Journal of Theoretical and Applied Information Technology
15 July 2012. Vol. 41 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

PUZZLE FAST RANDOM BIT ENCRYPTION TECHNIQUE FOR JOINT VIDEO COMPRESSION AND ENCRYPTION

[1]K. JOHN SINGH, [2]R. MANIMEGALAI

[1]Assistant Professor (Senior), School of Information Technology and Engineering
VIT University, Vellore, Tamil Nadu, India
[2]Research Supervisor, Anna University of Technology Coimbatore
Tamil Nadu, India
E-mail: johnsinghaj@yahoo.com , mmegalai@yahoo.com

ABSTRACT

Major issues in video data transfer over the internet are security and speed. For fast transmission over the network, the video data should be compressed before transmission. Joint compression and encryption algorithms employ compression before encryption for secured and fast data transfer over the internet. In this paper, a joint compression and encryption algorithm, Puzzle Fast Random Bit Encryption (PFRBE) algorithm, is proposed. As the proposed solution employs multi-level encryption along with key encryption, it is more secured than existing algorithms. Results obtained are compared with some popular joint compression and encryption algorithms such as Video Encryption Algorithm (VEA) and Real-time Video Encryption Algorithm (RVEA). Our analysis shows that the proposed solution, PFRBE, takes less CPU time and consumes less memory when compared to existing joint compression and encryption algorithms.

Keywords: *Video Encryption Algorithm (VEA), Puzzle Fast Random Bit Encryption (PFRBE), Real-time Video Encryption Algorithm (RVEA), Bit Padding*

1. INTRODUCTION

The revolution of multimedia and hypermedia has been a driving force behind fast and secured data transmission techniques. In general, video data takes more time for encryption, because of its large size. Since the size of video data is huge in volume, it needs to be compressed and encrypted to avoid security threats and delay. There are two strategies for this, namely, compression-independent encryption algorithms and joint compression and encryption algorithm. In compression-independent encryption algorithms, both compression and encryption are done independently as two different steps by employing suitable algorithms. This strategy consumes more time and memory. As the computation time is increased, overall system performance is decreased when compression-independent encryption algorithms are applied. But in joint compression and encryption algorithm, both the steps, namely, compression and encryption are integrated together as a single step. There are two approaches for joint compression and encryption algorithm: the first method employs encryption after compression and the second one does encryption before compression. Steps involved in both the approaches are illustrated in Fig. 1(a) and Fig. 1(b). In the first strategy, as encryption is done after compression we get two-fold advantages, namely, reduced data size and time. The second strategy encrypts data without compression and is time consuming. In general, any joint compression and encryption algorithm will provide two levels of security and consumes less time when compared to compression-independent encryption algorithms. Secure Motion Picture Experts Group (SECMPEG) [1], Video Encryption Algorithm (VEA) [2], Real-time Video Encryption Algorithm (RVEA) [3], are few examples for joint compression and encryption algorithms. Joint compression and encryption algorithms are faster in encrypting the video data due to selective encryption technique when compared to other video encryption algorithms.

Journal of Theoretical and Applied Information Technology
15 July 2012. Vol. 41 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

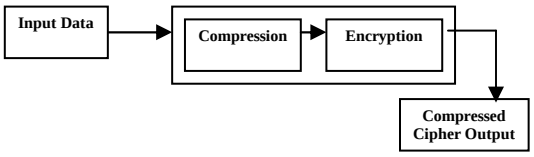

Fig. 1. (a): Compression Before Encryption

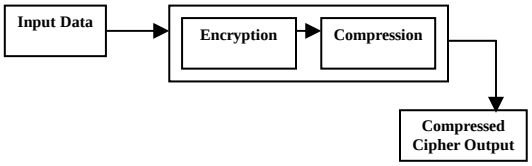

Fig. 1. (b): Compression After Encryption

The rest of the paper is organized as follows: The literature review is presented in Section 2. The proposed solution, Puzzle Fast Random Bit Encryption (PFRBE) algorithm is explained in Section 3. Section 4 presents the experimental results and Section 5 concludes the paper.

2. RELATED WORKS

Zeng et.al, have proposed a frequency domain scrambling approach in [4] which performs encryption after the Discrete Cosine Transform (DCT). Compression is done only on scrambled frames leading to low image quality. In addition, the proposed solution in [4] tends to consume more memory due to uncompressed frames.

The proposed solution in [5] performs compression and encryption with minimum overhead using random permutation and probabilistic encryption. It provides different levels of secrecy for various multimedia applications. The proposed strategy in [5] employs Discrete Cosine Transformation (DCT) to map smaller blocks with size 8 x 8 to bigger blocks with size 1x64. The output from DCT is uniformly quantized and all quantized coefficients are arranged in zig-zag order. Finally entropy coding is done for compression.

The joint compression and encryption algorithm proposed in [1], SECMPEG, does selective encryption using conventional encryption algorithms. The Video Encryption Algorithm (VEA) proposed in [2] encrypts all sign bits of DCT coefficients by using XOR – operation. VEA has the disadvantages of having known-plaintext attack and complex key management scheme. In known-plaintext-attack, if both the original and encrypted videos are available, the attacker can easily determine the secret key. To overcome known-plaintext attack, Shi et al., have proposed Real-time Video Encryption Algorithm (RVEA) in [3]. The XOR operation in VEA [2] is replaced with a conventional encryption algorithm in RVEA [3]. RVEA is a selective encryption algorithm which operates on the sign bits of both DCT coefficients and motion vectors of a MPEG compressed video. RVEA can use any secret key cryptography algorithms to encrypt selected sign bits. The proposed solution in [6], Multiple Huffman Table (MHT) converts entropy coders into encryption ciphers. The computational cost of this algorithm is less but this is more vulnerable to chosen-plaintext attack.

3. PUZZLE FAST RANDOM BIT ENCRYPTION

The steps employed in the proposed methodology for joint compression and encryption of video data, Puzzle Fast Random Bit Encryption (PFRBE) method, are shown in Fig. 2. There are two phases in the proposed solution, namely, compression process and encryption process. Compression process is done using a three-step procedure called Entropy Based Random Arithmetic Puzzle Transform (EBRAPT) method. Compressed video data is obtained by applying three steps, namely, entropy coding, puzzle transform technique and randomized arithmetic coding as shown in Fig. 2. The encryption process is done using Fast Random Bit Encryption (FRBE) [7]. Comparatively, encryption applied on compressed video data takes less time and thereby leads to high efficiency. In the second stage, the compressed data is encrypted using the DES algorithm [8]. The key is divided into four parts and is encrypted with a random number. Since both sender and receiver can use the same random number generator, key encryption solves key management and distribution problems**.** Bit padding is applied after encryption. In bit padding, a single set-bit (‘1’) is added along with many reset-bits (‘0’) to the data that is being encrypted. Hash function MD5 (Message Digest) is applied to convert the data with an arbitrary size into a fixed length hash value. Then, the hash values generated are encrypted using Salt algorithm [9]. It gives an

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

intermediate encrypted data which is called as Salt. The Salt consists of a random bit which is used to generate the key using a key derivation function PBKDF2 (Password-Based Key Derivation Function) [9]. Next, the key is encrypted using a random number and PKCS7 (Public-Key Cryptography Standards) padding is applied on the encrypted key.

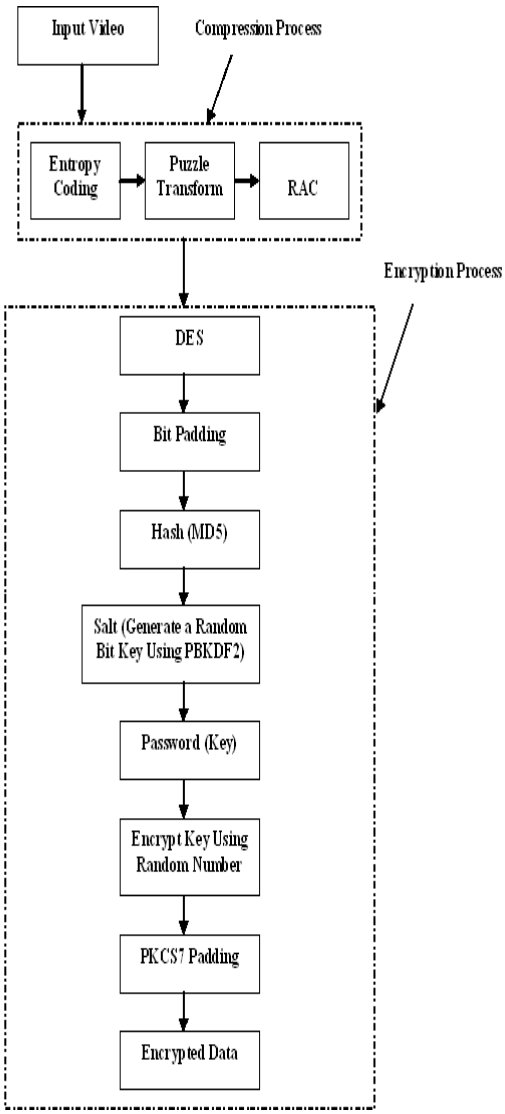

Fig. 2: Puzzle Fast Random Bit Encryption (PFRBE) Algorithm

3.1 Compression

Compression of the video is done using our proposed method called, Entropy Based Random Arithmetic Puzzle Transform (EBRAPT). During video compression, it is possible that the resolution and edges may likely to be changed which in turn will affect the quality. Special care should be taken to compress the data without changing its basic qualities. In EBRAPT, video data is compressed without changing its resolution. The steps employed in the proposed compression process (EBRAPT), namely, Entropy Coding, Puzzle Transform and Randomized Arithmetic Coding, are explained in the following subsections.

3.1.1 Entropy coding

The first step in compressing video data is entropy coding which is lossless compression technique. For a given video data, entropy coding generates variable length code called prefix code along with prefix value of each pixel. Each pixel value has common prefix value which helps in decompressing the data easily. Entropy coding generates intermediate compressed data.

3.1.2 Puzzle transform

The second step in the process of compression is puzzle transform technique. The intermediate compressed data, output from entropy coding, is given as input to puzzle transform technique. The intermediate compressed data is split into 8x8 block segments and then, the order of segments is changed randomly. It is assumed that the intermediate compressed frame is split into 526 blocks, namely, S_1, S_2,……, S_{526} as shown in Fig. 3. After applying puzzle transform method, we may get the order of blocks in random order, for example, S_{526}, S_{512}, S_{435}, …….., S_{30}. It is observed that puzzle transform method provides better security to an intermediate compressed data and therefore, the attacker cannot view or access the data.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

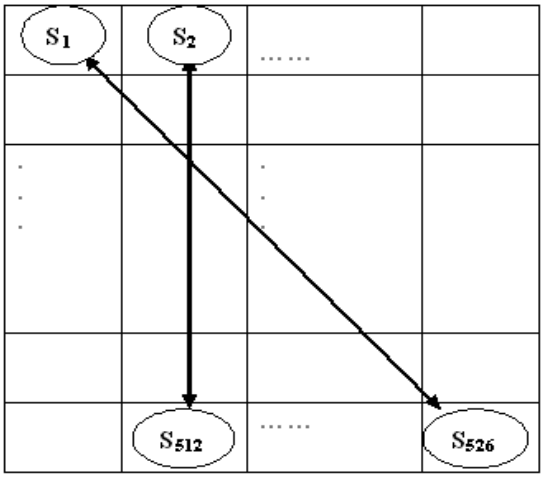

Fig. 3: 8x8 –Puzzle Transform

3.1.3 Randomized arithmetic coding

The third and final step in compression is randomized arithmetic coding. In this method, frequently used pixel value from each blocks are stored with fewer bits. Not so frequently used pixel values are stored with more bits. For example, consider that a pixel value 56 has occurred 18 times and 112 have occurred 24 times. Then, substitute 56 with 18 and 112 with 24. Here, replace its actual pixels with new values. This is one form of encryption. So it gives more security to the data. Suppose, P is the pixel and its number of occurrence is F_k. Now replace the entire P pixel with F_k. Suppose k=1, that is the number of occurrence is zero, then substitute all the pixel values with 1.

Table 1: 8x8 –Pixel Order

52	64	27	118	54	78	98	234
14	422	78	302	68	435	14	78
508	512	78	512	27	347	345	91
78	110	27	129	118	27	354	406
345	378	238	236	518	14	12	8
28	126	46	27	512	34	14	78
22	27	6	512	457	27	148	190
278	128	98	64	116	78	118	422

Table 2: 8x8 –Randomized Arithmetic Coded Pixel Order

52	2	6	3	54	7	2	234
4	2	7	302	68	435	4	7
508	3	7	3	6	347	2	91
7	110	6	129	3	6	354	406
2	378	238	236	518	4	12	8
28	126	46	6	3	34	4	7
22	6	6	512	457	6	148	190
278	128	2	2	116	7	3	2

Table.1 shows the 8x8 pixel arrangement for a single video frame. It should be noted that the table has same number of occurrences for some of the pixels. The pixel values in Table.1 are replaced with their number of occurrence by employing Randomized Arithmetic Coding technique. The output, i.e. the pixel arrangement after Randomized Arithmetic Coding is shown in Table 2.

3.2 Encryption

As illustrated in Fig. 2, the encryption process follows the compression process. Encryption is divided into three phases. In the first phase, namely, video encryption phase, encryption is done with padding and salt algorithm. At the end of the first phase 64-bit key is generated using the salt algorithm namely, PBKDF2 [9]. In the second phase, the key is divided into four parts with 24, 16, 16 and 8 bits and encrypted separately using random number. In the third phase, encryption is done using PKCS7 padding [11]. All phases are explained in detail in the next few sub-sections.

3.2.1 Video encryption using padding and salt

The FileInputStream and FileOutputStream are used to read and write raw bytes of data such as DES cipher and image data. Initially, the video data is converted in to frames and each frame is treated as an image [12]. Then DES encryption is applied to generate block ciphers. These block ciphers are converted to an

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

arbitrary sized data using bit padding method. In bit-padding, only a selected block of ciphers are padded resulting block cipher with few padded bit blocks. Normally bit padding is applied with 128 bit block cipher. In this work, bit padding with more than 128 bits is employed. As explained earlier additional bits are added with the key to provide more security. After encryption, each block of the file is stored in the buffer with size 64 bits. Then, hash method, MD5, is applied to convert arbitrary sized data into fixed length hash value [9]. Finally, salt method is applied to generate the key with random bits.

3.2.2 Key encryption using random number

In general, when video data is sent over the network both the sender and receiver will have the same random number generator and we assume the same in this work. The key is encrypted using random number. The generated key is divided into four parts as shown in Fig. 4. Encryption of the key is done using XOR operation to each part separately [12, 13]. Finally, each part is decrypted and they are joined to get the original key.

24	16	16	8

Fig. 4: Key Segment - During Encryption Using Random Number

3.2.3 Padding using PKCS7

In the first two phases, the data and the key are encrypted selectively. In the third phase, the data and key are padded using PKCS7 standard yielding the final encrypted block cipher and key cipher.

4. EXPERIMENTAL RESULTS AND ANALYSIS

To analyze the performance of the proposed algorithm, metrics such as execution time, CPU utilization and memory utilization are considered. Tables 3, 4, and 5 show that the proposed algorithm, PFRBE, is optimized and giving better performance in terms of execution time, CPU utilization ratio and memory consumption. PFRBE is faster and consume less memory when compared to other joint compression and encryption algorithms.

Table 3: PFRBE vs. Existing Algorithms: Comparison Based on Execution Time

File Name	File size in bytes	Execution Time in Seconds				
		SECMPEG [1]	VEA [2]	RVEA [3]	Zigzag [5]	PFRBE
car.flv	2560	20	25	15	18	10
planet.flv	4798	23	27	18	20	14
flow.flv	5262	26	29	20	24	17
boat.flv	6440	31	32	24	27	21

Table 4: PFRBE vs. Existing Algorithms: Comparison Based on CPU Utilization Ratio

File Name	File size in bytes	CPU Utilization Ratio in %				
		SECMPEG [1]	VEA [2]	RVEA [3]	Zigzag [5]	PFRBE
car.flv	2560	44	48	40	41	36
planet.flv	4798	45	52	44	45	39
flow.flv	5262	47	57	49	50	43
boat.flv	6440	62	63	57	58	50

Table 5: PFRBE vs. Existing Algorithms: Comparison Based on Memory Utilization

File Name	File size in bytes	Memory Utilization in Mbytes				
		SECMPEG [1]	VEA [2]	RVEA [3]	Zigzag [5]	PFRBE
car.flv	2560	952	974	951	953	942
planet.flv	4798	958	979	957	962	948
flow.flv	5262	966	983	964	968	954
boat.flv	6440	974	994	971	982	963

The reduced compression time and CPU utilization as shown in Tables 3 and 4 is due to reduced compressed video by intermediate steps such as cipher generation, key generation, key encryption and video cipher generation. From Table 5 the proposed joint compression and encryption method, Puzzle Fast Random Bit Encryption, consumes less memory than the existing algorithms. This is mainly due to the compressed video before encryption.

Table 6: Compression efficiency of PFRBE algorithm

File Name	File size in bytes	Encrypted File Size in bytes	Compression (%)	PSNR
car.flv	2560	1664	65	35.000
planet.flv	4798	3358	70	30.012
flow.flv	5262	3525	67	33.010
boat.flv	6440	4656	72	27.701

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 6 shows the results of compression ratio and Peak Signal to Noise Ratio (PSNR) for various files using the proposed solution. The high compression ratio indicates better performance of the proposed solution.

5. CONCLUSION

When confidential video data is sent over the network, two key factors to be considered are security and speed. The joint compression and encryption algorithms address the above two factors; they provide additional security when compared to other video encryption algorithms and consume less resource during encryption process [14]. In this paper, a joint compression and encryption algorithm, namely, Puzzle Fast Random Bit Encryption (PFRBE) method is proposed for fast and secured video transfer. Our observation shows that if the key is encrypted along with video data, it is not possible to break the key easily. Results obtained clearly indicate that the proposed method has the advantage of increased security and speed over the existing algorithms. The proposed method considers the compression and encryption of video data. Extending this approach to audio data and combined video-audio data is an interesting problem.

REFERENCES:

[1] Meyer J, Gadegast F., "Security Mechanisms for Multimedia Data with the example MPEG-1 video", Project Description of SECMPEG, Technical University of Berlin, 1995.

[2] Shi C, Bhargava B, "A Fast MPEG Video Encryption Algorithm", *In Proceedings of 6th ACM International Conference on Multimedia*, pp. 81-88, 1998.

[3] Shi C, Wang SY, Bhargava B., "MPEG Video Encryption in Real-time Using Secret Key Cryptography", *In Proceedings of International Conference on Parallel and Distributed Processing Techniques and Applications*, pp. 2822-2828, June 1999.

[4] Zeng W, Lei S., "Efficient Frequency Domain Selective Scrambling of Digital Video", *IEEE Transactions on Multimedia*, pp. 118-129, 2003; 5(1).

[5] Tang L., "Methods for Encrypting and Decrypting MPEG Video Data Efficiently", *In Proceedings of ACM International Conference on Multimedia*, pp. 219-229, November 1996.

[6] Wu C-P, Kuo C-CJ., "Design of Integrated Multimedia Compression and Encryption Systems", *IEEE Transactions on Multimedia*, pp. 829-839, October 2005; 7(5).

[7] K. John Singh and R. Manimegalai, "Fast Random Bit Encryption Technique for Video Data", *European Journal of Scientific Research*, Vol.64, No. 3, pp 437-445, November 2011.

[8] Dominik Engel and Andreas Uhl, "Secret Wavelet Packet Decompositions for JPEG 2000 Lightweight Encryption", *In Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 5, pp. 465-468, May 2006.

[9] A. S. Tosun and W. Feng, "A Lightweight Mechanism for Securing Multi-layer Video Streams", *In Proceedings of IEEE International Conference on Information Technology: Coding and Computing*, pp. 157-161, April 2001.

[10] Timothy E. Lindquist, Mohamed Diarra and Bruce R. Millard, "A Java Cryptography Service Provider Implementing One-Time Pad", *In Proceedings of IEEE International Conference on System Sciences*, pp. 1-6, January 2004.

[11] Anil Kr. Yekkala, Narendranath Udupa, Nagaraju Bussa and C.E. Veni Madhavan, "Lightweight Encryption for Images", *In Proceedings of IEEE International Conference on Consumer Electronics*, pp. 1-2, January 2007.

[12] Susie Wee and John Apostolopoulos, "Secure Scalable Streaming and Secure Transcoding with JPEG2000", *In Proceedings of IEEE International Conference on Image Processing*, vol. 1, pp. 205–208, September 2003.

[13] Simon Fong, "On Improving the Lightweight Video Encryption Algorithms for Real-time Video Transmission", *In Proceedings of IEEE third International on Communications and Networking*, pp.1287-1293, August 2008.

[14] K. John Singh and R. Manimegalai, "A Survey on Joint Compression and Encryption Techniques for Video Data", Journal of Computer Science, Vol. 8, No. 5, pp 731-736, February 2012.