# Improved Selective Encryption Techniques for Secure Transmission of MPEG Video Bit-Streams

Adnan M. Alattar
*Digimarc Corporation*
*One Centerpointe Drive, Suite 500*
*Lake Oswego, OR 97035-8615*
*e-mail: aattar@digimarc.com*

Ghassan I. Al-Regib[‡] and Saud A. Al-Semari[#]
*Electrical Engineering Department*
*King Fahd University of Petroleum and Minerals*
*Dhahran 31261, Saudi Arabia*
*e-mail:[‡]g929816@kfupm.edu.sa* *[#]semari@kfupm.edu.sa*

## Abstract

*This paper presents three new selective encryption techniques for secure transmission of MPEG-I video bit-streams. These techniques maintain higher security levels than previously proposed selective encryption techniques while maintaining reasonable processing times. In the first of these methods, the encryption is applied to the data associated with every $n^{th}$ I- macroblock. In the second method, the encryption is applied to the headers of all the predicted macroblocks as well as to the data associated with every $n^{th}$ I-macroblock. In the third method, encryption is applied to every $n^{th}$ I- macroblock as well as the header of every $n^{th}$ predicted macroblock. The last method, with n=2, is found to be the most efficient of the three proposed methods. This method achieves a 60-82% reduction in the processing time over "total" encryption, and simulation results show that the encrypted/decoded video is fully disguised.*

## 1. Introduction

The large processing time required by the DES and RSA encryption algorithms [1] hinders the secure transmission or storage and retrieval of MPEG-I [2,3] compressed bit-streams in real time. For example, the DES encryption or decryption of one second of an MPEG-I bit-stream (1.2 M bits) using a 500-MHz Pentium PC (which can encrypt roughly 650,000 bits per second) requires a processing time of 1.86 seconds. However, according to Shannon's theory of secrecy [4], perfectly compressed (zero redundancy) bit-streams are perfectly secure, and a compressed bit-stream is more secure than the corresponding uncompressed video. Hence, "total" encryption of a compressed bit-stream is not necessary, although a supplemental encryption can be used to improve the security level of the bit-stream. This supplemental encryption can be applied to selective portions of the bit-stream in order to combat the remaining redundancy in the bit-stream.

Spanos and Maples [5] reduce the encryption/decryption processing time of the video bit-stream by applying encryption to only the data associated with the intra-coded frames (I-frames) and suggested that the predicted frames (P and B frames) will inherit their required security from the security of the I-frames. Although this approach reduces the processing time by about 80-85%, Alattar and Al-Regib demonstrate [6] that the achieved security level of this scheme is insufficient. Furthermore, they suggest encrypting all I-macroblocks in all frames, in order to improve the security level. They report improvement in the security level while reducing the processing time over "total" encryption by 65-75%. However, they note that decoding the encrypted video sequences with an MPEG decoder might reveal the motion contents of the video. To overcome this problem, they encrypt the headers of the predicted macroblocks in addition to all intra-coded macroblocks (I-macroblocks) in all frames. This disguises the encrypted video completely but reduces the savings in processing time to 25-65% [6].

In this paper, we propose applying alternative encryption techniques to both the I-macroblocks and the headers of the predicted macroblocks in order to improve the performance of the methods proposed in [6]. For this purpose, three methods are proposed and empirically evaluated. The methods are discussed in Section 2, while the experimental procedure is explained in Section 3. Section 4 describes the simulation results and compare the new proposed methods with the methods presented in [6]. Several conclusions are presented in Section 5.

## 2. Alternative/Selective Encryption of MPEG video

Three new methods are proposed in this section to improve the performance of the method proposed by Alattar and Al-Regib [6] which calls for encrypting all I-macroblocks and the headers of all predicted macroblocks. Henceforth, this method will be referred to as Method 0. The improvement focuses on reducing the processing time while keeping a high security level, i.e., keeping the encrypted video completely disguised. In the first newly proposed method, the encryption is applied to the data associated with every $n^{th}$ I- macroblock. However, in the second method, the encryption is applied to the headers of



0-7803-5467-2/99/ $ 10.00 © 1999 IEEE

all the predicted macroblocks (non-intra-macroblocks in P- and B-frames) as well as to the data associated with every $n^{th}$ I-macroblock. In the third method, encryption is applied to every $n^{th}$ I- macroblock as well as the header of every $n^{th}$ predicted macroblock. The Data Encryption Standard (DES) is used as the encryption algorithm in the three methods.

### 2.1 Encrypting Intra-Coded Macroblocks Alternatively

In the first method, we propose to encrypt every other I-macroblock. Thus the data of every $n^{th}$ I-macroblock is encrypted using the DES, and the data of all other I-macroblocks is left intact. The value of $n$ is determined experimentally. Henceforth, this method will be referred to as Method I.

A counter is used to keep track of the encrypted macroblocks. This counter is updated at the beginning of each slice. Although this method leaves a certain number of I-macroblocks "unencrypted", these macroblocks are not expected to reveal information about the video. This is due to the differential-encoding scheme used to encode the DC-coefficients of the blocks [2,3]. Hence, an incorrect DC coefficient of one of the six blocks of a macroblock causes an intensity and color shift for the entire macroblock.

The processing time required by Method I, is less than that required when encrypting all I-macroblocks [6]. The actual savings depends on the value of $n$. The higher $n$ is, the higher the savings. For $n$=2, the required processing time is nearly half.

### 2.2 Encrypting the I-Macroblocks Alternatively and the Headers of All the Predicted Macroblocks

Although, Method I hides the intra information, the motion information is expected to be very clear when the encrypted video is played back. Therefore, to improve the security level of Method I, the headers of all the predicted macroblocks must be encrypted in addition to every $n^{th}$ I-macroblock. The DES algorithm is used to encrypt the header data. Henceforth, this method will be referred to as Method II.

Since the DES algorithm operates on 64 bits at a time, a 64-bit segment is encrypted starting from the header of each predicted macroblock. Depending on the size of the macroblock header, this segment may include the entire header, most of the header, or the entire header and some of the data in the macroblock. If the segment includes part of the header, then this causes a serious synchronization problem for any eavesdropper who tries to decode the bit-stream. This lends security to all the motion vectors although some of them may not be encrypted. If the segment includes part of the data of the macroblock, then this further increases the security of the transmitted video. Hence, Method II is expected to disguise the video completely.

The processing time required by Method II is less than that of Method 0 [6], which calls for encrypting all I-macroblocks and the headers of all the predicted macroblocks. As with Method I, the actual savings depend on the choice of $n$. However, since Method II additionally encrypts the headers of all predicted macroblocks, the savings in processing time is expected to be less than that of Method I.

### 2.3 Encrypting Alternatively Both the I-Macroblocks and the Headers of the Predicted Macroblocks

The third newly proposed method calls for encrypting every $n^{th}$ I-macroblock and the header of every $n^{th}$ predicted macroblock. This method requires much less processing time than Method 0. Henceforth, this method will be referred to as Method III.

Two counters are used in Method III- one to keep track of the encrypted I-macroblocks and the other to keep track of the encrypted headers of the predicted macroblocks. For synchronization purposes, both counters are reset at the beginning of each slice. As in Methods I and II, the DES algorithm is used to encrypt both the I-macroblock data and the header data. Since more data is encrypted using method III, the processing time required by this method is higher than those of the previous two methods. However, this processing time is roughly $\frac{1}{n}th$ of that required by Method 0 [6] which calls for encrypting all I-macroblocks and the headers of all predicted macroblocks.

## 3. Experiment Procedure

### 3.1 Interfacing the Encryption/Decryption System with an MPEG-I Encoder/Decoder

In order to interface the proposed encryption/decryption scheme with a standard MPEG-I codec, a bit-stream parser has been designed. This parser allows the encrypter to extract the required parts of the MPEG-I bit-stream and replace them with encrypted bits. Figure (1) depicts the interface between the bit-stream parser and the MPEG-I encoder and the encryption system. The stream parser is shown as the dotted box in the figure. A similar parser is used with the MPEG-I decoder to allow the decrypter to extract the encrypted parts from the bit-stream and replace them with the decrypted bits.

The internal structure of the bit-stream parser is shown within the dotted box in Figure (1). This parser operates on a full MPEG-I bit-stream. Such a bit-stream consists of multiplexed system, audio, and video streams [3]. In order to extract the video stream, the bit-stream parser first identifies the system header and then extracts its packs and

identifies the packets in each pack. All video packets that belong to the same picture are assembled and processed further by the video parser in order to extract the required information for encryption purposes. This information is then passed to the encryption sub-system. After the encryption sub-system encrypts the required portion of the bit-stream, the video parser replaces the original data in the video stream with the encrypted data. The bit-stream parser then re-packetizes the video stream and re-multiplexes it with the system and audio information and transmits it or stores it. A circular buffer is implemented to write the processed bits back to where they belong in the bit-stream.

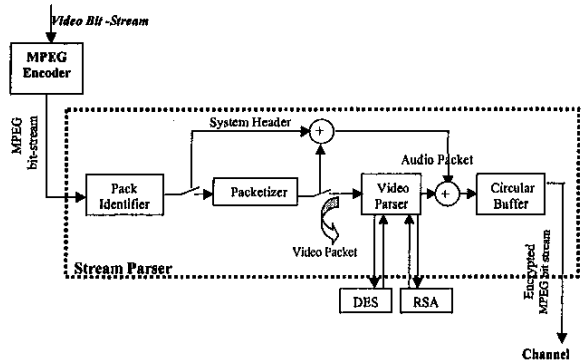


Fig. 1: A secure MPEG-I compressed video encoder with the structure of the MPEG-I bit-stream parser

### 3.2 Distribution of the DES Key

To protect the system from a "brute force" attack, the DES key is changed periodically and transmitted to the receiver in an encrypted form. If a cryptanalyst gets one of these keys and uses it to attack the system, he will not be able to decrypt the entire video. He will be able to decrypt only the frames encrypted with that key. The DES key is transmitted within the MPEG-I video bit stream in the USER_DATA section in the picture header. This location does not disturb the bit stream integrity, and the resulting bit-stream still conforms to the MPEG-I standard. Furthermore, this does not cause any delay or storage requirement at the receiver, since the encrypted data arrives at the receiver after the DES key. However, this introduces a slight transmission overhead. As in [6], it is recommended to change the DES key for every Group-of-Pictures (GOP). In this case, the resultant overhead is only 0.01%.

It is also advantageous to fortify the system security by encrypting the DES key using a public-key algorithm [1]. This has the advantage that the cryptanalyst can not apply the same method to attack the two encryption keys. In our implementation, the RSA algorithm is used to encrypt the DES key before embedding it within the video bit-stream.

The RSA standard is not used directly to encrypt/decrypt the data in the MPEG-I bit stream since it requires a long processing time [1]. This processing time is directly proportional to the size of the data to be encrypted. In this paper, we propose a 56-bit DES key; hence, the required processing time to encrypt this key using the RSA algorithm is negligible when compared to that required for encrypting an average MPEG-I frame of size 40,000 bits.

## 4. SIMULATION RESULTS

The three proposed encryption methods have been applied to several MPEG-I video clips with different contents and motion characteristics. The encrypted bit-stream has been decompressed using the VMPEG 1.7d decoder (by Stefan Eckart), and the decoded sequence has been visually evaluated. The value of $n$ has been empirically chosen to be 2. Thus, every other macroblock is encrypted. Using Method I, the motion in the video sequence can be easily identified when the video sequence is played back, though it is difficult to recognize the moving objects. The results are similar to those reported in [6] when all I-macroblocks in all frames were encrypted. Hence, Method I does not provide an adequate level of security, especially for highly sensitive videos, such as in military applications.

Figure (2.a) shows a frame from the *Paratroopers* video clip in which some paratroopers are jumping from an airplane. Figures (2.b and 2.d) show this frame encrypted using the method of encrypting all I-macroblocks and Method I, respectively. The results with other video sequences are similar.

In the second experiment, Method II is used to encrypt the video sequences. As in the previous experiment, $n$ is set to 2. Similarly, in the third experiment, Method III is applied to the video sequences and $n$ is set to 2. Decoding the video sequences encrypted using Methods II and III, with the VMPEG 1.7d decoder, does not reveal any motion information, and they are completely disguised. These results are very similar to those reported in [2], where all the I-macroblocks and the headers of all the predicted macroblocks are encrypted (Method 0). Figures (2.c, 2.e and 2.f) show the frame of Figure (2.a) after encrypting and decoding for Methods 0, II and III, respectively. It is clear that the frames do not reveal any information on the video sequence contents. Furthermore, playing back the video does not reveal any motion information. Experimental results for other videos lead to the same conclusions. Hence, it is sufficient to encrypt every other I-macroblock and the header of every other predicted macroblock to disguise the video completely.

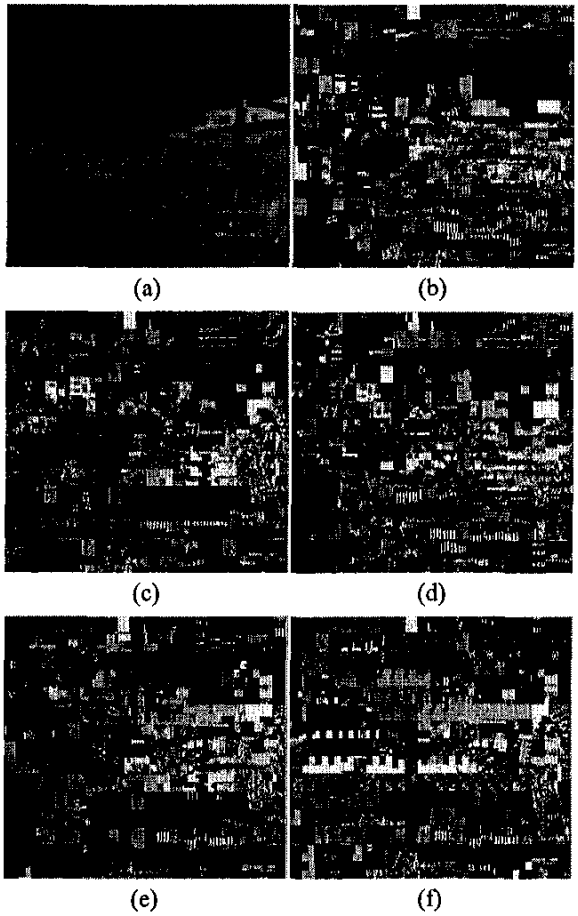

Fig. 2: (a) The original picture, (b) Decoded picture after encrypting all intra-macroblocks, (c) Decoded picture after applying Method-0, (d) Decoded picture after applying Method-I, (e) Decoded picture after applying Method-II, (f) Decoded picture after applying Method-III.

In addition, results indicate that Method I saves 82-88% of the processing time (compared to encrypting the whole video bit-stream), while Method II saves roughly 37-77%. Furthermore, Method III saves roughly 60-82% of the processing time. The latter two methods (II and III) give completely disguised MPEG video sequences similar to those obtained from Method 0. However, Method 0 saves only 21– 64%. These results are tabulated in Table (I). From these results, the savings in the processing time achieved by Method III as compared to Method 0 are obvious. Furthermore, no information can be perceived by playing back the encrypted video sequences resulting from Methods 0 and III.

The above simulation has been repeated for $n=3$, i.e., encrypting one of every three I-macroblocks and the headers of one of every three predicted macroblocks. The results show a clear degradation in the security level compared to the case where $n=2$, especially for the motion which can be easily recognized by playing back the encrypted MPEG video sequences. Hence, $n$ is recommended to be *2*. Choosing $n$ to be *one* turns Method III to be exactly Method 0 [6].

TABLE I. The reduction in the processing time achieved by the four methods for various video clips

| | **The Percentage of the Reduced Processing Time** | | | |
|---|---|---|---|---|
| **Sequence Name** | **Method 0** | **Method I** | **Method II** | **Method III** |
| *Fibon* | 24% | 84% | 55% | 62% |
| *Paratroopers* | 61% | 21% | 84% | 37% |
| *Creamedgates* | 62% | 24% | 85% | 38% |
| *Gulf_War* | 79% | 58% | 88% | 70% |
| *Kangaroo* | 80% | 60% | 85% | 74% |
| *Fish* | 82% | 64% | 87% | 77% |
| *Oj_Chase* | 71% | 41% | 88% | 54% |
| *Falcon* | 78% | 57% | 83% | 74% |

TABLE II. The inefficiency in Methods-0, II, and III and introduced by encrypting part of the data of the predicted macroblocks.

| | **The Inefficiency (%)** | | |
|---|---|---|---|
| **Sequence Name** | **Method 0** | **Method II** | **Method III** |
| *Fibon* | 37.56 | 47.76 | 37.58 |
| *Paratroopers* | 39.13 | 44.59 | 39.24 |
| *Creamedgates* | 38.31 | 49.85 | 42.62 |
| *Gulf_War* | 28.62 | 40.15 | 28.61 |
| *Kangaroo* | 16.62 | 26.21 | 16.59 |
| *Fish* | 19.11 | 29.67 | 19.04 |
| *Oj_Chase* | 36.47 | 45.86 | 12.02 |
| *Falcon* | 13.07 | 20.25 | 36.46 |

Finally, encrypting the headers of the predicted macroblocks introduces inefficiency. This inefficiency is due to the difference between the goal and the practical implementation. The goal is to encrypt only the headers of predicted macroblocks in order to disguise the motion information. However, since the DES uses 64 bits as an

input block, the 64-bit segment might include part of the macroblock data. Encrypting this data introduces inefficiency. On the other hand, these coefficients have most of the information in the block and hence, encrypting them improves the security level.

The inefficiency is calculated for each of the test video bit-streams, and is listed in Table (II) for Methods 0, II and III. The inefficiency is calculated by comparing the number of encrypted bits contained in the data part of the predicted macroblocks to the total number of encrypted bits. Accordingly, Methods 0 and III give nearly the same inefficiency since the compared quantities are reduced by nearly half in both methods. However, Method II produces a higher inefficiency since the first quantity is unchanged while the total number of bits being encrypted is reduced by an amount equivalent to half the number of bits in the I-macroblocks.

## 5. Conclusions

Three new alternative/selective encryption methods have been proposed and tested. These methods show that encrypting every other I-macroblock requires about half the processing time of encrypting all I-macroblocks. However, visual evaluation of the decoded video encrypted by this method reveals a lot of motion information. Also, it has been shown that encrypting both the I-macroblocks and the headers of the predicted macroblocks "alternatively" saves roughly 60-82% of the processing time, while no motion information is revealed from the encrypted-decoded MPEG video sequence. These are significant savings when compared to the 21– 64% savings achieved by applying the same method, but without alternative encryption [6]. Extending these methods to MPEG-II, MPEG-IV, H.261, and H.263+ is straightforward.

## Acknowledgement

The authors would like to thank Ms. Marcie L. Lucas for her help in preparing the manuscript.

## References

[1] B. Schneier, *Applied Cryptography, Second Edition, Protocols, Algorithms and Source Codes in C*, John Wiley & Sons, Inc., 1996.

[2] ISO/IEC 11172 Information Technology: Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1.5 Mbits/second, Part 1: System; Part 2: Video; Part 3: Audio, 1993.

[3] T. Sikora, "MPEG Digital Video Coding Standard," *IEEE Signal Processing Magazine*, September 1997.

[4] C. E. Shannon, "Communication Theory of Secrecy Systems", *Bell Systems Technical Journal,* July 1948, p.379, Oct. 1948, p.623.

[5] G. A. Spanos and T. B. Maples, "Security for Real-Time MPEG Compressed Video in Distributed Multimedia Applications," *Proceedings of the IEEE 15th Annual International Conference on Computers and Communications*, 1996.

[6] A. M. Alattar and G. I. Al-Regib, "Evaluation of Selective Encryption Techniques for Secure Transmission of MPEG Video Bit-Streams," *Proceedings of the 1999 IEEE Symposium on Circuits and Systems*, Orlando, Florida.