Fast and Secure Real-Time Video Encryption

C. Narsimha Raju, Ganugula Umadevi, Kannan Srinathan and C. V. Jawahar
International Institute of Information Technology, Hyderabad, India-500032.
{narsimha_raju@research., umadevi@research., srinathan@, jawahar@}iiit.ac.in

Abstract

Advances in digital content transmission have increased in the past few years. Security and privacy issues of the transmitted data have become an important concern in multimedia technology. In this paper, we propose a computationally efficient and secure video encryption algorithm. This makes secure video encryption feasible for real-time applications without any extra dedicated hardware. We achieve computational efficiency by exploiting the frequently occurring patterns in the DCT coefficients of the video data. Computational complexity of the encryption is made proportional to the influence of the DCT coefficients on the visual content. On an average, our algorithm takes only $8.32ms$ *of encryption time per frame.*

1. Introduction

Advances in multimedia technologies have popularized applications like video conferencing, pay-per-view, Video-On Demand (VOD), video broadcast, etc. In such applications, confidentiality of the video data during transmission is extremely important. This necessitates secure video encryption algorithms.

In the *naïve* approach for video encryption, the MPEG stream (bit sequence) is treated as text data, and encrypted using standard encryption algorithms like DES (Data Encryption Standard) [7], RC5 (Rivest Cipher), AES (Advanced Encryption Standard), etc. Though this approach is supposedly the most secure for video encryption, it is computationally infeasible for real-time applications.

Arguing that the full content of the video is not critical, *selective encryption* algorithms were proposed [1, 7, 8]. These methods encrypt a selected portion of the video data (for example headers of the video streams, I frames and I-blocks in P and B frames, I frames and motion vectors in P and B frames, etc.) using text-based encryption algorithms. This decreases encryption time. For real-time applications, *light-weight encryption* algorithms were also proposed [6, 5, 15]. These methods encrypt using simple XOR or encrypt selected bits of the video data (for example, sign bits of I frames, motion vectors, etc.). These encryption algorithms are much faster than selective algorithms. Also, they add less overhead on the codec. (Note that if encryption modifies the syntax of the MPEG bit stream, it adds overhead to the MPEG codecs.) Another category of algorithms is based on *scramble (permutation)* only methods, where the DCT coefficients are permuted to provide confusion. However, in most of these methods, computational efficiency comes at the cost of security.

Maples and Spanos [7] proposed a *selective* encryption algorithm called AEGIS. Using the DES algorithm, the AEGIS algorithm encrypts only the I frames of the MPEG video stream. However, Agi and Gong [1] showed that partial information leakage from the I-blocks in P and B frames renders AEGIS unsuitable for applications like military where each and every part of the video data is important. Qiao and Nahrstedt [8] proposed another selective Video Encryption Algorithm (VEA). In this algorithm, a chunk of I frame is divided into two halves. Both the halves are XORed and stored in one half. The other half is then encrypted using DES. The VEA provides good security and reduces the number of XOR operations significantly compared to AEGIS. These selective encryption algorithms are secure, but they are not practical for real-time implementations because they require high computational time.

Choon [6] proposed a *light-weight* and cost effective encryption algorithm based on the Shannon principle of diffusion and confusion. These principles can be achieved by permutation of macroblocks followed by XOR operation on the permuted macroblock. Choo [5] proposed another light-weight encryption algorithm on the uncompressed raw MPEG data named Secure Real-time Media Transmission (SRMT), which uses two block transpositions and a XOR operation. Tang [11] proposed a *scramble* based encryption algorithm using *permutation* of the DCT coefficients. The basic idea is to use a random permutation list to replace the zig-zag order of the DCT coefficients of a block to a 1×64 vector. Zeng and Lie [15] extended Tang's permutation range from a block to a segment, where each segment consists of several macroblocks. Within each seg-

978-0-7695-3476-3/08 $25.00 © 2008 IEEE
DOI 10.1109/ICVGIP.2008.100

ment, the DCT coefficients of the same frequency band are randomly shuffled within the same band. Apart from shuffling of the I frames, they also permute the motion vectors of P and B frames. However, light-weight encryption and scramble-only methods provide less security than the naive encryption. The drawback of these algorithms is that they trade-off security for speed.

Based on the state of the art in video encryption, we observe that:

- For complete and provable security of the video data, the entire video needs to be encrypted. However, a *naïve* encryption of the complete video stream is computationally infeasible.
- The encryption algorithm should not be susceptible to attacks like known-plaintext attack and ciphertext-only attack. Computational efficiency should not come at the cost of security.

In this paper, we address the apparent contradiction between computational efficiency and security. We demonstrate that they can co-exist. We argue that encryption of a video is more than just encrypting the associated pixels or DCT coefficients. Our algorithm (i) learns from the statistical properties (distributions) of the DC and AC coefficients from a large number of videos for deriving computational efficiency, (ii) realizes that the roles of DC and AC coefficients in a MPEG stream are different and that the level of security required for a specific coefficient is proportional to the (visual) influence in the video and (iii) smartly interleaves the Electronic Code Book (ECB) and Cipher Block Chaining (CBC) modes of encryption for adapting the text-based encryption algorithms to the DC and AC coefficients of the video data. The proposed algorithm on an average takes very less time to encrypt a video frame, which is significantly smaller compared to the state of the art secure video encryption algorithms as shown in Section 5.

2. Preliminaries

MPEG Background: A MPEG (Moving Picture Expert Group) [13] video is composed of a sequence of Group of Pictures (GOPs). Each GOP consists of three types of frames namely I, P and B. I frames are called intra-coded frames and are compressed without reference to any other frames. They are split into non-overlapping blocks (intra-coded) of 8×8 pixels which are then compressed using Discrete Cosine Transform (DCT), quantization (Q), zig-zag scan, followed by run-length coding and entropy coding (VLC). Figure 1 shows the block diagram of the MPEG video coding scheme.

The P and B frames are forward predictive and bi-directional predictive coded frames, respectively. These are

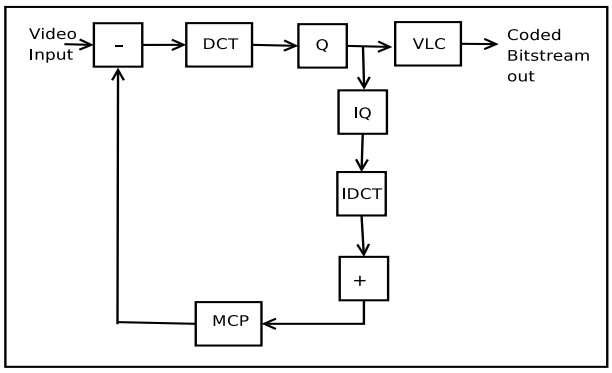


Figure 1. MPEG Video Coding Scheme

subjected to compensation by subtracting a motion compensation prediction. The residual prediction error signal frames are split into non-overlapping blocks (inter-coded) of 8×8 pixels which are compressed in the same way as the blocks of intra-frames. Sometimes, P and B frames also have some intra-coded blocks when better efficiency will be obtained using intra-coded compression [4]. These intra-coded blocks are called I-blocks in P and B frames.

Encryption/Decryption: The process of converting plaintext to ciphertext is called *enciphering or encryption*; restoring plaintext from ciphertext is *deciphering or decryption*. Both the encryption and decryption algorithms take a *key* (K) and plaintext/ciphertext as input. In the case of images, plaintext is a set of pixel values arranged in an orderly manner. Encrypting images/videos constitutes reordering these pixel values so that they convey no visual information about the original image. An image/video can also be encrypted in the compressed domain. In this case, the DCT coefficients are encrypted in such a way that the content is made illegible for the unauthorized. Only an authorized user can get back the original content using the decryption algorithm.

In cryptography, a **block cipher** is a symmetric key cipher which operates on fixed-length groups of bits, termed blocks, with an unvarying transformation. When encrypting, a block cipher might take an n-bit block of plaintext as input and output a corresponding n-bit block of ciphertext. The exact transformation is controlled using a second input – the secret key. Decryption is similar, takes an n-bit block of ciphertext together with the secret key and outputs the original n-bit block of plaintext. Examples of block-ciphers are RC5, AES, DES, Blowfish, etc. We use block ciphers in this work.

A block cipher operates in different modes. The predominant modes are **Electronic Code Book (ECB)** and **Cipher-Block Chaining (CBC)**. In ECB mode, the plaintext is divided into blocks and each block is encrypted separately. In

the CBC mode, each block of plaintext is XORed with the previous ciphertext block before being encrypted. This way, each ciphertext block is dependent on all plaintext blocks processed up to that point. In this mode, changes in the plaintext propagate forever in the ciphertext and encryption cannot be parallelized. Also, decryption cannot be parallelized.

3. Basis for Our Algorithm

Classical encryption schemes are designed for encryption of textual (or numeric) data. In general, video data is huge (a frame can have $40,000$ bits and their would be 25-30 frames per second). The information value of video data is far less than that of an equal amount of text data. Even then, for better security, video data is encrypted by using classical algorithms (DES, AES, RC5 etc.). This causes delay in processing and is not suitable for real-time applications. In this paper, we propose an equivalent (in terms of security requirements) algorithm that is suitable for real-time applications.

MPEG video data is often compressed using DCT. Each video frame is divided into sub-images and then DCT is applied on each 8×8 block. After coding, the 64 transformed coefficients are zig-zag ordered such that the coefficients are arranged approximately in the order of *increasing* frequency. These DCT transform coefficients can be classified into two groups, DC and AC. The DC coefficient is the mean value of a block. All other coefficients describe the variation around this DC value and these are referred to as AC coefficients. However, most of the energy is contained in the DC and a few AC coefficients. In order to apply classical encryption algorithms, we perform statistical analysis to find out the variation in DCT coefficients in a video block.

To study the behavior of DC and AC coefficients, we consider a set of fifty *random* MPEG videos taken from different sources. The selected videos consist of both benchmarked sources and fresh sources comprising of movie-clips, home videos, videos collected from the Web, as well as popular test videos like Foreman, Table Tennis, Mother, etc. The set has been chosen such that the videos exhibit a variety of motion characteristics with objects varying in size, shape, speed and color.

Statistical analysis is done on the DCT coefficients after the quantization operation and before entropy coding (refer Figure 1). We studied the distribution of the *DC* and *AC byte* values and calculated the percentage occurrences of the monogram *byte* values for various ranges. We found some really interesting behavior in the frequency distribution of both *DC* and *AC* coefficients.

For various video streams, Figure 2 shows the stack bar occupied by different range of DC coefficients. It can be

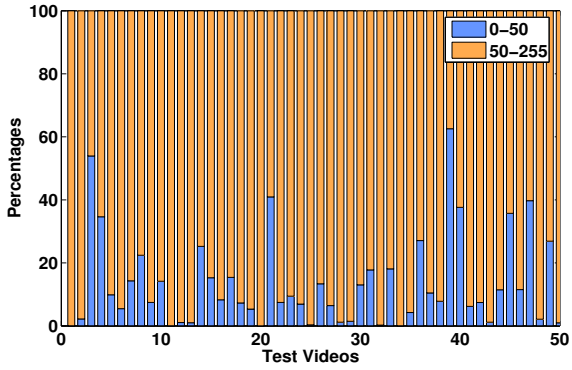


Figure 2. Percentage Variation Plot of DC Coefficients

observed from the figure that each stack bar is predominant with orange color, this pertains to the DC range of 50-255. Similarly dark blue color pertains to DC range of 0-50. From this it can be inferred that a high percentage (86.53%) of the DC coefficients values lie in the range of 50-255. Note that there could be isolated blocks and frames with high frequency DC values, but the average over a video clip is in the range $50-255$ most of the time. Similar analysis is conducted on the AC coefficients. Since the variation of the AC is quite different from the DC, we changed the ranges accordingly. We observed that around 94.04% of the ACs lie in the range of 0 to 20. This is due to the fact that lower frequency AC coefficients carry less energy [14].

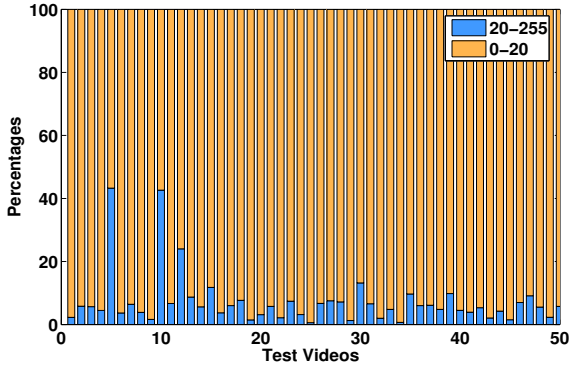


Figure 3. Percentage Variation Plot of Quadruple AC Coefficients

We further extended our analysis and considered four consecutive AC coefficients at a time (quadruples) and performed similar analysis (since AES, DES and RC5 are block ciphers). We observed the distribution of ACs in all the blocks of the video sequence. All combinations of the

$256 \times 256 \times 256 \times 256$ quadruples are theoretically valid. So, we checked the distribution of the quadruples for all the combinations of the byte values (0-255). It is found that most of the time (92.89%), all the AC values in the quadruples lie in the range of $0 - 20$. Analogous to the description given for Figure 2, Figure 3 shows the percentage occupied by the quadruples, whose AC coefficient ranges is in 0-20 (orange) and 20-255 (dark-blue). This analysis is also done for all the fifty video sequences. Motivated from this statistical analysis, we design an encryption algorithm for the video data based on quadruples behavior and the range of values these DCT coefficients take. We exploit the presence of such frequently occurring patterns to obtain the computational efficiency. Statistical analysis is done by discarding the sign of the DCT coefficients.

4. Our Algorithm

This section of the paper describes the proposed encryption scheme. Though we can use any of the algorithms like DES (Feistel Structure), RC5 for encryption, we used RC5 [9] to encrypt the DCT coefficients. RC5 has following suitable characteristics: it is a block-cipher with varying block size (32, 64 or 128 bits), varying key size (0 to 2040 bits) and variable number of rounds (0 to 255) (so that the user can choose the level of security appropriate for his application). It is a fast block cipher with a simple and easy-to-analyze structure. It also has adaptable word size in order to suit processors of different word lengths and flexibility of changing the parameters easily [2]. Figure 4 shows the block diagram of the encryption process.

Our approach uses $RC5$ [9] algorithm with key size of 128 bits and 20 rounds of operation. The *pre-processing* step of our algorithm is shown in Algorithm 1. The look-up table generated is used in the encryption and decryption stages of the algorithm. It is never modified in the entire process of encryption/decryption.

Algorithm 1 Pre-processing Step — Look-up Table Construction

Step 1: Generate all the combinations of quadruples from -20 to 20.
Step 2: Encrypt each quadruple using the ECB mode of RC5 encryption.
Step 3: Use the list of coefficients and their encrypted values as a look-up table for encryption of the AC coefficients.

The pre-processing step could be done before the encryption phase and in fact even before the input (video to be encrypted) is known. For frequently occurring AC quadruple, instead of undergoing the entire process of encryption, a mere look-up serves the purpose. Hence there is a drastic reduction in the encryption time despite the usage of a text-based algorithms. Note that this encryption does not compromise the security of the algorithm, since this (look-up) is used only with the relatively less significant AC coefficients.

Since the DC component carries most of the energy, it is the most significant value of the block. The encryption of the DC value plays a key role in the visibility of the video block. Some encryption algorithms only encrypt the DC coefficient while simply permutating the ACs [14]. In order to enhance the security, we apply CBC mode for encrypting the DC coefficients rather than the general ECB mode which is applied for the AC coefficients. The proposed scheme of encryption gains its advantages from not only the look-up table but also the dynamism of changing the mode of encryption based on the type of coefficient. Algorithm 2 explains the main encryption.

Algorithm 2 Main Encryption Algorithm

Step 1:
for each and every block of a frame **do**
  Step 1.1: Consider four consecutive ACs (AC_t, AC_{t+1},AC_{t+2}, AC_{t+3}).
  Step 1.2: Compare these coefficients against the look-up table for a *hit* or *miss*
  if *hit* **then**
    Replace the 4 AC coefficients with their encrypted values.
  else
    Apply the ECB mode of RC5 encryption with these ACs as input.
  end if
  Step 1.3: When DC coefficients are considered, collect four DC coefficient values in a block and encrypt them using the CBC mode of RC5 algorithm.
end for

At the receiver end, the authorized recipient first generates the look-up table for all the combinations of -20 to 20 quadruples and encrypts the quadruples in the way similar to the pre-processing step of the algorithm using the *key*. In the generated look-up table, the columns are exchanged and sorted in order to look up for the plain values when the cipher values are known. For obtaining the DC coefficients, the recipient can directly apply general RC5 decryption with the CBC mode.

Sometimes when there is less motion in objects and scenes, the encryption of the I frames in the videos will render the subsequent P/B frames difficult to perceive due to the dependency of the P/B frames on the I frames. When video content is not highly correlated in the temporal dimension, the I-blocks of the P/B frames will help in partial leakage of the image information [1]. In order to achieve

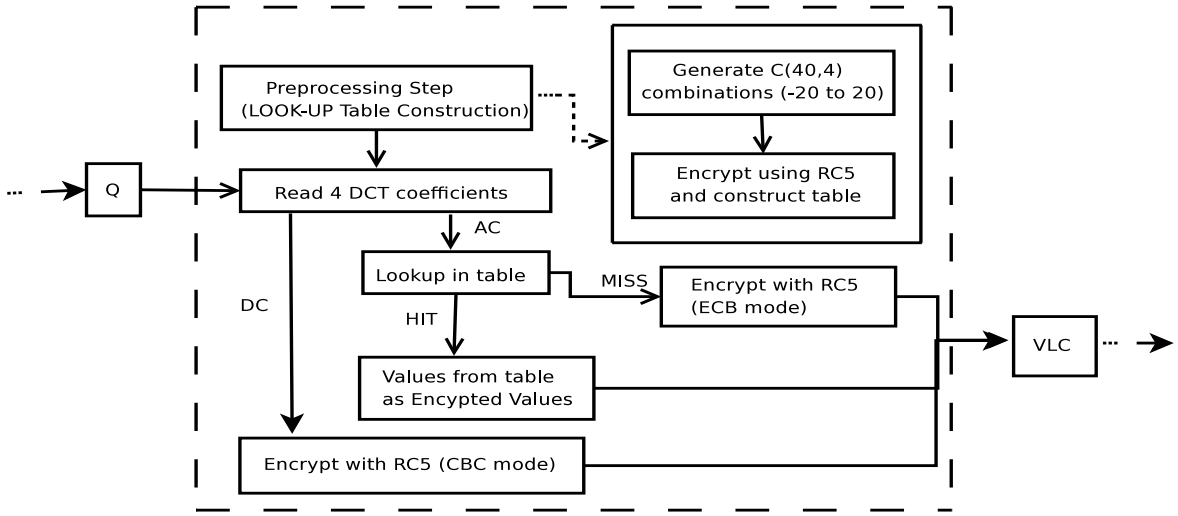


Figure 4. Block Diagram of Encryption Algorithm

better security, the DCT coefficients of the I-blocks also need to be encrypted. Identifying the I-blocks in P/B frames introduces overhead because one has to go through the entire MPEG bit stream. This time consuming process of identifying the I-blocks makes the naive approach a better choice in many cases [8]. For I-blocks in P and B frames, there is a substitute scheme that encrypts the motion vectors of the video sequence.

We concentrate only on the encryption of I frames because for real-time applications like pay-per-view there is no need to encrypt the motion vectors. Partial leakage of image information from the I-blocks in P/B frames might persuade a non-paying consumer to buy the video. In order to use our algorithm for applications where the entire video content is important, motion vectors of P and B frames are also encrypted along with I frames. The same look-up table (used for encryption of AC coefficients) can be used for encryption of these motion vectors.

4.1. Key Distribution

This part of the paper covers the RC5 key distribution to the end-user. To withstand brute-force attacks, the RC5 128 bit key is changed periodically and transmitted to the receiver in an encrypted form. The key can be encrypted using the standard public-key cryptosystems such as RSA. This means that two keys are used for the encryption and decryption process. One key is public (asymmetric) and the other key is private (symmetric). This setting has an additional advantage that the cryptanalyst needs to apply two different attacks since symmetric and asymmetric cryptosystems have to be tackled separately. This enhances the security level considerably [10].

In our method of encryption, the transmitter uses the RSA public key of the intended recipient to encrypt the RC5 key. At the receiver end, the recipient uses his private RSA key to decrypt this RC5 key. The decrypted RC5 key is then used to decrypt the rest of the bit stream by generating the decryption look-up table.

The RSA standard is not directly used to encrypt or decrypt the MPEG bit stream because the processing time required by it is proportional to the size of the data. In this work, we use a single 128 bit key, and hence the time required to encrypt this key using the RSA algorithm is very small when compared to the encryption of MPEG frame.

5. Experimental Results

In order to evaluate the performance of the proposed encryption scheme, we implemented the algorithm on a 1 GHz dual-core processor with a RAM of 1 GB. The algorithm is coded in C using the DALI [12] library, which supports coding of MPEG-1 and MPEG-2 videos. The video data used for analysis have different motion characteristics and varying resolution with a frame rate of 30 fps. The sample test video sequences include videos like Pond, Table Tennis, Training, Professor in lab, etc. Some of the test videos along with their frame numbers are shown in Figure 5. Figure 6 shows the images of the corresponding encrypted videos. It can be observed that the details in the video block are completely lost in the encryption process.

The time spent for the pre-processing step in our algorithm is on an average 4.12 seconds per video. Since this is done infrequently (only when key is renewed) and is an off-line process, it does not affect the encryption time of the video. Apart from this, the receiver's end should have

(a) Pond(75) (b) Cat(112) (c) Professor(268) (d) SateliteView(37)

(e) Street(2) (f) Table Tennis(145) (g) Training(885) (h) Woman(511)

Figure 5. Example Frames of Test Videos (with Frame Numbers)

a small amount of storage to store the frequent patterns and the corresponding encrypted coefficients.

Apart from providing security (which is explained in the next section), an encryption algorithm for real-time multimedia data should also have two characteristics. Firstly, the encryption time should not be high, as it delays the transmission. Secondly, the compression rate of the video should not decrease, making it difficult to transmit over a network. The proposed algorithm possesses both the characteristics with much less overhead on the codec. Discussion of these characteristics follows in detail:

The encryption time taken by our method is quite low and suited for real-time applications. For a quadruple of AC coefficient in a video frame, there is around 92.89% of hit-ratio to the look-up table. For the remaining ACs we need to use RC5 explicitly. This look-up is much faster compared to the *naïve* way (where a quadruple is encrypted each time it occurs). For the DC coefficients we used RC5 with CBC mode of operation. Intuitively, using a look-up table for encryption of both DC and ACs seems to be a faster alternative. However, this approach is least secure because the ECB mode of encryption is less secure than all the other modes of encryption and this might be the reason it has never been used in the state of the art.

A comparison of encryption time of our technique with other techniques is given in Figure 7. It can be observed that the average encryption time per frame taken by our technique is around $8.32ms$, which is less than that of other techniques like AES ($30.00ms$), RC5 (11.36), SRMT [5] ($10.00ms$) and XOR followed by scrambling [6] ($7.60ms$). Though the proposed method of encryption takes an equivalent time of XOR followed by scrambling method but, the security provided by our method of encryption is far better than XOR followed by scrambling method of encryption. This makes our algorithm more suitable for real-time video encryption.

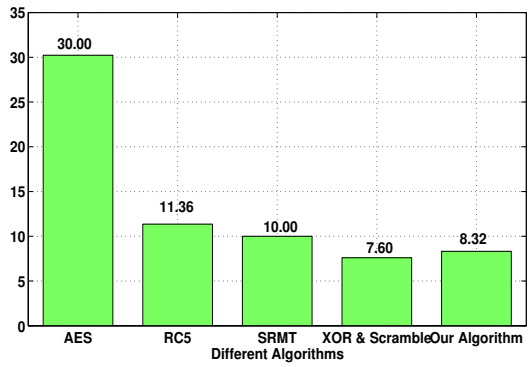
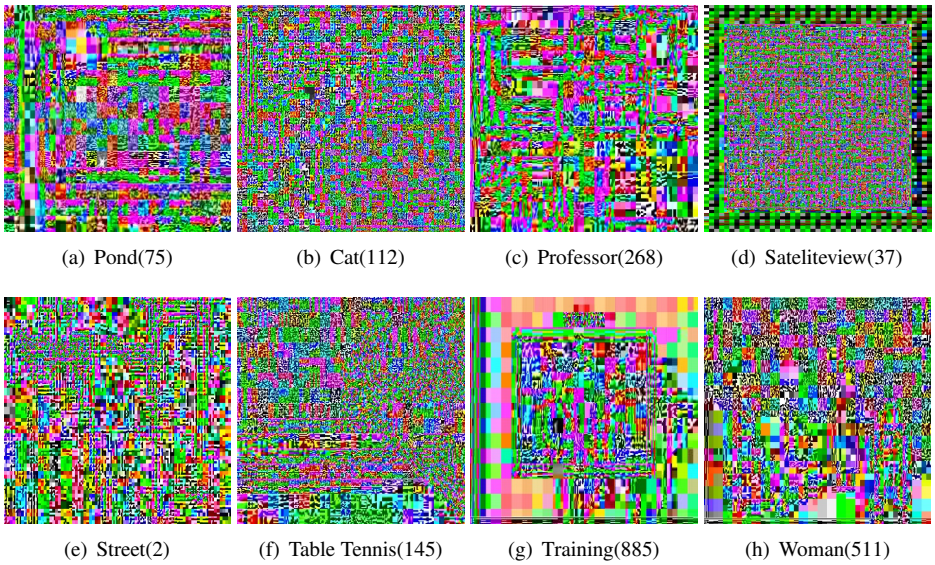


Figure 7. Encryption Time for Different Algorithms

The increase in video size by our method is same as that of the previous encryption technique [6]. When compared to scramble-only methods, though the overhead in size is high but is negotiable for better security. Table 1 tabulates the different videos files along with encrypted video size (overhead on the compression by our encryption algorithm) and number of HITS and MISS to the look-up table on the test videos.

6. Analysis

This section of the paper explains the achievements and the security analysis of the proposed encryption scheme.



(a) Pond(75) (b) Cat(112) (c) Professor(268) (d) Sateliteview(37)

(e) Street(2) (f) Table Tennis(145) (g) Training(885) (h) Woman(511)

Figure 6. Encrypted Frames of Test Videos

The algorithm satisfies the claims discussed in the introduction of this paper as follows:

- It succeeds in using text-based algorithms for videos by managing the computational overhead and hence suiting real-time applications. The security level provided is as good as security provided by RC5 or DES.
- It is selective, i.e., based on the criticality level of the DCT coefficient, the algorithm adjusts to provide optimum security (like CBC mode for DC coefficients).
- The algorithm succeeds in exploiting the statistical properties of the video for providing better encryption speed.

6.1 Security

In this subsection we discuss the security aspects of the proposed algorithm for different attacks. An encryption scheme is said to be *computationally secure* if it can withstand the following two criteria. Firstly, the cost of breaking the cipher should exceed the actual value of the encrypted information. Secondly, the time required to break the cipher should exceed the useful lifetime of the information. The proposed encryption satisfies both the criteria. The security of the algorithm is checked against *ciphertext-only*, *known-plaintext* and *chosen plaintext* attacks as these are the most important attacks.

Ciphertext-only attack: Ciphertext-only attack is a model for cryptanalysis where the attacker is assumed to have access to a set of ciphertexts and knows the encryption algorithm. This is the most difficult attack since the cryptanalyst has access only to the encrypted data. It is shown that [2], RC5 with four rounds of encryption and key size of 128 bits, can be broken if 2^{17} ciphertexts are available. Our scheme uses RC5 with 20 rounds of encryption and 128 bit key size. Hence, the security of the proposed method is certainly higher, which is not easy to break. So, the proposed algorithm is secure against ciphertext-only attack.

Known-plaintext attack: In known-plaintext attack, the attacker has access to both the ciphertext and plaintext along with the encryption algorithm. The attacker first constructs a partial table with both the plaintext and the corresponding encrypted coefficients from the videos collection. Since we used a 128 bit key for encryption he needs to try on an average 2^{127} combinations to know which combination of key being used, very difficult to break. Furthermore, the key is renewed at periodic intervals, making it more difficult for a cryptanalyst to break using known-plaintext attack. Hence the proposed method is robust against known-plaintext attack also.

Chosen-plaintext attack: A chosen-plaintext attack is an attack model for cryptanalysis which presumes that the attacker has the capability to choose arbitrary plaintexts to be encrypted and obtain the corresponding ciphertexts. According to the latest cryptanalysis [3], 12-round RC5 (with 64-bit blocks) is susceptible to a differential attack using 2^{44} chosen plaintexts. Even though the video size may look as an encouraging factor for the attacker, who wishes to get many ciphertext from a single video. The hidden strength of the algorithms is, he needs to get different combinations of ciphertext which is difficult because the analysis in Section 3 says most of ACs vary from (0 to 20). So he needs a large collection of videos encrypted with single key. This is made very difficult by renewing the key. Apart from that in [3],

Name	Size of Frames	Original Size (Kb)	Total Number of Frames	No. of I-Frames	Encrypted Video Size	HIT	MISS
Pond	352x244	5144	611	61	7400	997977	65072
Cat	704x576	19692	737	62	28048	909737	24169
Professor	352x240	10164	703	47	11040	159564	10925
Sateliteview	720x576	22252	2528	141	41472	2121612	50519
Street	640x480	9196	819	546	19865	2083735	143082
Table Tennis	352x240	1224	150	26	3196	215534	5042
Training	352x240	34168	7292	456	41148	997977	65072
Iceland	384x288	7668	1108	74	10376	180207	16182

Table 1. Encryption Overhead on Compression and Encryption Time per Frame

the author suggests that increasing the number of rounds to at least 16 will increase security against differential cryptanalysis. This was in-fact suggested by Rivest [9] that security will increase when the number of rounds of encryption is increased. Even though we used a 32 bit version of RC5, we increased the number of rounds to 20 so as to have good computational security.

7. Conclusions and Future Work

In this paper, we have proposed a computationally efficient, yet secure video encryption scheme. It uses RC5 for encryption of the DCT coefficients. The proposed scheme is very fast, possesses good security and adds less overhead on the codec. It slightly decreases the compression rate of the video, which is negotiable for higher security. Our future work would be to reduce the encrypted video size by modifying the default Huffman tables and hence come up with an ideal video encryption algorithm which takes less encryption time and causes no overhead on video size. It can also be extended to videos like MPEG-4, H.261, and H.264 etc.

References

[1] I. Agi and L. Gong. An empirical study of MPEG video transmission. In *Proc. of the Internet Society Symposium on Network and Distributed Systems Security*, pages 137–144, 1996.

[2] A. Biryukov and E. Kushilevitz. From differential cryptanalysis to ciphertext-only attacks. *Lecture Notes in Computer Science, Advances in Cryptology – Proceedings of CRYPTO'98*, 1462:72–88, 1998.

[3] A. Biryukov and E. Kushilevitz. Improved cryptanalysis of RC5. *Lecture Notes in Computer Science*, 1403:85–100, 1998.

[4] Z. Chen, Z. Xiong, and L. Tang. A novel scrambling scheme for digital video encryption. In *Proc. of Pacific-Rim Symposium on Image and Video Technology (PSIVT)*, pages 997–1006, 2006.

[5] E. Choo, L. Jehyun, L. Heejo, and N. Giwon. SRMT: A lightweight encryption scheme for secure real-time multimedia transmission. In *Multimedia and Ubiquitous Engineering*, pages 60–65, 2007.

[6] L. S. Choon, A. Samsudin, and R. Budiarto. Lightweight and cost-effective MPEG video encryption. In *Proc. of Information and Communication Technologies: From Theory to Applications*, pages 525–526, 2004.

[7] T. B. Maples and G. A. Spanos. Performance Study of a Selective Encryption Scheme for the Security of Networked, Real-Time Video. In *Proc. of Fourth International Workshop on Multimedia Software Development '96)*, 1995.

[8] L. Qiao and K. Nahrstedt. A new algorithm for MPEG video encryption. In *Proc. of First International Conference on Imaging Science System and Technology*, pages 21–29, 1997.

[9] R. L. Rivest. The RC5 encryption algorithm. In *Proc. of the Second International Workshop on Fast Software Encryption (FSE)*, pages 86–96, 1994.

[10] B. Schneier. Applied Cryptography Second Edition Protocols Algorithms and Source Codes in C, 1996.

[11] L. Tang. Methods for encrypting and decrypting MPEG video data efficiently. In *Proc. of ACM Multimedia*, pages 219–229, 1996.

[12] W. Tsang, B. Smith, S. Mukhopadhyay, H. H. Chan, S. Weiss, M. Chiu, and J. Song. The DALI multimedia software library. In *IEEE 2nd Workshop on Multimedia Signal Processing*, 1999.

[13] P. N. Tudor. MPEG-2 video compression. In *Electronics and Communication Engineering Journal*, December - 1995.

[14] T. Uehara and R. Safavi-Naini. Recovering DC coefficients in block-based DCT. In *Proc. of IEEE Transactions on Image Processing*, volume II, pages 3592–3596, 2006.

[15] W. Zeng and S. Lei. Efficient frequency domain selective scrambling of digital video. In *Proc. of the IEEE Transactions on Multimedia*, pages 118–129, 2002.