On Format-Compliant Iterative Encryption of JPEG2000

Thomas Stütz *
University of Salzburg
Department of Computer Sciences
Jakob Haringerstr. 2
Salzburg, Austria
+43 662 8044 6771
tstuetz@cosy.sbg.ac.at

Andreas Uhl †
University of Salzburg
Department of Computer Sciences
Jakob Haringerstr. 2
Salzburg, Austria
+43 662 8044 6303
uhl@cosy.sbg.ac.at

Abstract

Format-compliant encryption of JPEG2000 has attracted researchers for several years. Most benefits of format-compliant encryption result from the preservation of code-stream features, such as scalability. The possibility of reducing the complexity of encryption and to realize transparent encryption schemes are among the additional features of format-compliant selective encryption. Wu and Deng have proposed a format-compliant iterative encryption scheme for JPEG2000 on a CCP basis. In this paper we discuss whether it is possible to extend this approach to packet bodies and give an exact formula for the expected computational effort. The theoretical results are cross-verified by an experimental survey.

1. Introduction

The encryption of visual data has already been the topic of a considerable amount of research. Many contributions have been made for the encryption of the MPEG and JPEG standards [7] [1] [8]. An overview of image and video encryption can be found in [10]. The advantage of format-compliant encryption results from the preservation of compressed code-stream features in the encrypted domain. The JPEG2000 code-stream has many useful features, e.g., scalability and error resilience, which can be preserved through according encryption schemes. The preservation of the scalability of the compressed image data makes it possible to conduct rate adaption in the encrypted domain without even needing a decryption key. The rate adaption can be simply realized by dropping the enhancement quality packets. This property is of great benefit to the secure distribution of visual data, e.g. rate adaption for the varying bandwidth in wireless streaming applications.

Several approaches for format-compliant encryption of JPEG2000 have been proposed [5] [11] [4] [3] [13] [14] and there is also an upcoming amendment to the JPEG2000 standard (Part-8) covering this topic. All format-compliant code-stream encryption methods propose the encryption of the JPEG2000 packet bodies that contain the compressed coefficient data. In the packet bodies only the compressed image content is stored, while the other parts of the JPEG2000 file can be considered meta information, e.g. compression parameters, data structure, etc. The code-stream syntax imposes certain requirements on the packet data format, which can not be met by standard encryption methods. The iterative encryption approach of Wu and Deng [14] is capable of encrypting 100% of the packet body data while not producing any superfluous information. Other approaches encrypt only parts of the compressed coefficient data. The approaches of Conan [11] and Kiya [3] encrypt at most every half of a byte, while the two approaches of Wu and Ma [13] encrypt all bytes but the `0xff` byte and its successor. The approach presented by Apostolopoulos, Dufaux, e.a. [2] is capable of encrypting a higher ratio because it only preserves the `0xff` byte. The approach of Wu and Deng [14] works on CCPs (code-block contributions to packet). The borders of the CCPs have to be determined rather costly. Since the borders of the packet bodies can be easily found when applying according coding settings, approaches based on packet bodies can be implemented very efficiently. Hence it is of interest whether the iterative encryption approach can be applied to packet bodies or not.

In section 2 a short summary of the JPEG2000 features with a special focus on its code-stream syntax is given. The iterative encryption approach is presented in section 3 and the expected computational cost is determined. Sections 4

*The support of the Austrian Grid project is gratefully acknowledged.
†This work has been partially supported by the Austrian Science Fund, project no. 15170.

and 5 are dedicated to experiments. In section 4 we present statistical experiments which strongly suggest the correctness of the presented stochastic model, while in section 5 the actual CPP and packet lengths and their distributions are analyzed for various coding settings and a broad range of images.

2. An Overview of JPEG2000

After an optional color transform of the first three components, JPEG2000 employs a wavelet transform (either irreversible or reversible in Part-1) on the tiles of a component. The components can be partitioned into arbitrarily sized tiles. The wavelet transform is iteratively applied to the LL subband up to a certain level (usually in the range of 5-9). The coefficients of the DWT (discrete wavelet transform) are then grouped into code-blocks (usually 64x64 blocks, but they can be set to an arbitrary value). These code-blocks are independently arithmetically encoded using Taubman's EBCOT scheme. While the wavelet transform naturally imposes resolution scalability, the EBCOT scheme makes quality (distortion) scalability possible. A detailed description of the JPEG2000 standard can be found in [9].

2.1. The JPEG2000 Code-Stream Syntax

The main building block of the JPEG2000 code-stream is a packet. A packet contains contributions from code-blocks from a component, a tile, a quality, a resolution and a certain precinct. A precinct is yet another partition scheme, which groups code-blocks belonging to the same spatial region. A packet consists of the tag tree encoded header and the packet body, which contains the compressed coefficient data. Several code-blocks may contribute to a certain packet, the lengths of the code-block contributions to a packet (CCP) are stored in the packet header. The packet borders generally have to be found by a rather complex decoding procedure of the packet headers, which mainly consist of tag tree encoded data. However, there are coding parameters – introduced for easier random access and error resilience – that indicate packet borders through `SOP` (start of packet header) and `EPH` (end of packet header) marker sequences. These greatly reduce the effort to find packet borders, which in fact greatly reduces the complexity of the encryption of packet bodies. The code-stream syntax requires that there is no two byte sequence in excess of `0xff8f` contained in the packet body nor that it ends with `0xff`. The same rules apply to CCPs. However, the restriction that a CCP must not end with `0xff` is solely applied to avoid markers at CCP borders and is therefore of minor importance for code-stream-compliance.

2.2. The Number of Codewords

In order to assess the feasibility and computational cost of the iterative encryption approach it is necessary to determine the number of possible codewords $\mathtt{CW}_n$ for a n-byte CCP, which is the same for a packet body disregarding the requirement that CCPs must not end with `0xff`. A recursive definition[1] can be used:

Lemma 2.1 (Recursive definition of $\mathtt{CW}_n$)

$$\begin{aligned}\mathtt{CW}_1 &= 255\\ \mathtt{CW}_2 &= 255 * 255 + 144\\ \mathtt{CW}_n &= 255 * \mathtt{CW}_{n-1} + 144 * \mathtt{CW}_{n-2}\end{aligned}$$

The great advantage of this definition is that it is a linear recurrence relation which can be easily solved. The solution is a polynom of degree n which can rapidly be calculated.

Proof sketch 1 (Lemma 2.1)
The number of codewords with length 1 is 255.
*The number of codewords with length 2 is put together by those that start with a `0xff` (144) and those that do not (255*255).*
Generally all codewords of length n can be constructed by adding a byte with 255 possibilities to the codewords with length $n-1$ and by adding a `0xff` and a byte with 144 possibilities to the codewords with length $n-2$.

3. Iterative Encryption

The iterative encryption which works on CCPs was proposed by Wu and Deng in [14]. The CCPs are recursively encrypted until they are code-stream-compliant. The basic encryption algorithm is the following:
For all CCPs:

1. Encrypt the CCP.
2. Check if it contains a two byte sequence in excess of `0xff8f`.
   If yes goto 1 and re-encrypt the encrypted CCP.
3. Check if it ends with `0xff`.
   If yes goto 1 and re-encrypt the encrypted CCP.
4. Output the code-stream-compliant encrypted CCP.

Accordingly, for decryption the cipher text is iteratively decrypted until it is code-stream-compliant. Apparently, this approach is fully reversible and encrypts 100% of the packet body data.

Hence the packet body data is as secure as the underlying encryption algorithm's strength, in general the usage of

[1] Thanks to cbrown@brownsystems.com and CHHeckmann@gmail.com for their posts in math.sci

state-of-the-art block ciphers (e.g. AES) is suitable. The information contained in the meta-data (headers) is accessible. The image content which is contained in the packet bodies is protected.

Theoretically this approach can easily be extended to packet bodies, by simply iteratively encrypting the packet bodies. However, we have not yet discussed the computational cost of this approach, which will in general prevent the application of this approach on a packet body basis.

3.1. Computational Cost

If we regard encryption as a randomization process, the probability of randomly obtaining a code-stream-compliant codeword of length n is the number of CPP/packet body codewords of length n divided by all possible bytes. In [14] the number of codewords is estimated by 255^n, but applying the exact formulas (see definition 2.1) leads to an exact estimation. Hence the probability of obtaining a code-stream-compliant codeword of length n is

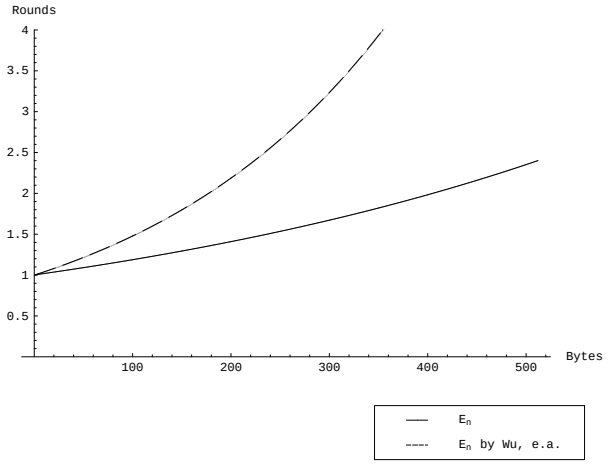

Figure 1. The expected number of rounds for up to 512 bytes

$$p_n = \frac{\mathsf{CW}_n}{256^n}. \tag{1}$$

Set q_n to $1 - p_n$. Because the encryption rounds are independent, the expected number of encryption rounds for a code-stream-compliant bitstream is (cf. [14]):

$$E_n = p_n + 2q_n p_n + 3q_n^2 p_n + \ldots \tag{2}$$

$$= \sum_{k=0}^{+\infty} (k+1) q_n^k p_n = \frac{1}{p_n}. \tag{3}$$

The number of coefficients contributing to a code-block is limited to 4096. Hence it is assumed in [14] that the CCP length is below 512. The usage of sufficient quality layers is necessary to reduce the CCP length and to provide a basis for this assumption. In section 5 the actual distributions of CCP lengths are evaluated for three different compression parameter settings. Compared to the estimation of Wu and Deng in [14] where the estimate of the expected number of rounds for a codeword with 512 bytes is 7.42, the actual expected number of rounds is 2.40. This substantial reduction of complexity raises the question if this approach can be applied to packet bodies, which could be parsed very easily using JPEG2000's marker sequences. As there are no explicit restrictions for the packet length, we analyzed the probability and the number of expected rounds for up to 5120 bytes. The outcome is shown in figures 2. Actually, if a maximum packet body length can not be guaranteed by appropriate compression parameters, the encryption of packet bodies with the iterative encryption scheme is not feasible, as the number of expected rounds increases exponentially. But if we can assure a maximum packet size, this approach remains feasible, e.g., for 1623 the number of expected rounds is strictly below 16.

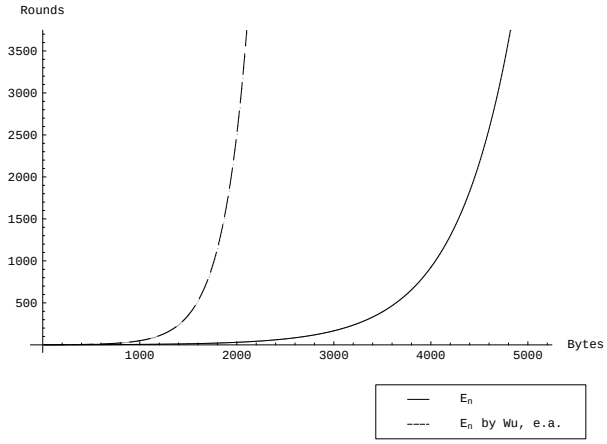

Figure 2. The expected number of rounds for up to 5120 bytes

A maximum packet body length can be achieved by reducing the precinct size to the size of a code-block. Thus the number of code-blocks contributing to a packet is three at most (one for every subband except the LL subband). Furthermore the size of the code-blocks can be reduced and the number of quality layers increased. The drawback of this solution is that the compression performance is reduced, because for every packet at least 1 byte for the packet header and 4 bytes for the two marker sequences SOP and EPH are needed.

4. Statistical Tests for the Stochastic Model

In this section we present experiments that give further evidence that the stochastic model is correct. In order to verify the stochastic model presented in section 3.1, it is necessary to conduct statistical tests. The encryption process can be considered a random process, actually an ideal cipher has this property. Hence encrypting a CCP or a packet body is equivalent to generating a random byte sequence of the same length. The probability of generating a compliant code-stream is theoretically obtained through our stochastic model. The relative frequency of the randomly created compliant code-streams should converge to the predicted probability.

4.1. Test Setup

The statistical tests were conducted in Mathematica. For every length m of a CCP or a packet the probability of format-compliant encryption is given by p_m (see section 3.1 equation 1). A random byte sequence of length m is generated and its code-stream compliance determined. Hence for every m the relative frequency of the code-stream-compliant outcomes should converge to p_m. Anyhow, the term "should converge" has to be expressed more precisely. Basically, we have a random experiment in which an event (code-stream compliance) occurs with a certain probability. We have to determine the number of iterations of the random experiment that is necessary to assess the actual probability of the event from the relative frequency of its occurrence with a certain level of confidence.

4.2. Confidence Intervals with Normal Distribution

Let p be the probability of an event (code-stream compliance). One experiment consists of n trials and results in k outcomes that are code-stream-compliant. For iterated experiments the number of outcomes K is a random variable, which follows a binomial distribution. The random variable $H = \frac{K}{n}$ therefore follows approximately a normal distribution $N(\mu, \frac{\sigma^2}{n})$, with $\mu = p$ and $\sigma^2 = p(1-p)$. Hence for large enough n the confidence interval for p is given by $p \pm \kappa_{1-\frac{\alpha}{2}} \frac{\sigma}{\sqrt{n}}$, where κ_β is the β quantile of the normal distribution. Since $\sigma^2 = p(1-p)$, the maximum value for σ is $\frac{1}{2}$. Hence the following holds for the error e:

$$e = \kappa_{1-\frac{\alpha}{2}} \frac{\sigma}{\sqrt{n}} \leq \kappa_{1-\frac{\alpha}{2}} \frac{0.5}{\sqrt{n}} \tag{4}$$

If we want the error to be small, we have to choose a big enough n. The error e is smaller than 3.10% for 1000 iterations with a probality of error of 5%. If our stochastic model is correct, at least 95% of the results are within ± 3.1 of our prediction. Additionally a hypothesis test has been conducted, which is based on a non-approximated binomial distribution.

4.3. Hypothesis Test

For iterated experiments with n trials the number of outcomes K follows a binomial distribution with parameters n and p. Let α be the desired probability of error. We test the hypothesis that the probability p is equal to p_0 against the alternative hypothesis that p is greater than p_0 with a probability of error of $\frac{\alpha}{2}$. Therefore a critical region for the number of outcomes is defined by:

$$\sum_{i=\lambda_r}^{n} \binom{n}{i} p_0^i (1-p_0)^{n-i} \leq \frac{\alpha}{2} \tag{5}$$

Additionally we test the hypothesis that the probability p is equal to p_0 against the alternative hypothesis that p is smaller than p_0 with a probability of error of $\frac{\alpha}{2}$. Hereby the critical region for the number of outcomes is defined by:

$$\sum_{i=0}^{\gamma_r} \binom{n}{i} p_0^i (1-p_0)^{n-i} \leq \frac{\alpha}{2} \tag{6}$$

If the actual number of outcomes k is in excess of λ_r or below γ_r, the hypothesis $p = p_0$ is rejected with a probability of error α. Hence we can construct upper and lower bounds for k.

4.4. Results

In this section the results for codewords up to 5120 bytes are presented. For codewords up to a length of 5120 bytes an experiment with 1000 iterations has been conducted for every 10th length. In figure 3 the predicted probability and the upper and lower bounds obtained with normal distribution (nd) and fratio distribution (fd) are shown. These results strongly suggest that the stochastic model is correct.

In figure 4 our results are compared to those of Wu and Deng, their estimation of the number of rounds and their results. Interestingly, their first and their last empirical results are close close to our calculation of the expected number of rounds, while the remaining results are closer to their approximation of the expected number of rounds. The probability of a format-compliant encryption and therefore the expected computational effort solely depends on the codeword length. Hence the next section is dedicated to empirical results of the lengths and distributions of CCPs and JPEG2000 packets.

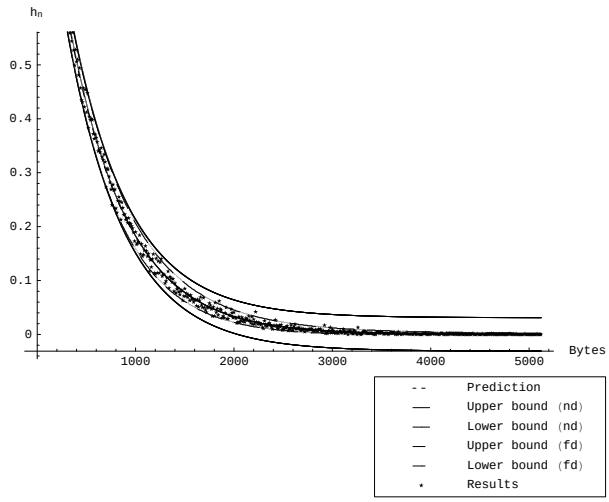

Figure 3. Experimental results for up to 5120 bytes

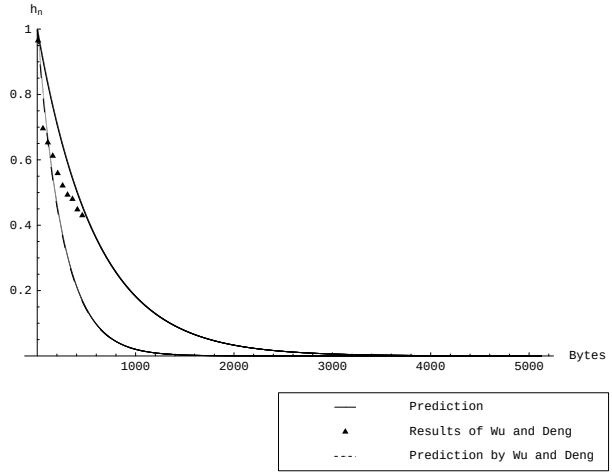

Figure 4. Experimental results for up to 5120 bytes

5. CCP and Packet Lengths and Distributions

The CCP and JPEG2000 packet lengths depend on the compression parameters and the source image. The CCP and packet lengths have been evaluated for a set of 1035 images consisting of standard test images (lena, baboon, ...) and video testing sequences like foreman and akiyo. All compression parameters, if not stated otherwise, are set to JJ2000's default values. In the tables nqls indicates compression without quality layers, jj2k the JJ2000 default number of quality layers (32) and pr32 indicates a code-block and precinct size of 32x32 and 32 quality layers, which is assumed to lead to small packets and CCPs. The lower the CCP or packet length, the lower the expected computational effort for encryption. If packet based encryption is feasible, the computationally expensive parsing of CCP borders is omitted and instead a simple marker parsing procedure can be applied.

5.1. CCP Lengths and Distributions

Table 1 shows the number of CCPs, mean and maximum value and the standard deviation of the CCP lengths for the test set.

setting	avg. ccps	mean	max	stdev
nqls	45.25	638.31	4808	647.10
jj2k	346.88	83.29	2135	104.04
pr32	551.25	54.46	1292	47.94

Table 1. The CCP lengths for the test set

For no quality layers, in fact, 8.94% of the CCP lengths are in excess of 1623. However, these 8.94% of CCP lengths contribute 29.75% to the overall packet body bytes. Hence quality layers have to be applied to reduce the computational complexity of this approach to a sensible range. For JJ2000 default settings only 0.0014% of the CCP lengths exceed 1623, which contribute only 0.1039% to the overall packet body bytes. No CCP length is in excess of 1623 for pr32 setting.

As the results have shown, the iterative encryption approach is feasible for CCPs if the right compression parameters (enough quality layers) are chosen. Similar considerations can be made for the application of this approach to packet bodies.

5.2. Packet Lengths and Distributions

Table 2 summarizes the results for the packet lengths of JPEG2000 compressed images of the test set. Compression without quality layers produces extremely long packet bodies and is therefore not well suited for the iterative encryption approach. JJ2000's default settings generate an average packet length which can be encrypted with the iterative encryption approach, however, there are extreme outliers. The average packet length for 32x32 precincts is very low even though there are outliers. With no quality layers, 20.35% of the packet lengths are in excess of 1623. The packets in excess of 1623 take up 95.47% of the overall packet body bytes. Thus the iterative encryption approach can not be efficiently applied to the packet bodies of JPEG2000 compressed images with no quality layers.

With quality layers this percentage improves to only 2.236%, but these packet bodies contribute about 59.47% to

setting	avg. packets	mean	max	stdev
nqls	12.00	2407.50	2445405	26665.62
jj2k	126.22	228.89	684175	3322.96
pr32	3675.22	8.20	101828	176.75

Table 2. The packet lengths for the test set

the overall packet body bytes. Hence the iterative encryption approach on a packet body basis can not be applied to images compressed with JJ2000's default settings.

For a precinct size 32x32 only 0.00176% of the packet lengths exceed 1623. These 0.00176% contribute 7.598% to the overall packet body bytes. These results are the consequence of the extremely high ratio of empty packets (over 90%), which reduce the average packet body length and the coding performance, especially when SOP and EPH markers are inserted into code-stream.

The pr32 compression parameters seem to make the iterative encryption approach on a JPEG2000 packet body basis possible on average (mean is 8.2 and stdev is 176.8), but the possibility of extreme outliers is given, as the results are strongly influenced by the enormous amount of empty packets.

6. Conclusion

In this paper we have analyzed a format-compliant encryption approach for JPEG2000. Through the exact formula for the number of codewords for CCPs and packet bodies disregarding CCPs we have given an exact analysis of the computational cost of this approach. Although the computational complexity of the approach has been found to be far less than estimated in [14] we conclude on the basis of our experimental results that the usage of quality layers is inevitable. Hence the iterative encryption approach of Wu and Deng is not a general encryption approach for JPEG2000 compressed images, but strongly linked to according encoding parameters. There are compression parameters that seem to make the usage of the iterative encryption approach for packet bodies possible, the simpler parsing procedure through marker sequences is made at the expense of a severely reduced compression performance and the risk of extreme outliers, which greatly increase the computational complexity.

Therefore, future work will include the analysis and consideration of other format-compliant encryption approaches for JPEG2000.

References

[1] B. Bhargava, C. Shi, and Y. Wang. MPEG video encryption algorithms. *Multimedia Tools and Applications*, 24(1):57–79, 2004.

[2] J. A. F. Dufaux, S. Wee and T. Ebrahimi. JPSEC for Secure Imaging in JPEG 2000. In *SPIE Proc. Applications of Digital Image Processing XXVII*. SPIE, Aug. 2004.

[3] H. Kiya, D. Imaizumi, and O. Watanabe. Partial-scrambling of image encoded using JPEG2000 without generating marker codes. In *Proceedings of the IEEE International Conference on Image Processing (ICIP'03)*, volume III, pages 205–208, Barcelona, Spain, Sept. 2003.

[4] Y. Mao and M. Wu. Security evaluation for communication-friendly encryption of multimedia. In *Proceedings of the IEEE International Conference on Image Processing (ICIP'04)*, Singapore, Oct. 2004. IEEE Signal Processing Society.

[5] R. Norcen and A. Uhl. Selective encryption of the JPEG2000 bitstream. In A. Lioy and D. Mazzocchi, editors, *Communications and Multimedia Security. Proceedings of the IFIP TC6/TC11 Sixth Joint Working Conference on Communications and Multimedia Security, CMS '03*, volume 2828 of *Lecture Notes on Computer Science*, pages 194 – 204, Turin, Italy, Oct. 2003. Springer-Verlag.

[6] D. Z. Shiguo Lian, Jinsheng Sun and Z. Wang. A selective image encryption scheme based on JPEG2000 codec. In Y. N. K. Aizawa and S. Satoh, editors, *aSmart Card Research and Applications*, volume 3332 of *Lecture Notes in Computer Science*, pages 65–72. Springer Verlag, 2004.

[7] S. Shin, K. Sim, and K. Rhee. A secrecy scheme for MPEG video data using the joint of compression and encryption. In *Proceedings of the 1999 Information Security Workshop (ISW'99)*, volume 1729 of *Lecture Notes on Computer Science*, pages 191–201, Kuala Lumpur, Nov. 1999. Springer-Verlag.

[8] T. Stütz and A. Uhl. Image confi dentiality using Progressive JPEG. In *Proceedings of the International Conference on Information, Communications & Signal Processing, ICICS '05*, Bangkok, Thailand, Dec. 2005.

[9] D. Taubman and M. Marcellin. *JPEG2000 — Image Compression Fundamentals, Standards and Practice*. Kluwer Academic Publishers, 2002.

[10] A. Uhl and A. Pommer. *Image and Video Encryption. From Digital Rights Management to Secured Personal Communication*, volume 15 of *Advances in Information Security*. Springer-Verlag, 2005.

[11] Y. S. V. Conan and S. Thomann. Symmetric block cipher based protection: Contribution to JPSEC. ISO/IEC JTC 1/SC 29/WG 1 N 2771, Oct. 2003.

[12] S. Wee and J. Apostolopoulos. Secure transcoding with JPSEC confi dentiality and authentication. In *Proceedings of the IEEE International Conference on Image Processing (ICIP'04)*, Singapore, Singapore, Oct. 2004.

[13] H. Wu and D. Ma. Effi ecient and secure encryption schemes for JPEG2000. In *Proceedings of the 2004 International Conference on Acoustics, Speech and Signal Processing (ICASSP 2004)*, pages 869–872, May 2004.

[14] Y. Wu and R. H. Deng. Compliant encryption of JPEG2000 codestreams. In *Proceedings of the IEEE International Conference on Image Processing (ICIP'04)*, Singapure, Oct. 2004. IEEE Signal Processing Society.

[15] R. D. Y. Wu and D. Ma. ImAccess: A method for JPEG2000 access control. ISO/IEC JTC 1/SC 29/WG 1 meeting, Seoul, Mar. 2003.