to $\binom{64}{7}^{396}$, which is equivalent to guessing approximately 10^5 encrypted bits. In this situation, a brute-force attacker would rather guess the cryptographic key used in generating shuffling tables, which is 128 bits long.

Another security aspect is how many bit planes should be encrypted in order to provide sufficient protection. From our study using a number of video sequences, we found that when the sign information and the first three bit planes are unknown, adding lower bit planes to base layer video will degrade the quality of base layer video, both perceptually and in terms of PSNR. This is because without the higher bit planes and the sign information, the lower bit planes behave like random noise. A similar observation has also been introduced in the streaming video literature [44]. Another observation from the FGS literature is that the first three bit planes, especially the second bit plane, contribute most to the refinement of the quality [5]. Hence, we believe that encrypting the first three bit planes along with the sign information can provide sufficient protection for most multimedia applications.

D. Other Forms of Constrained Shuffling

Since shuffling auditory signals temporally or visual signals spatially can easily make the shuffled signal unintelligible, random shuffle among self-contained coding units (such as the macroblocks in compressed video), has been a popular tool for multimedia encryption and appears in various forms [13], [23]–[25]. We refer to this encryption method as *coded block shuffling* (CBS). A major drawback for block shuffling alone lies in the fact that the information within a block is perfectly retained. An attacker can exploit the correlation across the blocks (such as the continuity of edges and similarity of colors and textures) and reassemble the shuffled blocks with a moderate number of trials.[4] The reassembly effort can be significantly smaller than the brute force search as many unlikely search directions are pruned [32]. Therefore, block shuffling alone is often not a secure encryption operation. We will incorporate block/macroblock shuffling as a complementary building block to our two proposed operations and explore their combinations in the design of an encryption system in Section VI.

V. Measuring Security for Multimedia Encryption

In this section, we introduce a notion of *multimedia-oriented security* to evaluate the security against approximation recovery. To illustrate this concept, we shall use the security of visual data as an example and propose two visual security scores. The principles behind these security scores can be extended to auditory and other types of multimedia.

A. Approximation Recovery

One simple and common way to evaluate the security is to count the number of brute-force trials in order to break the encryption, which is proportional to $\min\{|\text{ clear-text space }|, |\text{ key space }|\}$, where $|\cdot|$ denotes cardinality. Aside from the brute-force search, there are also notions of security that quantify the security of a system in terms of the amount of resources needed to break it [29], [30]. However, the traditional all-or-nothing situation in generic data security is not always appropriate for measuring the security of multimedia encryption [24]–[26]. Beyond the exact recovery from ciphertext, it is also important to ensure partial information that is perceptually intelligible is not leaked out from the ciphertext. Many forms of multimedia data, such as image, video, audio and speech, contain inherent spatial and/or temporal correlation. The encrypted multimedia content may be approximately recovered based on the syntax, context, and the statistical information known as *a priori*. This is possible even when the encrypted part is provably secure according to some generic security notions. For example, in MPEG-4 video encryption [23], when motion vector fields are encrypted and cannot be accurately recovered, a default value 0 can be assigned to all motion vector fields. This approximation sometimes results in a recovered frame with fairly good quality for frames having a limited amount of motion. Additionally, the statistical information, neighborhood patterns, and/or smoothness criterion can help estimate an unknown area in an image

[4]This is usually true when the block is large enough. For small blocks, such as block of size 2×2, the correlation information between blocks is hard to exploit.

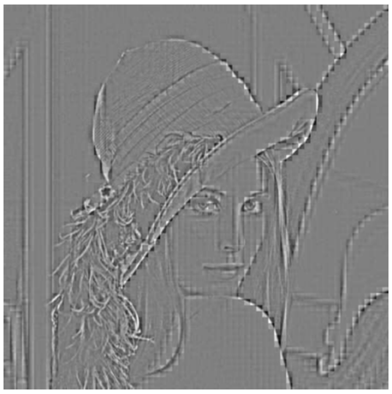


Fig. 9. Approximated *Lena* image by setting all DC coefficients to 0.

[31] and automatically reorder shuffled image blocks [32]. Although these estimations may not be exact, they can reveal perceptually meaningful information once the estimated signal is rendered or visualized.

As an example, we have shown in Fig. 5 the experimental result of encrypting the DC prediction residue of the *Lena* image. Although the directly rendered version of the encrypted *Lena* image in Fig. 5 is highly obscured, an attacker can obtain edge information by setting the DCs to a constant and observing the resulting image shown in Fig. 9. We can see that the edge and contour of the approximated *Lena* image is clearly comprehensible, which suggests that it is necessary to encrypt other components in addition to DCs.

Since the value of multimedia content is closely tied with its perceptual quality, such value composition should be reflected in access control and confidentiality protection. From the considerations presented above, we propose to evaluate the security of multimedia encryption using the following framework. After encryption, the encrypted media is first undergone some approximation attacks. We then use perceptual similarity scores to measure the amount of information leakage about the original media data through the approximated media. The results can indicate the security of the encryption scheme against the approximation attacks. Next, we discuss the methods and tradeoffs of measuring visual similarity for encryption applications.

B. Visual Similarity Scores

Studies on human visual system have shown that the optical characteristic of eyes can be represented by a low-pass filter [35], and that human eyes can extract coarse visual information in images and videos in spite of a small amount of noise and geometric distortion. The important information extracted by human visual system includes spatial-luminance information and edge and contour information [36]. Motivated by these studies, we design a luminance similarity score and an edge similarity score to reflect the way that human perceives visual information. These scores can quantitatively measure the perception-oriented distance between the clear-text copy of multimedia and the attacker's recovered copy from the encrypted media. The proposed scores are inspired by the recent work on automated image quality measurement [33], [34] that incorporated human perceptual properties.

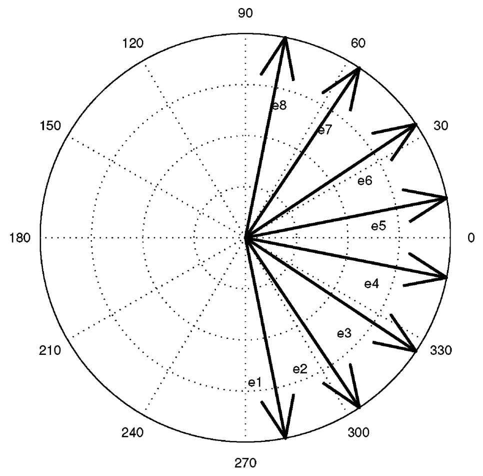


Fig. 10. Eight representative edge directions used in ESS score evaluation.

1) Luminance Similarity Score: To capture the coarse luminance information, we introduce a block-based luminance similarity score. We assume that two given images are preprocessed to be aligned and scaled to the same size. These two images are first divided into blocks in the same way, using 8 × 8 or 16 × 16 nonoverlapping blocks. Then, the average luminance values of the ith block from both images y_{1i} and y_{2i} are calculated. We define the luminance similarity score LSS as

$$LSS \triangleq \frac{1}{N}\sum_{i=1}^{N} f(y_{1i}, y_{2i}). \tag{9}$$

Here, the function $f(x_1, x_2)$ for each pair of average luminance values is defined as

$$f(x_1,x_2) \triangleq \begin{cases} 1, & \text{if } |x_1 - x_2| < \frac{\beta}{2} \\ -\alpha \text{ round } \left(\frac{|x_1-x_2|}{\beta}\right), & \text{otherwise} \end{cases}$$

where the parameters α and β control the sensitivity of the score. Since the images under comparison may be corrupted by noise during transmission or be misaligned by a few pixels, such noise and perturbation should be suppressed during similarity estimation. The resistance to minor perturbation and noise can be achieved by appropriately choosing the scaling factor α and the quantization parameter β. In our experiments, α and β are set to 0.1 and 3, respectively. A negative LSS value indicates substantial dissimilarity in the luminance between the two images.

2) Edge Similarity Score: The edge similarity score measures the degree of resemblance of the edge and contour information between two images. After the images are partitioned into blocks in the same way as in the LSS evaluation, edge direction classification is performed for each block by extracting the dominant edge direction and quantizing it into one of the eight representative directions that are equally spaced by 22.5°, as shown in Fig. 10. We use indices 1 to 8 to represent these eight directions, and use index 0 to represent a block without edge. Denoting e_{1i} and e_{2i} as the edge direction indices for the ith block in two images, respectively, the edge similarity score (ESS) for a total of N image blocks is computed as follows:

$$ESS \triangleq \frac{\sum_{i=1}^{N} w(e_{1i}, e_{2i})}{\sum_{i=1}^{N} c(e_{1i}, e_{2i})}. \tag{10}$$

Here, $w(e_1, e_2)$ is a weighting function defined as

$$w(e_1,e_2) \triangleq \begin{cases} 0, & \text{if } e_1 = 0 \text{ or } e_2 = 0 \\ \left|\cos\left(\phi(e_1) - \phi(e_2)\right)\right|, & \text{otherwise} \end{cases}$$

where $\phi(e)$ is the representative edge angle for an index e, and $c(e_1, e_2)$ an indicator function defined as

$$c(e_1,e_2) \triangleq \begin{cases} 0, & \text{if } e_1 = e_2 = 0 \\ 1, & \text{otherwise.} \end{cases}$$

The score ranges from 0 to 1, where 0 indicates that the edge information of the two images is highly dissimilar and 1 indicates a match between the edges in the two images. A special case arises when the denominator in (10) is zero, which happens when both input images are "blank" without any edge. We assign an ESS score of 0.5 to this special case. In our experiments, the input images are partitioned into nonoverlapping 8 × 8 blocks, and the Sobel operator is used for edge detection [35]. The dominant edge direction of a block is determined by a majority voting inside the block according to the number of pixels associated with each representative direction by the Sobel operator.

C. Evaluation Framework for Multimedia-Oriented Security

When evaluating the image similarity, we first calculate the ESS and LSS scores between the attacked/approximated image and the original image, and then compare the scores with two pre-determined thresholds, ESS_{th} and LSS_{th}, respectively. An encrypted image/video is said to *pass* the similarity test against a certain attack if both the ESS and the LSS are lower than the thresholds. In our experiments, we set ESS_{th} to 0.5 and LSS_{th} to 0.

The proposed similarity scores exhibit a tradeoff between capturing coarse semantic information and texture details of images. The sensitivity of the two similarity scores to image details can be controlled by the size of the block partition. When small blocks (e.g., 4 × 4) are used, a small amount of noise or geometric distortion can result in scores that indicates dissimilarity for two similar images. We refer to this type of misclassification as a *miss*. From security point of view, such a miss would lead to a security breach. When blocks with larger size (e.g., 32 × 32) are used, the scores tend to identify some images as similar when their details are different. We refer to this type of misclassification as a *false alarm*. As preventing information leak is the main concern in many access control applications, usually there are relatively stringent requirements on keeping misses as low as possible, while allowing to tolerate a moderate amount of false alarms. Given these considerations, we suggest using 8 × 8 or 16 × 16 blocks in block partition.

The two similarity scores are intended to measure the amount of information leakage through an attacked image. To this end, other types of perception-based similarity scores, such as robust image hashing [38], can also be incorporated to measure the image similarities, especially on the luminance similarity aspect. These hashing methods measure image similarity through the similarity of hash vectors using the normalized Hamming distance [39] and the receiver-operating-characteristics (ROC) formulation [38].

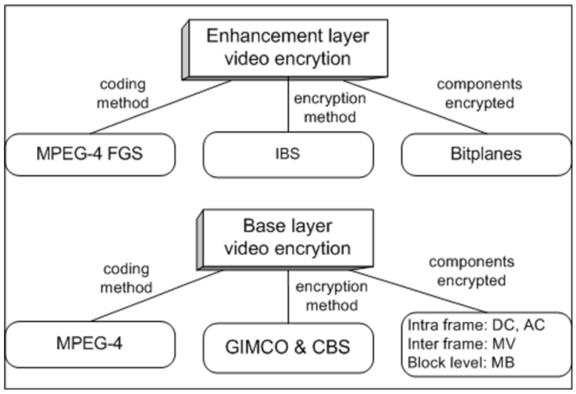


Fig. 11. Video encryption system description. The encryption system is divided into two layers and for each layer candidate encryption components and methods are listed.

The design philosophy of the LSS and ESS scores can be extended to other types of multimedia, such as audio and speech. Similar to the design of audio hash [39], we can first segment an auditory signal into a set of temporal frames, and analyze the components in various frequency ranges from each frame. The results from each interested frequency range (such as low frequency corresponding to LSS, and high frequency corresponding to ESS) can be examined and compared to arrive at an auditory similarity measurement.

VI. Video Encryption System Design

In this section, we present a framework for a video encryption system that employs the building blocks proposed in this paper and from the literature. Using this encryption system, several example configurations are presented and the encrypted videos are compared in terms of the security against brute-force and approximation attack, the friendliness to delegate processing, and the compression overhead.

A. System Setup

Designed with scalable video in mind, the video encryption system has two layers as shown in Fig. 11. The base-layer video is coded with the MPEG-4 standard and the enhancement layer with the MPEG-4 FGS standard. The size of the group of pictures (GOP) is set to 15 and all predicted frames are set to P frames. For each layer, we provide candidate encryption methods and components to be encrypted. Our experiments are conducted on a Dell workstation with 1.8-GHz Pentium IV CPU and 512-MB RAM.

The two encryption operations proposed earlier in this paper, namely, the generalized index mapping with controlled overhead (GIMCO) and the intra bitplane shuffling (IBS), lend themselves naturally as building blocks for this system. A third building block is the coded block shuffling discussed in Section IV-D. The index mapping encryption can be applied to intra block DC/AC coefficients and inter block motion vector (MV) residues, the intra bitplane shuffling encryption can be applied to FGS bitplanes, and the coded block shuffling can be applied to macroblock (MB) coding units. We use AES [41] with a 128-bit key to generate the pseudo-random numbers for all encryptions.

TABLE I
INDEX MAPPING ENCRYPTION OVERHEAD COMPARISON

	Encryption System Settings		*Approximation Attack Settings*
(E1)	encrypting intra block DC residue by index mapping;	(A1)	set all intra block DC coefficients to 0;
(E2)	encrypting inter block MV residue in the first two P frames of a GOP, and all intra block DC residues;	(A2)	set all intra block DC coefficients to 0 and set the encrypted motion vector values to 0;
(E3)	encrypting all the components in E2, plus the first two (in the zig-zag scan order) non-zero AC coefficients of intra block;	(A3)	including all the approximations in A2, plus set the encrypted AC coefficients to 0;
(E4)–(E6)	correspond to E1–E3 plus macro-block shuffling in the compressed bit-stream, respectively;	(A4)–(A6)	the same as A1–A3, respectively.

TABLE II
ENCRYPTION AND ATTACK SETTINGS FOR SECURITY ANALYSIS

	Encryption System Settings		*Approximation Attack Settings*
(E1)	encrypting intra block DC residue by index mapping;	(A1)	set all intra block DC coefficients to 0;
(E2)	encrypting inter block MV residue in the first two P frames of a GOP, and all intra block DC residues;	(A2)	set all intra block DC coefficients to 0 and set the encrypted motion vector values to 0;
(E3)	encrypting all the components in E2, plus the first two (in the zig-zag scan order) non-zero AC coefficients of intra block;	(A3)	including all the approximations in A2, plus set the encrypted AC coefficients to 0;
(E4)–(E6)	correspond to E1–E3 plus macro-block shuffling in the compressed bit-stream, respectively;	(A4)–(A6)	the same as A1–A3, respectively.

B. Bitrate Overhead Study for Index-Mapping Encryption

As discussed in Section III, the index-mapping based encryption introduces bitrate overhead. The overhead can be controlled by carefully choosing the set of components to encrypt and by using the set partitioning technique. In this part, we study the bitrate overhead under different encryption settings. A test video with 4000 frames in QCIF format is constructed by concatenating nine classic video clips, including *Carphone*, *Claire*, *Foreman*, *Grandma*, *Miss America*, *Mother-Daughter*, *Salesman*, *Suzie*, and *Trevor*. After an encryption range is chosen for each component, the range can be further partitioned into two subsets. For example, the encryption range $[-63,64]$ for DC residue can be partitioned into $[-31,32]$ and $([-63,-32] \cup [33,64])$. We also tested to encrypt the first two nonzero AC coefficients along the zig-zag scanning order in each intra block. The encryption range of motion vector residue is $[-16,15.5]$ with half pixel accuracy, which is also the MPEG-4 standard coding range. The encryption ranges of AC coefficients and MV residue can also be partitioned into subsets in a similar fashion as that of DC encryption. Since the MVs are predictively coded, the encryption of MV in one frame is able to propagate the scrambling effect to future frames, suggesting that encrypting MV of a subset of frames would be sufficient. For comparison, we include one experiment in which MVs from the first two P frames in a GOP are encrypted, and another experiment in which all MVs are encrypted. The latter serves as an upper bound for the bitrate overhead by MV encryption.

The above mentioned components are encrypted individually and the compressed file sizes are shown in Table I. From Table I, we can see that the compression overhead incurred by encrypting DC prediction residue ranges from 3% to 7%, depending on the encryption range and whether set partitioning is used. Encrypting AC and MV will generally incur an overhead larger than encrypting DC, especially when the motion vectors from all P frames are encrypted. However, limiting the encrypted MVs to the first two P frames in each GOP can reduce this overhead to around 5%. Also shown throughout Table I is that set partitioning is an effective way to control the overhead. These results provide a basis for designing a video encryption system, which is presented in the next subsections.

C. Base-Layer Video Encryption

In this part, we present experimental results for base-layer video encryption and analyze the security for different configurations. Four video clips, from fast-motion to slow-motion, are used in our experiment. They are the *Football*, the *Coastguard*, the *Foreman*, and the *Grandma*, and each is 40 frames long. Encryption is performed under the settings shown in the left column of Table II, where the encryption of DC, AC, and/or MV is based on the proposed generalized index mapping. The DC and AC encryption ranges are chosen as $[-63,64]$ and $[-32,32]$ with set partitioning, respectively; the motion vector encryption is applied to the first two P frames in each GOP. Additionally, macroblock shuffling in the compressed bitstream is applied to every frame. Each encrypted video complies with the syntax prescribed in the MPEG-4 standard. We also consider approximation attacks to these settings to emulate an adversary's action, and list them in the right column of Table II. The security for these encryption configurations are discussed below in details.

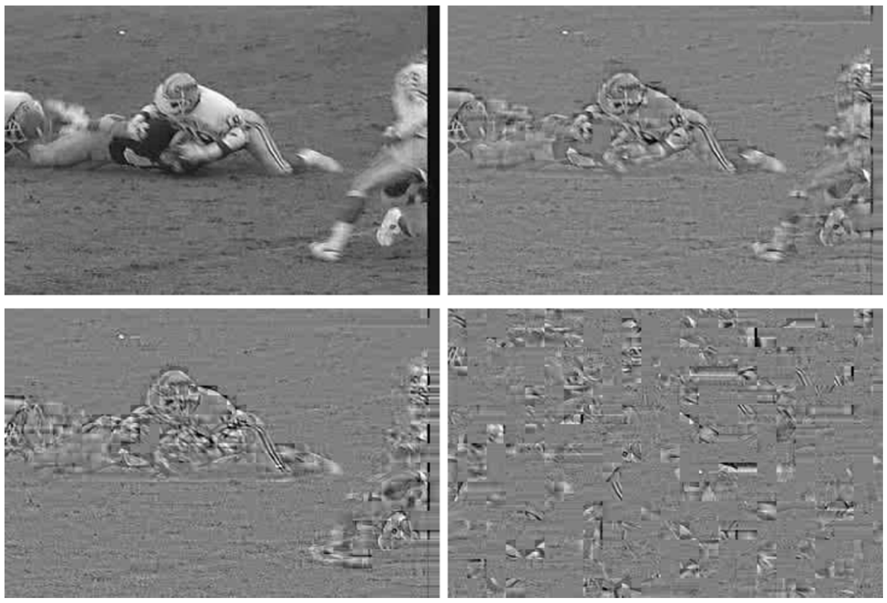


Fig. 12. Encryption results for *Football*. Approximation attacks are performed after encryption. The encryption-approximation settings are: (top left) unencrypted; (top right) E1-A1; (bottom left) E2-A2; (bottom right) E5-A5.

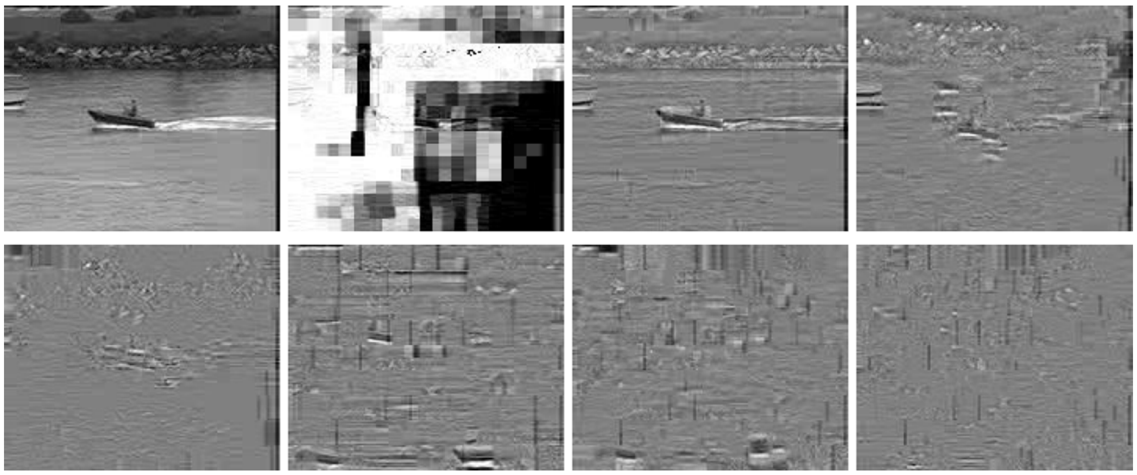


Fig. 13. Encryption results for *Coast Guard*. The encryption-approximation settings are: (top row, left to right) unencrypted, E1, E1-A1, E2-A2; (bottom row, left to right) E3-A3, E4-A4, E5-A5, E6-A6.

1) Security Against Exact Recovery by Exhaustive Search: To accurately recover an original I frame from the encrypted one, an attacker needs to recover all the DC coefficients. For P frames, recovering the values of motion vectors is also necessary. In the above configuration, each DC coefficient and motion vector component has 6 bits encrypted. From the discussion in Section V, each I frame in QCIF format has 2376 equivalent DC bits encrypted and the encrypted motion vectors in a P frame is equivalent to 1188 bits. Since a 128-bit key is used, the security against exact recovery by exhaustive search is determined by the cryptographic primitive with a 128-bit key.

2) Visual Security Against Approximation Recovery: To evaluate the visual security for our encryption system, we first encrypt the test video and then apply approximation attacks to the encrypted video. After that we obtain the ESS and LSS scores of the approximated video and compare them with the thresholds, $ESS_{th} = 0.5$ and $LSS_{th} = 0$. An encryption is considered not secure enough when either score is above the corresponding threshold.

Fig. 12 and Fig. 13 show the video encryption results under different settings for the *Football* and *CoastGuard* clips. The results presented are for Y components as they carry most of the information about the video. Visual examination suggests that encrypting DC alone still leaks contour information after approximation attacks, while extending encryption to MV and/or some ACs helps diffuse the contour to reduce the information leakage. Furthermore, shuffling self-contained coding unit such as macroblocks, coupled with the above value encryption, can scramble the content to a completely unintelligible level. Table III lists the average ESS and LSS (averaged over the total 40 frames) of the videos after approximation recovery. From the average LSS and ESS scores we can see that, when coded block shuffling is not used as an encryption tool, only the LSS score is below its security threshold of 0, and the ESS

TABLE III
PERCEPTION BASED SECURITY MEASURES FOR VIDEO ENCRYPTION

	Football		*Grandma*		*Coastguard*		*Foreman*	
encryption-approximation settings	ESS	LSS	ESS	LSS	ESS	LSS	ESS	LSS
E1-A1	0.70	-0.78	0.64	-2.13	0.79	-1.18	0.71	-1.42
E2-A2	0.53	-0.85	0.46	-2.13	0.43	-1.19	0.43	-1.48
E3-A3	0.53	-0.86	0.30	-2.13	0.40	-1.20	0.40	-1.48
E4-A4	0.12	-0.93	0.05	-2.13	0.07	-1.20	0.07	-1.47
E5-A5	0.13	-0.92	0.05	-2.13	0.06	-1.21	0.06	-1.45
E6-A6	0.12	-0.92	0.04	-2.13	0.04	-1.20	0.05	-1.47

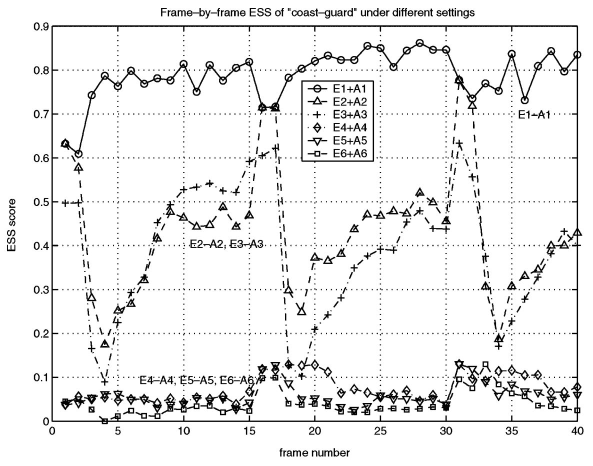


Fig. 14. Frame-by-frame ESS of *Coastguard* video sequence under different settings. The corresponding settings are listed in Table II.

score is around or above its threshold of 0.5. This indicates that the encryption leaks out shape information and is not secure enough, which we can also observe from Fig. 12 and Fig. 13. Once the coded block shuffling is incorporated in the encryption, the ESS and LSS indicate that the encryption is secure against approximations. These results concur with the visual examination.

To examine the detailed ESS scores, we plot the frame-by-frame ESS score of *Coast Guard* under different encryption-attack settings in Fig. 14. The top curve is from the attacked video with DC encrypted only, which confirms that encrypting DC alone still leaves some contour information unprotected. The two middle curves are the results involving MV encryption for inter-blocks and AC encryption for intrablocks, where the ESS scores are low at the beginning of a GOP and increase substantially toward the end of the GOP. This is because as it approaches the end of a GOP, motion compensation becomes less effective and the compensation residue provides a significant amount of edge information. Such observation suggests that if we can only afford the bitrate overhead to encrypt two P frames in a GOP, the two encrypted P frames should be interleaved, such as choosing the 1st and the 8th P frames in a GOP of 15 frames. On the other hand, by incorporating the shuffling of macroblock coding units, the resulting ESS measurements are consistently around 0.1 or lower.

3) Relative Overhead: Table IV lists the compression overhead for four videos under each encryption settings. We can see

TABLE IV
RELATIVE COMPRESSION OVERHEAD OF THE ENCRYPTED VIDEOS

	Football	*Foreman*	*Coastguard*	*Grandma*
E1 and E4	1.29%	1.75%	3.15%	6.96%
E2 and E5	3.88%	6.41%	8.74%	11.11%
E3 and E6	6.47%	9.62%	11.54%	24.61%

that the overhead is low for high-complexity, fast-motion video such as the *Football* and the *Foreman*, and relatively high for low-complexity, slow-motion video such as the *Grandma* and the *Coastguard*. As we go from the setting E1 to E3, more components are encrypted and thus the overhead increases. We also see that the coded block shuffling approach does not introduce overhead, as shown in setting E4 to E6. Overall, the overhead of 4%–11% by the *E1*, *E2*, *E4*, and *E5* is comparable to that of a direct adaptation of generic encryption to multimedia as discussed in Section II-A. Considering both security and compression overhead, we have found that the *E5* setting provides a very good tradeoff. This setting is a combination of block shuffling and selective value encryption via generalized index mapping.

D. Protecting FGS Enhancement Layer Video

We use ten frames from the *Foreman* video sequence to demonstrate the protection of the enhancement data while preserving the FGS characteristics from the source coding. The proposed intra bitplane shuffling encryption is applied within each 8×8 block and the sign bit of each coefficient is encrypted using a stream cipher. To allow for a better visual examination of the protection effects on the enhancement data, we combine the encrypted FGS bitplanes with a clear-text base layer.

For most natural images, the coded DCT coefficients have decreasing dynamic range versus the frequency. As such, we emulate an approximation attack, whereby for each significant bitplane, all the "1s" of the bitplane is put to the lowest possible frequency bins. A total of six encryption-attack settings are used, namely:

a) to shuffle the first bitplane with clear-text base layer;
b) to approximate the bitplane of (a) with the correct signs;
c) to approximate the bitplane of (a) with random signs;
d) to shuffle the first two bitplanes;
e) to approximate the bitplanes of (d) with the correct signs;
f) to approximate the bitplanes of (d) with random signs.

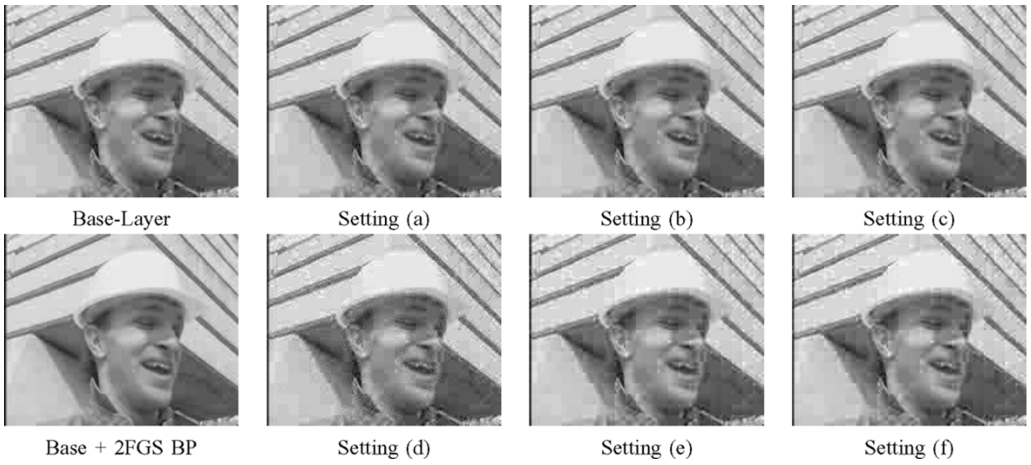


Fig. 15. Encryption results for *Foreman* FGS video. Top row, left to right: base layer picture; and the encryption-attack settings (a), (b), (c). Bottom row, left to right: base layer plus 2 clear-text FGS bitplanes; and the encryption-attack settings (d), (e), (f).

Fig. 15 shows the encrypted and attacked versions of the *Foreman* FGS video. The first row shows the encryption and approximation results using MSB only [settings (a)–(c)], and the second row shows the results using the first two bit planes [settings (d)–(f)]. The blocky artifacts in the settings (d)–(f) are clearly more pronounced than in the settings (a)–(c). This suggests that using more encrypted bit planes will worsen the approximation attack results. Within each row, the visual difference is very subtle. This observation verifies that without knowing the decryption key to the enhancement layer, an attacker cannot obtain a more refined video than the base-layer video using approximations from the encrypted enhancement layer.

Table V lists the corresponding average PSNR, LSS and ESS of the videos under the six encryption-attack settings. From the table, we can see the approximation recovery can only reduce a little luminance error in terms of LSS and PSNR in the approximated video compared to the encrypted video, and the edge similarity in terms of ESS remains imperfect and has little improvement after attack. This demonstrates that the proposed method can protect the premium quality version of the content in a FGS compatible way, with a separate key from base-layer video encryption.

TABLE V
INTRA BITPLANE SHUFFLING AND APPROXIMATION ATTACK

	(a)	(b)	(c)	(d)	(e)	(f)
PSNR (dB)	28.59	28.76	28.74	27.39	27.87	27.50
LSS	0.28	0.34	0.34	0.28	0.34	0.29
ESS	0.85	0.85	0.85	0.85	0.85	0.85

VII. CONCLUSION

In summary, we have shown in this paper the importance and feasibility of incorporating signal processing into multimedia encryption, in order to address the issues of confidentiality protection in multimedia applications. Through two proposed atomic encryption operations that can preserve standard compliance and are friendly to delegate processing, we have provided quantitative analysis and demonstrated that a good tradeoff can be made between the security and bitrate overhead. We have pointed out the need of quantifying the security against approximation attacks that are unique to multimedia, and have proposed a set of multimedia-oriented security scores to complement the security metrics for generic data. Using video as an example, we have presented a systematic study on how to integrate different atomic operations to build a video encryption system. Our experiments have shown that by strategically integrating selective value encryption, intrabitplane shuffling, as well as spatial permutation, the resulting encryption system can achieve a good tradeoff among security, friendliness to delegate processing, and bitrate overhead.

APPENDIX A
DETAILS ON OVERHEAD ANALYSIS FOR GENERALIZED INDEX MAPPING

In this Appendix, we present the derivations for the overhead analysis of generalized index mapping as discussed in Section III. We start with the a default entropy codebook being used to encode encrypted data and prove (3).

Recall that r denotes a probability distribution of $\{r_i \triangleq P(R = i) = 2^{-l_i}/\sum_k 2^{-l_k}\}$. Representing the constant in the denominator as c, we have $l_i = -\log r_i - \log c$. Expanding δL leads to

$$\delta L = \sum_i (q_i - p_i) l_i \tag{11}$$

$$= -\sum_i (q_i - p_i)\log r_i - \sum_i (q_i - p_i)\log c \tag{12}$$

$$= \left(-\sum_i q_i \log r_i + \sum_i q_i \log q_i\right) + \left(-\sum_i q_i \log q_i + \sum_i p_i \log p_i\right) + \left(-\sum_i p_i \log p_i + \sum_i p_i \log r_i\right) \tag{13}$$

$$= D(q\|r) + D(p\|q) - D(p\|r). \tag{14}$$

In the above derivation, the second summation in (12) is zero; and the second term in (14) is obtained by exploiting the piecewise constant property of q such that $\sum_{i\in S_j} p_i = \sum_{i\in S_j} q_i = |S_j|q^{(S_j)}$, leading to

$$\begin{aligned}-\sum_i q_i \log q_i &= -\sum_j \left(\sum_{i\in S_j} q_i\right) \log q^{(S_j)}\\ &= -\sum_j \left(\sum_{i\in S_j} p_i\right) \log q^{(S_j)}\\ &= -\sum_j \left(\sum_{i\in S_j} p_i \log q_i\right)\\ &= -\sum_i p_i \log q_i.\end{aligned}$$

Using the same technique, we can prove (4) for the overhead in the adaptive entropy coding case. That is

$$\begin{aligned}\delta L &= H_{\{q_i\}} - H_{\{p_i\}}\\ &= -\sum_i q_i \log q_i + \sum_i p_i \log p_i\\ &= -\sum_i p_i \log q_i + \sum_i p_i \log p_i\\ &= D(p\|q).\end{aligned}$$

Appendix B
Details on Overhead Analysis for Intrabitplane Shuffling

In this Appendix, we analyze the zero-run symbol distribution after intrabitplane shuffling as discussed in Section IV.

As we have seen in the main text, a macroblock in MPEG4 FGS coding consists of four 8×8 luminance blocks, and zero-run symbols are formed within each block. Intra bitplane shuffling can be done within a block or within a macroblock. For generality, we refer to the block to which shuffling is applied as *shuffling block* and denote its size as n; and similarly, the block to which run-length encoding is applied as *coding block* and its size n_B. In the case of shuffling within an 8×8 block, the two blocks are identical and $n = n_B = 64$; while, for the case of shuffling within a macroblock and run-length encoding within an 8×8 block, we have $n_B = 64$ and $n = 256$. We assume that there are n_1 "1s" in the shuffling block, hence the total number of zero-run symbols is n_1 both before and after shuffling. We derive the zero-run symbol histogram conditioned on n_1.

We use an indicator function $I_k^{(d)}$ to denote the following event of the bitplane of interests for coefficients in a coding block: when ones appear in both kth and $(k+d+1)$th position and zeros in between (i.e. the bits from $(k+1)$th position to $(k+d)$th position forms a zero-run symbol with run-length d), $I_k^{(d)} = 1$; otherwise, $I_k^{(d)} = 0$. The range of k to be considered for $I_k^{(d)}$ is between 0 and $(n_B - d - 1)$, where $I_0^{(d)} = 1$ indicates the first zero-run symbol of the block has run d. The expected number of symbols with zero-run being d in a coding block can be obtained as the follows:

$$\begin{aligned}E(N_{B,d}|n_1) &= E\left[\sum_{k=0}^{n_B-d-1} I_k^{(d)}|n_1\right]\\ &= \sum_{k=1}^{n_B-d-1} \Pr\left(I_k^{(d)} = 1|n_1\right)\\ &\quad + \Pr\left(I_0^{(d)} = 1|n_1\right)\\ &= (n_B - d - 1) \times \frac{\binom{n-d-2}{n_1-2}}{\binom{n}{n_1}} + \frac{\binom{n-d-1}{n_1-1}}{\binom{n}{n_1}}\\ &= \prod_{i=0}^{d-1}\left(1 - \frac{n_1}{n-i}\right) \times \frac{n_1}{n-d}\\ &\quad \times \left[n_1 - \frac{(n-n_B)(n_1-1)}{n-d-1}\right].\end{aligned}$$

When $n = n_B$, the results can be simplified as

$$E(N_d|n_1) = \prod_{i=0}^{d-1}\left(1 - \frac{n_1}{n-i}\right) \times \frac{n_1^2}{n-d}.$$

References

[1] A. Puri and T. Chen, *Multimedia Systems, Standards, and Networks*. New York: Marcel Dekker, 2000.

[2] H. Hacigümüs, B. R. Iyer, and S. Mehrotra, "Efficient execution of aggregation queries over encrypted relational databases," in *Lecture Notes in Computer Science 2973*. New York: Springer-Verlag, 2004, pp. 125–136.

[3] W. Trappe and L. C. Washington, *Introduction to Cryptography With Coding Theory*. Englewood Cliffs, NJ: Prentice-Hall, 2001.

[4] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*. Boca Raton, FL: CRC, 1996.

[5] 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.

[6] B.-L. Yeo and B. Liu, "Rapid scene analysis on compressed video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 5, no. 6, pp. 533–544, Dec. 1995.

[7] F. C. Mintzer, L. E. Boyle, A. N. Cazes, B. S. Christian, S. C. Cox, F. P. Giordano, H. M. Gladney, J. C. Lee, M. L. Kelmanson, A. C. Lirani, K. A. Magerlein, A. M. B. Pavani, and F. Schiattarella, "Toward on-line, worldwide access to vatican library materials," *IBM J. Res. Develop.*, vol. 40, no. 2, pp. 139–162, 1996.

[8] T. Lookabaugh and D. C. Sicker, "Selective encryption for consumer applications," *IEEE Commun. Mag.*, vol. 42, no. 5, pp. 124–129, May 2004.

[9] H. M. Radha, M. van der Schaar, and Y. Chen, "The MPEG-4 fine-grained scalable video coding method for multimedia streaming over IP," *IEEE Trans. Multimedia*, vol. 3, no. 1, pp. 53–68, Mar. 2001.

[10] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Trans. Signal Process.*, vol. 41, no. 12, pp. 3445–3462, Dec. 1993.

[11] J. Baumgartner, "Deciphering the CA conundrum," *Commun. Eng. Design*, Mar. 2003.

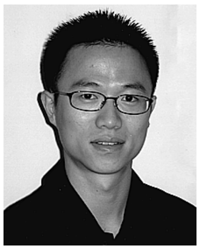
[12] M. Johnson, P. Ishwar, V. Prabhakaran, D. Schonberg, and K. Ramchandran, "On compressing encrypted data," *IEEE Trans. Signal Process.*, pt. 2, vol. 52, no. 10, pp. 2992–3006, Oct. 2004.

[13] L. Qiao and K. Nahrstedt, "Comparison of MPEG encryption algorithms," *Int. J. Comput. Graph.*, vol. 22, no. 3, 1998.

[14] N. Yeadon, F. Garcia, D. Hutchison, and D. Shepherd, "Continuous media filters for heterogeneous internetworking," *Proc. SPIE*, Jan. 1996.

[15] M. Wu, R. Joyce, H.-S. Wong, L. Guan, and S.-Y. Kung, "Dynamic resource allocation via video content and short-term traffic statistics," *IEEE Trans. Multimedia*, vol. 3, no. 2, pp. 186–199, Jun. 2001.

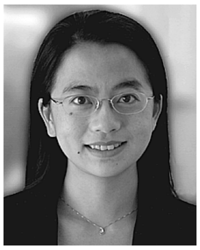
[16] Y. Wang, S. Wenger, J. Wen, and A. Katasggelos, "Error resilient video coding techniques," *IEEE Signal Proess. Mag.*, vol. 14, no. 4, pp. 61–82, Jul. 2000.
[17] J. Wen, M. Muttrell, and M. Severa, "Access control of standard video bitstreams," presented at the Int. Conf. Media Future, Florence, Italy, May 2001.
[18] M. Wu and Y. Mao, "Communication-friendly encryption of multimedia," presented at the IEEE Multimedia Signal Processing Workshop, St. Thomas, U.S. Virgin Islands, Dec. 2002.
[19] S. Wee and J. Apostolopoulos, "Secure scalable streaming enabling transcoding without decryption," presented at the IEEE Int. Conf. Image Processing, Thessaloniki, Greece, Oct. 2001.
[20] C. Yuan, B. Zhu, Y. Wang, S. Li, and Y. Zhong, "Efficient and fully scalable encryption for MPEG-4 FGS," presented at the IEEE Int. Symp. Circuits and Systems, Bangkok, Thailand, May 2003.
[21] A. S. Tanenbaum, *Computer Networks*. Englewood Cliffs, NJ: Prentice-Hall, 2003.
[22] MPEG-21 Part-4: Intellectual Property Management and Protection (Working Document), 2001.
[23] J. Wen, M. Severa, W. Zeng, M. H. Luttrell, and W. Jin, "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.
[24] L. Tang, "Methods for encrypting and decrypting MPEG video data efficiently," in *Proc. 4th ACM Int. Conf. Multimedia*, Boston, MA, Nov. 1996, pp. 219–229.
[25] W. Zeng and S. Lei, "Efficient frequency domain video scrambling for content access control," in *Proc. ACM Multimedia*, Orlando, FL, Nov. 1999.
[26] T.-L. Wu and S. F. Wu, "Selective encryption and watermarking of MPEG video," presented at the Conf. Image Science, Systems, Technology, Las Vegas, NV, 1997.
[27] C.-P. Wu and C.-C. Kuo, "Efficient multimedia encryption via entropy codec design," *Proc. SPIE*, vol. 4314, Jan. 2001.
[28] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, 2nd ed. New York: Wiley, 1991.
[29] M. Bellare, "Practice-oriented provable security," presented at the 1st Int. Workshop Information Security, 1998.
[30] M. Bellare, A. Desai, E. Jokipii, and P. Rogaway, "A concrete security treatment of symmetric encryption," in *Proc. IEEE 38th Symp. Foundations of Computer Science*, 1997.
[31] Y. Wang, Q.-F. Zhu, and L. Shaw, "Maximally smooth image recovery in transform coding," *IEEE Trans. Commun.*, vol. 41, no. 10, pp. 1544–1551, Oct. 1993.
[32] A. Pal, K. Shanmugasundaram, and N. Memon, "Automated reassembly of fragmented images," in *Proc. Int. Conf. Multimedia and Expo*, Baltimore, MD, Jul. 2003.
[33] Z. Wang and A. C. Bovik, "A universal image quality index," *IEEE Signal Proess. Lett.*, vol. 9, no. 3, pp. 81–84, Mar. 2002.
[34] Z. Wang, L. Lu, and A. C. Bovik, "Video quality assessment based on structural distortion measurement," *Signal Process.: Image Commun.*, vol. 19, no. 1, Jan. 2004.
[35] A. K. Jain, *Fundamentals of Digital Image Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1988.
[36] D. J. Field, A. Hayes, and R. F. Hess, "Contour integration by the human visual system: Evidence for a local 'association field'," *Vis. Res.*, vol. 33, no. 2, pp. 173–193, Jan. 1993.
[37] A. V. Oppenheim and J. S. Lim, "The importance of phase in signals," *Proc. IEEE*, vol. 69, no. 5, pp. 529–541, May 1981.
[38] A. Swaminathan, Y. Mao, and M. Wu, "Image hashing resilient to geometric and filtering operations," in *Proc. Multimedia Signal Processing Workshop*, Oct. 2004.
[39] M. K. Mihcak and R. Venkatesan, "A tool for robust audio information hiding: a perceptual audio hashing algorithm," presented at the 4th Int. Information Hiding Workshop, Apr. 2001.
[40] R. V. Cox, D. E. Bock, K. B. Bauer, J. D. Johnston, and J. H. Snyder, "The analog voice privacy system," *AT&T Tech. J.*, vol. 66, pp. 119–131, Jan.–Feb. 1987.
[41] J. Daemen and V. Rijmen, "AES Proposal: Rijndael," [Online], Available: http://csrc.nist.gov/CryptoToolkit/aes/rijndael/Rijndael-ammended.pdf.
[42] D. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," presented at the IEEE Symp. Security and Privacy, May 2000.
[43] D. E. Knuth, *The Art of Computer Programming*, 3rd ed. Reading, MA: Addison-Wesley, 1997, vol. 2.
[44] H. Cai, B. Zeng, G. Shen, and S. Li, "Error-resilient unequal protection of fine granularity scalable video bitstreams," presented at the IEEE ICC, Paris, France, Jun. 2004.
[45] Digital Compression and Coding of Continuous-Tone Still Images. (JPEG), ISO/IEC IS 10918.



Yinian Mao (S'04) received the B.E. degree in electrical engineering from Tsinghua University, Beijing, China, in 2001. He is currently pursuing the Ph.D. degree in signal processing and communications in the Electrical and Computer Engineering Department, University of Maryland, College Park.

He was a Research Intern at Microsoft Research, Redmond, WA, in 2004. His research interests include information security and multimedia signal processing.

Mr. Mao is a coauthor of a paper on media security that won the Student Paper Contest in the 2005 International Conference on Acoustic, Speech, and Signal Processing (ICASSP'05).



Min Wu (S'95–M'01) received the B.E. degree in electrical engineering and the B.A. degree in economics (both with the highest honors) from Tsinghua University, Beijing, China, in 1996, and the Ph.D. degree in electrical engineering from Princeton University, Princeton, NJ, in 2001.

Since 2001, she has been on the faculty of the Department of Electrical and Computer Engineering and the Institute of Advanced Computer Studies, University of Maryland, College Park, where she is currently an Associate Professor. Previously, she was with NEC Research Institute and Signafy, Inc., Princeton, in 1998, and with Panasonic Information and Networking Laboratories, Princeton, in 1999. She co-authored two books and holds five U.S. patents on multimedia security and communications. Her research interests include information security and forensics and multimedia signal processing.

Dr. Wu received a CAREER award from the U.S. National Science Foundation in 2002, a George Corcoran Education Award from University of Maryland in 2003, a TR100 Young Innovator Award from the MIT Technology Review Magazine in 2004, and a Young Investigator Award from the U.S. Office of Naval Research in 2005. She is a co-recipient of the 2004 EURASIP Best Paper Award and the 2005 IEEE Signal Processing Society Best Paper Award. She is an Associate Editor of IEEE SIGNAL PROCESSING LETTERS and served as Publicity Chair of the 2003 IEEE International Conference on Multimedia and Expo. She served as a Guest Editor of the Special Issue on Media Security and Rights Management published in October 2004 by the *EURASIP Journal on Applied Signal Processing*.