EVALUATION OF SELECTIVE ENCRYPTION TECHNIQUES FOR SECURE TRANSMISSION OF MPEG-COMPRESSED 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
Electrical Engineering Department
King Fahd University of Petroleum and Minerals
Dhahran 31261, Saudi Arabia
e-mail: gregib@zajil.net

ABSTRACT

"Total" encryption of MPEG-I compressed bit-streams requires a large processing time. To reduce this processing time, Spanos and Maples [1] propose encrypting only the data in the Intra-coded frames. In this paper, experimental results are presented which indicate that Spanos and Maples' method does not provide an adequate level of security. A better method that calls for encrypting all the Intra-coded macroblocks (I-MB's) in all frames is then presented. Although this method achieves a 69% reduction in the processing time over "total" encryption, the motion content of the video sequence is apparent. Therefore, another method that calls for encrypting the headers of all predicted macroblocks (MBs) in addition to encrypting the data in all I-MB's is presented. Simulation results for this method show that the decoded video is fully disguised, but at the expense of decreasing the savings in the processing time to 21-60% of the processing time of "total" encryption.

1. INTRODUCTION

Secure transmission of digital video has long been a top priority for military applications, and is an increasingly important issue for commercial TV broadcast and internet-based applications such as video-on-demand.

Digital video is usually compressed using an MPEG-I or -II algorithm in order to save on the transmission bandwidth and storage requirements [3]. If secure transmission or storage is required, the compressed bit-stream can be treated as data and encrypted with the Data Encryption Standard (DES) or the RSA encryption algorithm. However, since compressed video bit-streams are typically huge and the encryption/decryption algorithms are relatively slow, the encryption/decryption process results in a tremendous processing time. For example, the DES encryption or decryption of one second of an MPEG-I bit-stream (1.2 M bits) using a 266-MHz Pentium PC (which can encrypt 344,000 bits per second) requires a processing time of 3.49 seconds [4]. Hence, encryption and decryption hinder real-time transmission or storage and retrieval of digital video bit-streams.

Spanos and Maples [1] attempt to reduce the encryption/decryption processing time by applying encryption to only the data associated with the I-frames. They suggest that the P- and B- frames inherit their required security from the security of the I-frames. Although this approach reduces the encryption processing time by about 80-85%, it is shown in this paper that the achieved security level of this scheme is not adequate. Agi and Gong [2] suggest increasing the frequency of I-frames in the video sequence in order to improve the achieved security level. Unfortunately, this reduces the efficiency of the compression algorithm and in turn reduces the achieved savings on the processing time.

In this paper, we first present some experimental results in order to show that the selective-encryption method proposed by Spanos and Maples [1] does not provide an adequate level of security. Then we propose two new methods that provide higher security levels at relatively low processing times.

2. SELECTIVE ENCRYPTION OF MPEG-COMPRESSED BIT-STREAMS

Encrypting the data in the Intra-coded frames (I-frames) as suggested by Spanos and Maples [1] is not adequate for high-security applications. This is because the P- and B- frames contain many Intra-coded macroblocks (I-MBs). As transmission time progresses, these macroblocks and the use of motion compensation cause replenishment of most of the image. The replenished areas reveal the motion content of the video when the encrypted video is decoded with a standard MPEG-I decoder. The amount of the revealed information is directly proportional to the size of the replenished areas, which in turn is directly proportional to the number of I-MBs. Typically, about 25% of the macroblocks in a P or B frame are encoded as I-MBs. This percentage, however, increases to about 100% when a scene change is encountered. In this case, if the encoder does not encode this frame as an I-frame, most of the data in this frame is not encrypted. In order for a decoder to enforce an I-frame at the scene change, the encoder must first employ a sophisticated scene change detector.

Two new methods are proposed in this section with the goal of improving on the performance of the method proposed by Spanos and Maples. In the first method, the encryption is applied to the data associated with all I-MBs (in all I-, P-, and B-frames). In the second method, the encryption is applied to the headers of all the predicted MBs (non-I-MBs in P- and B-frames) as well as to the data associated with all the I-MBs. The DES is used in both methods.

In this paper, Spanos and Maples' method is referred to as Method-I. Our proposed method of encrypting all the I-MBs is referred to as Method-II, and our proposed method of encrypting the headers of the predicted MBs in addition to all the I-MBs is referred to as Method-III.

2.1 Encrypting All I-MBs

The data from all the MBs in each I-frame is encrypted together. It is first divided into 64-bit segments; then the DES

0-7803-5471-0/99/$10.00©1999 IEEE

algorithm is applied to each segment. If the last segment is less than 64 bits, it is not encrypted.

The I-MBs in the P- and B-frames are encrypted differently than those in the I-frames in order to allow the receiver to identify the encrypted macroblocks. In this case, each macroblock is encrypted separately, and encryption starts immediately after the Macroblock_Type parameter in the header of the I-MB. This parameter is left unencrypted. The rest of the data in each I-MB is divided into 64-bit segments and the DES algorithm is applied to each segment independently. If the size of the last segment is less than 64 bits, it is left unencrypted. In order for the decryptor to recognize the unencrypted segment (if there is any), it attempts to decode the Huffman codes before decrypting. If it can successfully decode the DCT coefficients, then the segment is the last segment and it is considered unencrypted; otherwise it continues decrypting the data.

2.2 Encrypting the Headers of the Predicted MBs

The headers of the predicted MBs can be encrypted in two ways. The first method is similar to that used for encrypting the data in the I-MBs in a P- or B-frame. However, since the size of each of these headers ranges between 27 and 121 bits, only one segment of 64-bits is encrypted. If the size of the header is more than 64 bits, then the last part (after the first 64 bits) of the header is excluded. However, if the size of the header is less than 64-bits, the first part of the macroblock data is included with the header to form a 64-bit segment. This data is encrypted along with the header. The inefficiency caused by including this data is discussed in Section 3.3. Excluding part of the header or including part of the macroblock data in the encrypted 64-bit header segment causes a synchronization problem if an eavesdropper decides to skip over the encrypted header and attempts to decode the DCT coefficients directly.

The second method of encrypting the headers of the predicted MBs is more efficient, but requires the transmission of additional information to help in decoding the bit-stream. In this method, the headers of all macroblocks in a slice are grouped together into a sub-bit-stream called the header sub-bit-stream, and the data of all macroblocks is grouped into another sub-bit-stream called the data sub-bit-stream. Then, the header sub-bit-stream is divided into segments of 64-bits each (the last segment may be less than 64 bits). Each of these 64-bit segments is then passed to the DES algorithm for encryption. If the length of the last segment is less than 64 bits, it is left unencrypted. Finally, the data sub-bit-stream is concatenated with the encrypted header sub-bit-stream and transmitted.

This method produces a bit-stream that does not conform to the MPEG-I standard. To be able to decode this bit-stream, the length of the header sub-bit-stream in each slice must be transmitted to the receiver as overhead information. This information can be encoded in the user data section of the slice. In this case, decoding of each slice alternates between decoding the header sub-bit-stream and the data sub-bit-stream. It starts by decoding the header of the first MB from the header sub-bit-stream then switches to decoding the data of the MB from the data sub-bit-stream. The same process is repeated for all other MB's in a slice.

On the other hand, the bit-stream resulting from the first method conforms to the MPEG-I standard; hence it does not require the transmission of any overhead information or restructuring at the decoder. Also, this method better disguises the MPEG-I bit-stream, since the upper-left DCT-coefficients of the first block in each P- or B-macroblock are also disguised. These coefficients are the DC and the low-order DCT coefficients, which carry most of the information in a block. For these reasons this method is preferred over the first choice and it is implemented in this paper.

2.3 Distribution of the DES Key

To guard against brute force attack, the DES key is changed periodically and transmitted to the receiver in an encrypted form. If a crypt-analyst manages to get one of these keys and to use it to attack the video, he will not be able to compromise the entire video. He will be able to decrypt only the frames encrypted with that key. The 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. Also, this does not also 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. This overhead is 88 bits (32 bits for the USER_DATA_START_CODE and the remaining 56 bits for the DES key) every time the DES key is transmitted. This overhead is tolerable if the key is not changed too frequently. One extreme case is to change the key for every frame. In this case, the overhead is 88 bits per frame, which is about (0.16%) of the 50,000-bit average size of an MPEG-I frame. Another extreme case is to use one key for the whole MPEG-I video sequence. The overhead in this case is only 88 bits for the entire video bit-stream. A good compromise is to change the DES key for every Group-of-Pictures (GOP). In this case, the overhead is only 0.01%.

The DES key is encrypted before it is embedded in the bit-stream. A public-key crypto-system such as the RSA standard algorithm is used for this purpose. This means that two keys (one is public and the other is private) are used in the encryption/decryption process. This has the advantage that the crypt-analyst can not apply the same method to attack the two keys. Private key (symmetric) and public-key (asymmetric) algorithms are normally attacked differently. This enhances the security level considerably [4]. In our implementation, the transmitter uses the RSA public-key of the intended recipient to encrypt the DES key, and the intended recipient uses his private RSA key to decrypt the DES key. Finally, he uses the DES key to decrypt the rest of the bit-stream.

The RSA standard is not used directly to encrypt/decrypt the data in the MPEG-I bit stream, because it requires a long processing time [4]. 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.

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

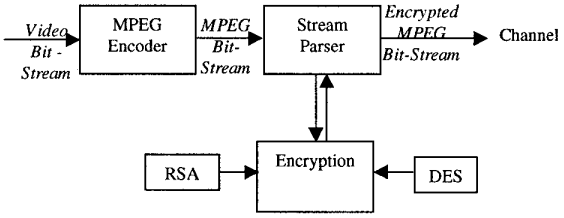

Fig. 1: A Secure MPEG-I Compressed Video Encoder

In order to interface the proposed encryption/decryption scheme with a standard MPEG-I codec, a bit-stream parser is 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. 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.

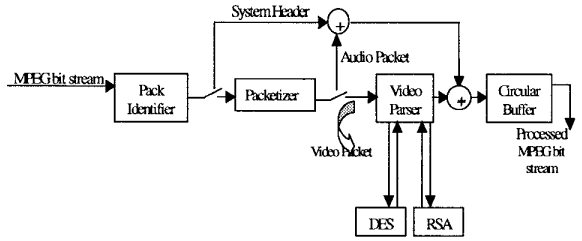

Fig. 2: The Structure of the MPEG-I Bit-Stream Parser

The internal structure of the bit-stream parser is shown in Figure (2). This parser operates on a full MPEG-I bit-stream. Such a bit-stream consists of multiplexed system, audio, and video streams. 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

3. SIMULATION RESULTS

The three selective encryption methods described in Section 2 were implemented and tested with several MPEG-I bit-streams representing video sequences of various characteristics and motion content. These bit-streams are listed in Tables (I) and (II).

In the first experiment, only the I-frames of each bit-stream were encrypted as proposed by Spanos and Maples [1] (Method-I). Then, the encrypted bit-stream was decompressed using the VMPEG-I 1.7d-Lite (by Stefan Eckart) decoder, and the decoded sequence was visually evaluated. Figure (3.a) shows an example of a decoded image from the "Paratroopers" bit-stream. This frame reveals many features of the original video. For example, the wings of the airplane and the shoulders and feet of the paratroopers can be easily identified. As explained before, this is due to encoding many of the macroblocks in the P- and B-frames as Intra macroblocks when the motion estimation algorithm of the encoder fails to predict the macroblock from the reference frame(s). This usually happens at the areas of the image where the motion is either rotation or deformation.

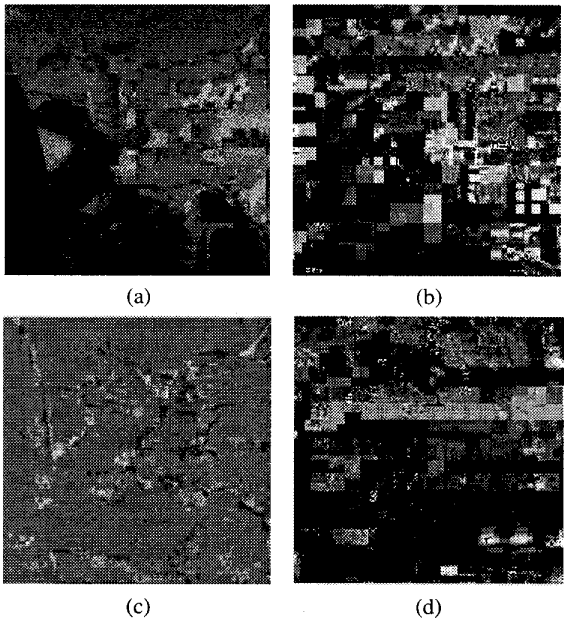

Fig. 3: (a) Decoded picture after encrypting intra-frames only
(b) Decoded picture after encrypting all intra-macroblocks
(c) Decoded picture after zeroing all intra-macroblocks
(d) Decoded picture after encrypting all Intra macroblocks and the headers of predicted macroblocks

Motion compensation fails at the first frame of a scene change, which caused some of the decoded frames in the test sequences to appear completely unencrypted. As expected, each of these frames was found to be a P- or B-frame at the beginning of a new scene. Moreover, playing each of the decoded video clips clearly showed the motion of the objects in that video sequence, though without revealing the objects' identities. For example, it was difficult to identify that the moving objects in the "Paratroopers" bit-stream were paratroopers jumping from an airplane. However, motion content is very important information, and it must be disguised in highly sensitive applications such as those of the military. We can therefore, conclude that the approach of Spanos and Maples [1] does not provide an adequate level of security for these applications.

In the second experiment, the same video sequences were encrypted using Method-II, which calls for encrypting all I-

MBs in all frames. The encrypted bit-streams were then decoded using a VMPEG1.7d-Lite decoder, and the result for the "Paratroopers" video sequence is shown in Figure (3.b). This figure shows that the image is completely scrambled at the macroblock level. When the standard MPEG-I decoder was forced to skip over the encrypted part by replacing it with zeros, the image in figure (3.c) was obtained. This image reveals more features of the video sequence than the image shown in Figure (3.b). In Figure (3.c), the airplane and the paratroopers can be clearly identified. However, details such as those of the plane can not be identified. Contrary to the results obtained from Method-I, the first frame at the beginning of each scene was completely disguised. Although, playing the decoded video clip still clearly revealed the motion of the objects in the video, the overall results clearly indicate that this method achieves a higher security level than Method-I.

In the third experiment, the same video sequences were encrypted using Method-III, which calls for encrypting the headers of all the predicted MBs in addition to the data in all Intra MBs. The encrypted bit-stream was then decoded with a VMPEG 1.7d-Lite decoder. Figure (4.d) shows a decoded frame, which does not reveal any of the features of the original video. Moreover, playing the resultant video does not reveal any motion information. This enhancement in the security level was at the expense of increasing the processing time.

TABLE I. The Reduction in the Processing Time Achieved by the Three Methods for Various Video Clips

	The Percentage of the Reduced Processing Time		
Sequence Name	**Method-I**	**Method-II**	**Method-III**
"Fibon"	78.0	67.4	23.6
"Paratroopers"	83.8	68.5	21.4
"Creamedgates"	80.5	69.4	23.9
"Gulf_War"	80.9	75.8	58.1
"Kangaroo"	83.1	70.4	59.7
"Fish"	86.9	74.7	64.4
"Oj_Chase"	81.4	76.1	41.7
"Falcon"	80.2	65.1	57.0

Table (I) tabulates the savings in the processing time achieved for the various test bit-streams under each of the three encryption methods. This table indicates that encryption Method-I saves 78-87% of the processing time, while encryption Method-II saves 65-76%, and encryption Method-III saves 21-60% of the processing time. The table also shows that the reduction in the processing time of Method-III highly depends on the nature of the video. The first three video sequences have simple and highly predictable motion compared with the other video clips. This increases the number of predicted MBs and decreases the bits associated with each of these predicted MBs. Since 64 bits are encrypted for each of the headers in these predicted macroblocks, the percentage of bits encrypted for the first three videos is higher than that for the other sequences. This also can be explained by the inefficiency of Method-III.

As explained before, encrypting the headers of the predicted MBs in Method-III introduces inefficiency. This inefficiency was calculated for each of the test video bit-streams, and is listed in Table (II). This inefficiency is due to the use of a fixed length input block of 64 bits with the DES algorithm, which may unnecessarily include some low frequency DCT coefficients. However, since these coefficients have most of the information in the block, encrypting them improves the security level.

TABLE II. The Inefficiency in Method-III Introduced by Encrypting Part of the Data of the Predicted MB.

Sequence Name	**The Inefficiency (%)**
"Fibon"	37.56
"Paratroopers"	39.13
"Creamedgates"	38.31
"Gulf_War"	28.62
"Kangaroo"	16.62
"Fish"	19.11
"Oj_Chase"	36.47
"Falcon"	13.07

5. CONCLUSIONS

It has been empirically shown that encrypting only the I-frames in an MPEG-I bit-stream does not provide an adequate security level for sensitive applications. It has also been shown that encrypting the data associated with all I-MBs in all frames has a better security level, but it does not conceal the motion in the video sequence and it has a higher processing time. Finally, it has been shown that encrypting the headers of all predicted MBs as well as encrypting the data associated with all I-MBs has the best security level. However, this technique achieves less savings in the processing time than the other two methods. In this case, the savings is 21-64%, depending on the nature of the bit-stream. Analysis of brute-force attack by an eavesdropper on the three selective encryption methods is currently under investigation. Extending these methods to MPEG-II, MPEG-4, H.261, and H.263+ is straightforward.

ACKNOWLEDGEMENT

The authors would like to thank Ms. Marcie L. Lucas for editing the manuscript.

REFERENCES

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

[2] Iskander Agi and Li Gong, "An Empirical Study of Secure MPEG-I Video Transmissions," *Proceedings of SNDSS*, 1996.

[3] ISO/IEC 11172 Information Technology: Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1.5 Mbits/second, Part1: System; Part2: Video; Part3: Audio, 1993.

[4] Bruce Schneier, *Applied Cryptography, Second Edition, Protocols, Algorithms and Source Codes in C*, John Willey & Sons, Inc., 1996.