# MPEG Encryption by Zigzag, Partitioning and Swapping

**A. F. M. Suaib Akhter [†], Saiful Islam[††], Md. Jakir Hossain[†††], Rupam Deb[††††], Dr. Md. Basir Uddin[†††††]**

Department of Computer Science and Engineering, Dhaka International University, Bangladesh[†]
Department of Electrical, Electronic and Telecommunication Engineering, Dhaka International University, Bangladesh[††]
Department of Electrical and Electronic Engineering, Dhaka University of Engineering and Technology, Bangladesh [†††-†††††]

**Abstract:**
As transferring video through internet and cellular phone become popular day by day, the importance of encrypting some of these also increases. To achieve that, many researchers proposed a lot of algorithms to encrypt MPEG files. In this paper we have proposed a new algorithm for MPEG encryption by using combination of Zigzag, Partitioning and Swapping. In this algorithm encryption done by rearranging the bits and also swapping of blocks of the MPEG file. As we are using N-bit partitioning there are $2^N$ partitions and will apply zigzag rule on each of the partitions. Then will rearrange M-number of partitions by swapping. For decryption, key will guide to rearrange the partitions and then the procedure is just opposite of the encryption. As we are only rearranging the bit and block sequences the file size remains the same after encryption. Increasing the value of N as well as M will increases the complexity of the algorithm. Except this, the major advantage is that it needs a very short time to encrypt and decrypt but near to infinite time for cryptanalysis.

***Key words:***
*Cryptography, MPEG, Partitioning, Swapping, Zigzag.*

## 1. Introduction

As the internet speed is becoming higher and higher, information passing through video or MPEG format becomes more popular then text or image. At the same time to ensure the MPEG data security the necessity of effective encryption algorithm becoming higher and higher. Today, there are many powerful cryptographic algorithms in the marketplace for text data. These algorithms are not suitable for video. Due to high size and the relationship between the pixels the value of any given pixel can be reasonably predicted from the value its neighbors [1].

There are several algorithms to solve video encryption problem. Most of them are complicated and take higher time and space for encryption and decryption. Some of them are not considered as secure. So retrieving particular video in a way that is both effective and efficient remains an open problem. It is necessary to design special encryption algorithms for multimedia data because of their special characteristics such as its coding structure, large amount of data, and real-time constraints [2].

Among them the speed of Zigzag-Permutation Algorithm is very fast and is almost the same as the MPEG encoding/decoding time [3]. But this algorithm compromised security as it does not withstand the known-plaintext attack, therefore should NOT be considered as secure [4].

In our proposed algorithm rather then making it complex our intention is to change the bit sequences and also some of the blocks of the target file using several ways. And to remove the problem of zigzag permutation we use partitioning and swapping with zigzag. As the number of partitions and also number of swaps are not constant, it will not further vulnerable for known-plaintext attack. Moreover our proposed method will overcome the size problem as the encrypted file size will remain the same as input file.

We have tried to encrypt MPEG file because "Principles are the same and a suitable MPEG-1 cipher can be modified for MPEG-2 or MPEG-4" [5].

## 2. Partitioning Method, Zigzag rule, Swapping and Decryption

### 2.1 Partitioning Method

Partitioning is possible in various ways. For example, let N = 4 bits. Using 4 bits we can partition bit sequences into $2^4$ or 16 parts. Each part will get equal number of bits. If there is m-number of bits in the data file, each partition will get the ceiling of $[M/2^N]$ number of bits. So, except the last one each partition will get same number of bits.

Let, the data bit is 1000 bits long. As N= 4 each partition will get ($1000/2^4 = 62.5$ or) 63 bits. And thus in the last or $2^{N\text{ th}}$ partition there will (1000 - 63x15) 55bits.

Manuscript received May 5, 2010
Manuscript revised May 20, 2010

## 2.2 Zigzag rule

A line or course that proceeds by sharp turns in alternating directions is known as zigzag rule. By this rule first bit of selected bits will be placed in right position. Again second bit will place in left and third bit will be placed in right position and so on. Here is an example of zigzag:

Let's select two bit BIT = "1010010100101011"
$BIT_{Zigzag}$= [Empty]
First bit of BIT = 1
It will be placed in the right position of $BIT_{Zigzag}$.
So, $BIT_{Zigzag}$= "1"

Second bit of BIT = 0
It will be placed in the left position of $BIT_{Zigzag}$.
So, $BIT_{Zigzag}$ = "01"

Third bit of BIT=1
It will be placed in the right position of $BIT_{Zigzag}$.
So, $BIT_{Zigzag}$ = "011"

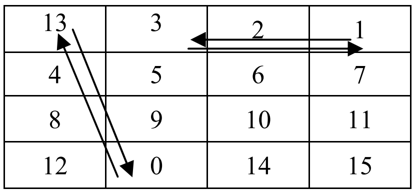
Forth bit of BIT= 0
It will be placed in the left position of $BIT_{Zigzag}$.
So, $BIT_{Zigzag}$ = "0011"

After placing all bits BIT will become
$BIT_{Zigzag}$ = "1000110011000111"

By this procedure all bits of selected partition will be converted. Then position of the partitions will also change by swapping.

## 2.3 Swapping

In our key first i-bit represent the value of N and remaining j-bits used for swapping. Each N number of bits from j will select the partitions for swapping.
Number of swap (M) = j/N;
Let, KEY = <u>0100</u> <u>1101 0011</u>.
Here i = 4 so j = key-4 = 8.
So, M= 8/4 = 2 swaps.
As, N = $0100_b = 4_d$ there will $2^4 = 16$ partitions. Then $j1=1101_b = 13_d$, so 13th partition will swap with the 1st partition. The arrow signs show how the position of the partitions will change by swapping.



| | | | |
|---|---|---|---|
| 13 | 3 | 2 | 1 |
| 4 | 5 | 6 | 7 |
| 8 | 9 | 10 | 11 |
| 12 | 0 | 14 | 15 |

## 2.3 Decryption

Decryption process generally depends on the key. As KEY = i+j, so from first i-bits the value of N can be determined. Then each N-bit from j will select the partitions to swap.
For example, when
KEY = 0100 1101 0011 and i = 4:
0100 = 4 = N so there are $2^N$ or 16 partition. Remaining 8 bits (j) divided by 4 will give M = 2. Now each N or 4-bit will converted into decimal and declare which partitions to swap.

As we have got the value of N, following the same procedure used in encryption method number of bits in each partition can be determinable. Then reverse of zigzag rule will apply on each partition. After that, by rearranging the partitions in reverse order) according to the key will return the original data bits.

## 3. Algorithm for Encryption and Decryption

### 3.1 Algorithm for Encryption

```
    //Input: MPEG file ($M_F$, KEY)
    //Output: Encrypted MPEG file ($M_F$)
 Algorithm Encryption ($M_F$, KEY)

  Let $T_F$ is any temporary MPEG file Name;
  KEY = i + j ;
  N ← Fetch first i-bit from KEY and convert into
        Decimal;

  $BIT_{Partition}$ ← Ceiling of [$M_F / 2^N$];
  Z ← 0 ;
  FOR a ← 0 to $2^N$-1
          Partition[a] ← Append bits from Z to
                          $BIT_{Partition}$ th position ;
          Z ← Z + $BIT_{Partition}$ ;
          Z++ ;
          X ← Partition[a];
          ENCR[a] ← Zigzag (X);

  p ← 0 ;
  b ← 0 ;
  While ( b != j )
          q ← Fetch N-bit from j[b]th position and
                convert into decimal;
          Swap Partition[p] with partition[q];
          b ← b + (N+1) ;
          p++ ;
  END WHILE;

  $T_F$ ← Concatenation of bits from ENCR [0] to
```

```
        ENCR [$2^N$-1] according to their new Position.

  Delete $M_F$
  Rename $T_F$ to Original File ($M_F$)

 FUNCTION Zigzag (X)
     LeftRight ← 1;
     $BIT_{Zigzag}$ ← "";
     FOR i ← 0 to Length [X]
         IF (LeftRight = = -1)
                 Reverse the $BIT_{Zigzag}$ ;
                 Concatenation $BIT_{Zigzag}$ with X[i];
                 Reverse the $BIT_{Zigzag}$ ;
                 LeftRight ← LeftRight * -1;
         ELSE
                 Concatenation $BIT_{Zigzag}$ with X[i];

     RETURN $BIT_{Zigzag}$;
```

## 3.2 Algorithm for Decryption

In decryption a new function ZigzagReverse(X) used instead of Zigzag(X). Remaining of the algorithm are same as encryption.

```
     //Input: Encrypted MPEG file ($M_F$), Key (KEY)
     //Output: MPEG file ($M_F$),

 Algorithm Decryption ($M_F$, KEY)

  Let $T_F$ is any temporary MPEG file Name;

  N ← Fetch first 4-bit from KEY and convert into
        Decimal.
  $BIT_{Partition}$ ← Ceiling of [$M_F$ / $2^N$];
  M ← 0;
  FOR i ← 0 to $2^N$-1
          Partition[i] ← Append bits from M to
                         $BIT_{Partition}$ th position ;
          M ← M + $BIT_{Partition}$ ;
          M++ ;
          X ← Partition[i];
          ENCR[i] ← ZigzagReverse (X);

  p ← M ;
  b ← j ;
  While ( b != 0 )
          q ← Fetch previous N-bit from j[b]th position and
              convert into decimal;
          Swap Partition[p] with partition[q];
          b ← b - (N+1) ;
          p-- ;
  END WHILE;

  $T_F$ ← Concatenation bits from ENCR [0] to
          ENCR [$2^N$-1] according to their new Position.

  Delete $M_F$
  Rename $T_F$ to Original File ($M_F$)
 FUNCTION ZigzagReverse(X)

     $BIT_{ZR}$ ← " ";

     FOR i ← 0 to Length[X]
         Concatenation of ( X[0 to mid-1] ) with
         ( X[mid+1 to Length[X] );
         mid ← Length[X] / 2 ;
         Concatenation of ( $BIT_{ZR}$) with ( X [mid]);

 RETURN $BIT_{ZR}$;
```

## 4. Performance, Depth and Security analysis

### 4.1 Performance Analysis

We analyze 5 MPEG files by this method. We used Intel(R) Core(TM)2 Dou CPU, E 4600 @2.40 GHz, 2MB of RAM machine. After running the software build in C we get following data (in Table-2).

Table 2: Performance Analysis

| Name | Size (KB) | Encryption Time (in second) | Size (KB) [After Encryption] |
|---|---|---|---|
| a.mpeg | 9,644 | 0.274 | 9,644 |
| b.mpeg | 25,067 | 0.824 | 25,067 |
| c.mpeg | 36,336 | 1.593 | 36,336 |
| d.mpeg | 41,798 | 1.264 | 41,798 |
| e.mpeg | 72,484 | 2.308 | 72,484 |

Here it is found that, it requires a very short time to encrypt. Another thing is noticeable that the encrypted file sizes remain the same as main file.

### 4.2 Depth of Encryption

In this algorithm MPEG files are encrypted using three methods. So, the cryptanalysis needs to determine the random partitions, then after applying reverse of zigzag again he needs to swap some partitions. Thus, if the key is known also, determination of the original file is still hardly impossible.

Partitioning using N-bits there are $2^N$ partitions. If the file size is S-bit then $2^S$ number of operations needed to determine the original file. Moreover to determine the swap, at least $2^M$ number of operations is needed. So, total $2^N * 2^S * 2^M$ number of operation is needed for cryptanalysis. That is quit impossible. If anyone try for cryptanalysis and fail to do it properly, then file format with information will be changed and never not possible to get original file i.e. file will be corrupted. So, it is a highly secured algorithm.

### 4.3 Security Analysis

Zigzag, Partitioning and swapping algorithm basically change the sequence of the bits as well as blocks of a video file to encrypt. Zigzag permutation is another algorithm which also encrypt video file by changing the bit sequences. But, this algorithm compromised security as the permutation list is vulnerable for known plaintext attack [2].

In our algorithm rather then using permutation we are using partitioning and swapping with zigzag. Moreover the value of N and M are not fixed. Security of the algorithm increases if the value of N or M increase. Although the determination of key will not suggest for zigzagging or partitioning even swapping, algorithm is safe from known-plaintext attack.

Another encryption algorithm called Video Encryption Algorithm (VEA) used key for encryption. But, the algorithm suffers from increasing key size problem as there are several keys are used in the Video Encryption Algorithm [4].

We also have to use key but size is not fixed. If we increase the value of N or M the size of the key will increase, but at the same time the complexity will increase much higher compared to size. Moreover, as it does not increase the size of the original file this key size is affordable.

## Conclusion:

There is no algorithm satisfy both security, and speed requirements trade-off [6]. But, in zigzag, partitioning and swapping algorithm within a very short amount of time we can provide a secure encryption without increasing the size of the file. We can say that our research decreases the gap between security and speed. By increasing value of N or M we can make it more secure and complex. Decryption algorithm is very near to the encryption algorithm and both are time consuming. As we are just changing the bit sequences it is also applicable for encrypting sound, image or text file. In future we will try to encrypt in such a way so that the encrypted file size become less then the original file (compressed).

## References

[1] Schneier, B., Applied Cryptography: Protocols, Algorithms. And Source Code in C, 2nd edition, John Wilen & Sons, Inc., 1996.
[2] L. Tang, "Methods for Encrypting and Decrypting MPEG Video Data Efficiently ," In Proceedings of The Fourth ACM International Multimedia Conference (ACM Multimedia'96), pages 219-230, Boston , MA, November 1996.
[3] J. But, G. Armitage, "An Evaluation of Current MPEG-1 Ciphers and their Applicability to Streaming Video," Australian Telecommunications Networks & Applications Conference 2004, (ATNAC2004), Sydney, Australia, December 8-10, 2004.
[4] X. Liu and A. M. Eskicioglu, "Selective Encryption of Multimedia Contents in Distribution Networks: Challenges and New Directions," IASTED International Conference on Communications, Internet, and Information Technology (CIIT 2003), Scottsdale, AZ, November 17-19,2003.
[5] J. But, G. Armitage, "An Evaluation of Current MPEG-1 Ciphers and their Applicability to Streaming Video," Australian Telecommunications Networks & Applications Conference 2004, ( ATNAC2004), Sydney, Australia, December 8-10, 2004
[6] M. A. Hashish, “The MPEG Encryption Algorithms, Survey and Analysis” http://www.mohamedhashish.com/index.php

**A. F. M. Suaib Akhter** who born in Rajshahi, Bangladesh, has completed B. Sc. in the Department of Computer Science and Information Technology (CIT) from Islamic University of Technology(IUT), Gazipur, Bangladesh. With keen interest in network security have done several research in the field of Network security specially in Cryptography. Now he is working as a Lecturer in Computer Science and Engineering(CIT) dept. in Dhaka International University, Dhaka, Bangladesh.

**Saiful Islam** who born in Kotwalpara, Gopalgonj, Bangladesh, has completed M. Sc. in the Department of Electrical & Electronic Engineering (EEE) and B.Sc. in Engineering in Comuter Science & Engineering (CSE) from Dhaka University of Engineering & Technology (DUET), Gazipur, Bangladesh. His research interest is Cryptography He is a Senior Lecturer in Electricl, Electronics and Telecommunication Engineering dept from 2006 to 2010 in Dhaka International University, Dhaka, Bangladesh.

**Mohammad Jakir Hossain** who born in Shobharampur, Comilla, Bangladesh. He obtained his M. Sc. Engineering degree from Electrical & Electronic Engineering (EEE) department and B.Sc. Engineering degree from Electrical and Electronic Engineering (EEE) department from Dhaka University of Engineering & Technology (DUET), Gazipur, Bangladesh. His research interest is Image processing. He is an Assistant Professor in Electrical and Electronic Engineering dept. in Dhaka University of Engineering & Technology, Gazipur, Dhaka, Bangladesh. Mr. Hossain is now Member of the Engineers Institution, Bangladesh (IEB).

**Rupam Deb** who born in Hathazari, Chittagong, Bangladesh. He obtained his B.Sc. Engineering degree from Computer Science and Engineering (CSE) department of Dhaka University of Engineering & Technology (DUET), Gazipur, Bangladesh. He has interest in Image processing and Software Engineering. He is an Assistant Professor in Computer Science and Engineering dept. in Dhaka University of Engineering & Technology, Gazipur, Dhaka, Bangladesh. Mr. Deb is now Member of the Engineers Institution, Bangladesh (IEB).

**Dr. Md. Bashir Uddin**
B.Sc Engg. (BITR), M.Sc Engg. & Ph. D. (BUET)