On the Security of an MPEG-Video Encryption Scheme Based on Secret Huffman Tables

Shujun Li[1,⋆], Guanrong Chen[2], Albert Cheung[3], Kwok-Tung Lo[4], and Mohan Kankanhalli[5]

[1] Fachbereich Informatik und Informationswissenschaft, Universität Konstanz, Fach M697, Universitätsstraße 10, 78457 Konstanz, Germany
[2] Department of Electronic Engineering, City University of Hong Kong, Kowloon, Hong Kong, China
[3] Department of Building and Construction and Shenzhen Applied R&D Centres, City University of Hong Kong, Kowloon, Hong Kong SAR, China
[4] Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong SAR, China
[5] School of Computing, National University of Singapore, 117590 Singapore
Shujun.Li@uni-konstanz.de, {EEGCHEN,clacc}@cityu.edu.hk, enktlo@polyu.edu.hk, mohan@comp.nus.edu.sg

Abstract. This paper re-studies the security of an MPEG-video encryption scheme based on secret Huffman tables. The present cryptanalysis shows that: 1) the key space of the encryption scheme is not sufficiently large against divide-and-conquer (DAC) ciphertext-only attack; 2) its security against the chosen-plaintext attack is very weak. The insecurity is mainly due to the separated use of different Huffman tables for different sets of syntax elements. A brief discussion on how to improve this MPEG-video encryption scheme is also given.

1 Introduction

The extensive use of digital images and videos in today's digital world makes the security and privacy issues become more important. To fulfill such an increasing demand, various encryption algorithms have been proposed in recent years as possible solutions to content protection of digital images and videos [1, 2, 3, 4, 5, 6, 7, 8, 9, 10], among which MPEG videos attract special attention due to its prominent prevalence in consumer electronic markets [11, 12, 13]. As an important way of designing MPEG-video encryption schemes, secret Huffman tables have been suggested in some designs [2, 7, 14, 15, 16, 17, 18].

The MPEG-video encryption scheme proposed in [14] (i.e., Algorithm 1 in [2]) is a light-weight scheme, which encrypts the plain-video by shuffling VLC (variable-length coding) entries of same size in each Huffman table. However, because the bit length of each VLC codeword does not change, the position of each VLC codeword in the video stream does not change either. Thus, an attacker can uniquely locate (and thus determine) all VLC codewords contained

⋆ The corresponding author, personal web site: http://www.hooklee.com

T. Wada, F. Huang, and S. Lin (Eds.): PSIVT 2009, LNCS 5414, pp. 898–909, 2009.

in the cipher-video stream, if the plain-video stream or an independent part (such as a picture or a slice) is known. Once all distinct VLC codewords are obtained, the whole secret Huffman table is uniquely reconstructed and the encryption scheme is broken. That is, this light-weight scheme is not secure against known/chosen-plaintext attacks. In addition, as pointed out in [2], the key space of this encryption scheme is very limited (especially for Huffman tables with a small number of VLC entries), so even a brute-force attack may be feasible.

The MPEG-video encryption scheme proposed in [15] can be considered as an enhanced version of that in [14]. In this scheme, five different Huffman tables are shuffled separatedly and the shuffling operations are generalized to work on VLC entries with different sizes, in the hope that the key space can be enlarged and the security against plaintext attacks can be improved. Furthermore, as a second guard on the security, random bit flipping operations are also introduced to further encrypt each secret Huffman table.

In [7, 16, 17, 18], multiple Huffman tables (MHT) are introduced, from which one table is secretly chosen for the encryption of each VLC codeword. A so-called "Huffman tree mutation process" is also proposed in [7, 16, 17] to derive more candidate Huffman tables from several original tables. Some cryptanalysis results about known-plaintext attacks have been reported recently [19, 20].

This paper mainly focuses on some security problems with the MPEG-video encryption scheme proposed in [15]. Our cryptanalysis shows that this scheme is not sufficiently secure against DAC (divide-and-conquer) ciphertext-only attack and very weak against chosen-plaintext attack.

The rest of this paper is organized as follows. In the next section, a brief introduction to the MPEG-video encryption scheme under study is given. Then, the cryptanalysis results are presented in detail in Sec. 3. Finally, Section 4 gives a brief discussion on how to improve the security of the MPEG-video encryption scheme, and the last section concludes this paper.

2 MPEG-Video Encryption Scheme Under Study

In MPEG-1/2 standards, Huffman coding is used to realize lossless compression of quantized DCT coefficients. Each Huffman tree is represented as a 1-D Huffman table, which transforms an input value into a VLC codeword. There are in total 15 Huffman tables used in MPEG-2 standard [12] (less in MPEG-1 standard [11]), among which 10 ones (Tables B-1 to B-9) are used for coding syntax elements in various headers and the following six ones are for visual information – DCT coefficients in each block and motion vectors in each macroblock:

- Table B-10: for encoding motion vectors;
- Table B-11 (not used in MPEG-1 standard): for encoding the differential motion vectors of the dual prime prediction;

- Table B-12: for encoding the bit size of the differential values of DC coefficients in intra luminance blocks;
- Table B-13: for encoding the bit size of the differential values of DC coefficients in intra chrominance blocks;
- Table B-14: for encoding all DCT coefficients of non-intra blocks and AC coefficients of intra blocks with $intra\_vlc\_format = 0$, where $intra\_vlc\_format$ is a picture-specific flag defined in MPEG-2 standard (which does not exist for MPEG-1 videos and the value shall be taken as 0);
- Table B-15 (not used in MPEG-1 standard): for encoding all AC coefficients of intra blocks with $intra\_vlc\_format = 1$.

The encryption scheme proposed in [15] is designed by concealing the original Huffman tables, i.e., using different (secret) Huffman tables to replace the original ones. Five Huffman tables used for coding visual information, B-10, B-12, B-13, B-14 and B-15, are chosen to be kept secret. Table B-11 is not selected, since it is only a very small Huffman table with three entries. The five secret Huffman tables are derived from the original ones by performing the following two encryption operations.

- *Shuffling VLC codewords:* grouping all VLC codewords into several subsets according to their bit sizes, and then randomly shuffling these VLC codewords within each sub-set.
- *Random bit flipping:* randomly flipping the last bit of each VLC codeword, and adjusting (if needed) other VLC codewords to keep the prefix rule valid.

After encrypting all the five Huffman tables, the bit sizes of some (at least one) VLC codewords should be slightly changed to resist known-plaintext attacks (as discussed in Sec. 1 of this paper), but the change should not be too much to avoid a large influence on compression efficiency.

In [15], the key space was estimated by enumerating all "good" encryption methods[1] of shuffling and random bit flipping operations carried out on some selected significant (not all) VLC codewords, as shown in Table 2. As a result of the large key space, the scheme was considered sufficiently secure.

In [15], an additional measure is suggested to further enhance the security against plaintext attacks – reshuffling the Huffman tables after a certain number

Table 1. Number of good encryption methods of each Huffman table (Table 3.6 of [15])

Huffman table	Number of good encryption methods
B-10	$3!$
B-12	$7! \times 2^6$
B-13	$6! \times 2^8$
B-14	$6!$
B-15	$16!$
Total	$(3!) \times (7! \times 2^6) \times (6! \times 2^8) \times (6!) \times (16!) \approx 2^{92}$

[1] An encryption method is "good" if it can produce unintelligible images.

of frames. In the following, we will mainly consider the basic scheme without the reshuffling mechanism. The effect of the reshuffling mechanism will be discussed later in Sec. 4.

3 Cryptanalysis

In this section, the security of the aforementioned MPEG-video encryption scheme based on secret Huffman tables is reconsidered, and it is found that the scheme is not so secure as evaluated in [15]. In this section, the terms and notations in MPEG-2 standard [12] will be used, except those only existing in MPEG-1 videos. The following terms are used throughout this section to facilitate the description: 1) the term "picture" is used instead of "frame", since the encryption scheme is independent of the syntactic differences between a picture and a frame; 2) macroblock is abbreviated as "MB"; 3) the term "MB header" is used to denote the set of all syntax elements occurring before the first encoded block in an MB (if none of the blocks is coded, the MB header is the MB itself).

3.1 Ciphertext-Only Attack

The ciphertext-only attack is the most common attack in practice, since in general the communication channels are open to the public, which means that an attacker can observe as many ciphertexts as possible and then use them to break an encryption scheme [21]. There are two different goals in a ciphertext-only attack: recovering the plaintexts and recovering the secret key. This paper mainly focuses on the recovery of the secret key, i.e., the secret Huffman tables in the MPEG-video encryption scheme under study.

The simplest ciphertext-only attack is to exhaustively search all possible keys to find the unique correct one (or an equivalent key), which is called brute-force attack [21]. Here, a criterion is needed to verify each searched key. For the MPEG-video encryption scheme under study, the occurrence of syntax errors can serve as such a criterion for detecting wrong keys. When a wrong Huffman table is used to decode a cipher-picture, syntax errors may occur in the decoding procedure due to (but not limited to) the following reasons.

- As mentioned in Sec. 1, to ensure the security against plaintext attacks, there should be at least two distinct bit sizes for each input value in a Huffman table. However, once the bit size of a VLC codeword is wrong, all the following syntax elements in the current slice cannot be correctly located and decoded.
- For each Huffman table, not all FLC codewords of a specific bit size are valid VLC codewords.
- All stuffing bits at the end of a slice should be zero bits.
- There may exist some marker bits (must be "1" to avoid "start code emulation", i.e., the occurrence of fake start codes) in the bit stream:
  - (for MPEG-2 videos only) when $concealment\_motion\_vectors = 1$ in an intra-block, there exists a marker bit in the MB header;

  - (for MPEG-1 videos only) in D-picture, the last bit of each MB must be a marker bit named *end_of_macroblock*.
- There exist some constraints on the decoded syntax elements:
  - each decoded DCT coefficient should not be out of the range [−2048, +2047];
  - each decoded motion vector should not be out of the range defined in Table 7-8 of the MPEG-2 standard [12], and must be within the reference picture after adding the coordinates of the predicted MB.
- There must be an EOB VLC codeword at the end of each block, before which the total number of decoded DCT coefficients must not be greater than 64.
- The number of MBs within each picture should not be greater than a maximal value.
- Some slice headers may be skipped when the video is decoded with wrong Huffman tables, which is forbidden for most videos (for example, an MPEG-1/2 video stream with a restricted slice structure).

By detecting syntax errors occurring in the decoding procedure, one can distinguish most wrong Huffman tables. In addition, there exist a lot of information redundancies in decoded video. Therefore, even when no syntax error is detected, one can still distinguish a wrong Huffman table if there exist too many abrupt changes around the borders of adjacent blocks. Finally, note that if no syntax error is found for a wrong key, then this wrong key tends to be an equivalent key to decrypt the cipher-video (or some part of the cipher-video), which happens when some VLC codewords are not involved in the encoding process of the plain-video or part of it.

Because the five Huffman tables are used for different sets of syntax elements of the whole video bit-stream, it is possible to separatedly guess them one by one. This means that one can use the so-called divide-and-conquer (DAC) attack [21] to break the MPEG-video encryption scheme. In other words, the key space of the encryption scheme will be the ***sum***, not the ***product***, of the sub-key-spaces of the five tables. Next, let us see how to separatedly break the five Huffman tables by detecting syntax errors in the video decoding procedure.

Reconstructing Table B-10. Following the MPEG-2 standard, Tables B-12/13/14/15 are all independent of the decoding of the first MB header in a slice[2], which makes the separated reconstruction of Table B-10 possible. When a wrong Table B-10 is used, the following syntax errors may occur when the first MB header of a slice is decoded.

- Some decoded motion vectors may be invalid, especially for those MBs near the picture edge.
- When $concealment\_motion\_vectors = 1$ in an intra-block, the marker bit in the MB will be wrong (i.e., equal to 0) with a probability of 0.5 (under the assumption that each bit in the video stream is distributed uniformly) and then a fake start code might also occur.

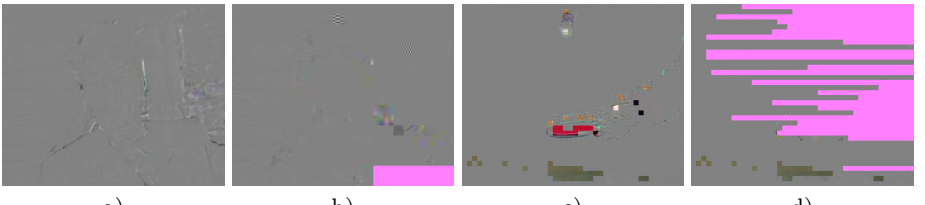
[2] All other MBs cannot be located without knowing Tables B-12/13/14/15.

– When *macroblock_pattern* = 1, "0000 0000 0" never occurs in *coded_block_pattern* encoded by Table B-9.

Since in each slice only the first MB header can be used to detect syntax errors about Table B-10, the average number of involved syntax elements may be too small, especially for pictures with a small number of slices and/or slow motion. Under such a condition, one has to exhaustively search for Table B-10 and Table B-14 together such that all motion vectors can be used.

Reconstructing Table B-14. Since all DCT coefficients in a non-intra MB are encoded with Table B-14, syntax errors may occur when a wrong Table B-14 is used to decoded a non-intra MB. Considering most MBs in a P/B-picture are non-intra MBs, the occurrence probability of such errors will be relatively high.

To test how frequently syntax errors of this kind occur in real attacks, we observed the decoding process by exchanging the following two VLC codewords in Table B-14 – "00101" and "000110", which represent RLE codewords (0,3) and (1,2), respectively[3]. For a large number of test MPEG-1/2 videos, syntax errors all occurred in the first P-picture (i.e., the 2nd picture of the whole video). Figure 1 shows the results for an MPEG-1 video "Carphone" (of size 176×144) and an MPEG-2 video "Tennis" (of size 704×576), where the pink areas (light grey areas on the printed version of the paper) in the decoded pictures denote decoding failures caused by syntax errors (the same hereinafter). Note that all the pictures are displayed as raw data (i.e., the differential pictures) since the reference I-pictures are still unknown at this stage of attack. If the whole Huffman table is heavily shuffled, syntax errors will definitely occur more frequently.



a) b) c) d)

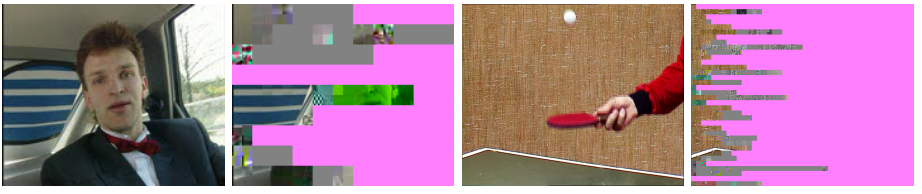
Fig. 1. The decoded results of an MPEG-1 video "Carphone" and an MPEG-2 video "Tennis", when only two VLC codewords were exchanged in Table B-14: a) the 2nd picture of "Carphone"; b) the decoded 2nd picture of "Carphone"; c) the 2nd picture of "Tennis"; d) the decoded 2nd picture of "Tennis"

Reconstructing Table B-12. Once Table B-14 is reconstructed, Table B-12 can be further exhaustively searched for in intra MBs with $intra\_vlc\_format = 0$. If all intra MBs in all known plain-videos are encoded with $intra\_vlc\_format = 1$, Table B-12 has to be exhaustively searched for together with Table B-15 (see

[3] These two VLC codewords were taken from the "good" VLC codewords selected by the authors of [15] to shuffle the corresponding Huffman table. The same rule also applies to other experiments in this paper.

below), which is generally a rare event when an attacker can collect a number of cipher-pictures to carry out the ciphertext-only attack.

In the case that only two VLC codewords, "00" and "01", in Table B-12 were swapped, we tested the decoding results for some MPEG-1/2 videos. Two results are shown in Fig. 2. Note that the swapped VLC codewords have the same bit size, so a stronger shuffling shall cause much more syntax errors.



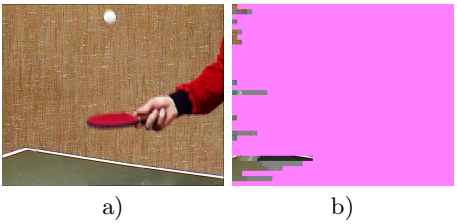
a) b) c) d)

Fig. 2. The decoded results of the MPEG-1 video "Carphone" and the MPEG-2 video "Tennis", when only two VLC codewords were exchanged in Table B-12: a) the 1st picture of "Carphone"; b) the decoded 1st picture of "Carphone"; c) the 1st picture of "Tennis"; d) the decoded 1st picture of "Tennis"

Reconstructing Table B-15. If Table B-12 has been successfully guessed, Table B-15 can be exhaustively searched for in luminance blocks of intra MBs with $intra\_vlc\_format = 1$, just like the case of reconstructing Table B-14. If Table B-12 cannot be separatedly broken, Tables B-12 and B-15 have to be exhaustively searched for together.

By swapping two VLC codewords "00101" and "000110", which represent RLE codewords (2,1) and (4,1), respectively, in Table B-15, we tested the decoding results of some MPEG-2 videos (note that this table is not used in the MPEG-1 standard). Figure 3 gives one result for the MPEG-2 video "Tennis".

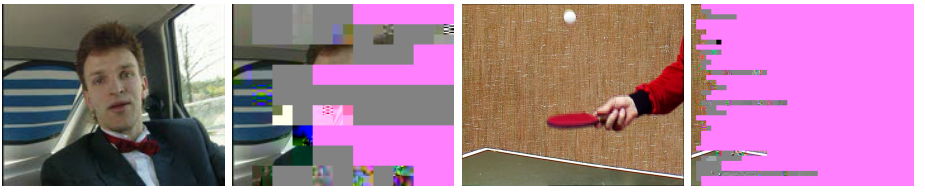
Reconstructing Table B-13. After Tables B-12, 14 and 15 are broken, Table B-13 can be exhaustively searched for in chrominance blocks of intra MBs. If there are intra MBs with $intra\_vlc\_format = 0$, Table B-13 can be exhaustively broken immediately after Table B-14 is broken, without knowing Table B-15.



a) b)

Fig. 3. The decoded result of the MPEG-2 video "Tennis", when only two VLC codewords were exchanged in Table B-15: a) the 1st picture of the original video; b) the decoded 1st picture

By exchanging two VLC codewords, "01" and "10", in Table B-13, we tested the decoding results of some MPEG-1/2 videos. The results corresponding to the MPEG-1 video "Carphone" and the MPEG-2 video "Tennis" are shown in Fig. 4. Once again, many syntax errors can be observed.



a) b) c) d)

Fig. 4. The decoded results of the MPEG-1 video "Carphone" and the MPEG-2 video "Tennis", when only two VLC codewords were exchanged in Table B-13: a) the 1st picture of "Carphone"; b) the decoded 1st picture of "Carphone"; c) the 1st picture of "Tennis"; d) the decoded 1st picture of "Tennis"

Finally, based on the above analysis, we can estimate the complexity of the DAC attack under four different conditions as follows:

- *when Table B-10 is separatedly searched for*:
  - *when Table B-12 is separatedly searched for*: $(3!)+(7!\times 2^6)+(6!\times 2^8)+(6!)+(16!)\approx 2^{44.3}$;
  - *when Table B-12 is searched for together with Table B-15*: $(3!)+(7!\times 2^6)\times(16!)+(6!\times 2^8)+(6!)\approx 2^{62.5}$;
- *when Table B-10 is not separatedly searched for*:
  - *when Table B-12 is separatedly searched for*: $(3!)\times(6!)+(7!\times 2^6)+(6!\times 2^8)+(16!)\approx 2^{44.3}$;
  - *when Table B-12 is searched for together with Table B-15*: $(3!)\times(6!)+(7!\times 2^6)\times(16!)+(6!\times 2^8)\approx 2^{62.5}$.

One can see that in all cases the complexity is much smaller than the one estimated in [15]: $(3!)\times(7!\times 2^6)\times(6!\times 2^8)\times(6!)\times(16!)\approx 2^{92}$.

As long as a VLC codeword appears frequently, syntax errors caused by encrypting it will occur with a high probability, which means that this VLC codeword will be reconstructed easily (i.e., wrong guesses of the VLC codeword can always be recognized). On the other hand, if a VLC codeword does not appear very frequently such that syntax errors do not occur for a given MPEG-video, the performance of encrypting this VLC codeword will not be "good" enough, so it should not be included in the key. In other words, a "good" key for encryption performance actually means a "bad" key that can be easily guessed with the DAC brute-force attack. Therefore, the efficiency of the DAC attack is ensured in a natural way. In addition, as shown in our experiments, only a single cipher-picture (or even several slices in a single cipher-picture) would be enough to carry out the above DAC attack effectively.

3.2 Chosen-Plaintext Attack

The chosen-plaintext attack is a very strong attack, in which one can (intentionally) choose some plaintexts and observe the corresponding ciphertexts to break an encryption scheme [21]. With the help of some chosen plaintexts and ciphertexts, it is possible to directly determine the secret Huffman tables without exhaustively guessing them in all possible candidates. In the following, we show how to choose a few number of plain-MBs to carry out a successful chosen-plaintext attack.

Reconstructing Table B-10. Choose a P-picture, in which there are a number of consecutive slices that contains only one "Not Coded" non-intra MB. In this case, there will be only one slice header and one MB header in each slice. Then, one can easily locate the only MB header in each slice, and extract a bit segment composed of two motion vectors from the MB header. In the extracted bit segment, the values of *motion_residuals*, *dmvectors*, and the sign bits of the motion vectors can be chosen to uniquely distinguish each *motion_code*, i.e., each VLC codeword encoded with the secret Table B-10. If necessary, $f\_code[r][s]$ can also be intentionally chosen to help the extraction of the VLC encoded *motion_codes*. By choosing the values of n consecutive *motion_codes* to be the n values corresponding to unknown VLC codewords, the whole secret Table B-10 can be reconstructed. Since $n = 3$ for the secret Table B-10 under study and each MB in a P-picture has two motion vectors, only 2 MBs (in 2 slices) are needed for this purpose. If a B-picture is chosen, then 1 MB is enough since there can be four motion vectors.

Reconstructing Table B-14. After reconstructing Table B-10, one can continue to break Table B-14 by choosing a block in a non-intra MB with the following pattern: "$(run_1, level_1)$, $(run_e, level_e)$, $\cdots$, $(run_i, level_i)$, $(run_e, level_e)$, $\cdots$, EOB" where $(run_i, level_i)$ is the i-th entry in the secret Table B-14 and $(run_e, level_e)$ is an Escape RLE codeword. By choosing $(run_e, level_e)$ properly, one can easily recognize each VLC codeword corresponding to the RLE codeword $(run_i, level_i)$. If a single block cannot contain all the encrypted VLC codewords, one more block can be chosen. For the encryption scheme under study, all the 6 encrypted VLC codewords in Table B-14 can be put in the same block, so only one chosen-block in a single non-intra MB is enough. Note that this table can also be broken in a similar way by choosing one intra blocks, after Tables B-12 and B-13 are firstly recovered as described in next paragraph.

Reconstructing Tables B-12/13. Since AC coefficients of intra-blocks are encoded in a similar way to the motion vectors, the method of reconstructing Table B-10 can also be used to break Tables B-12 and B-13. To break the entry corresponding to $dct\_dc\_size = s$, choose an intra-block as follows: "*level*, EOB", where the DC coefficient *level* has s significant bits. Then, the video bitstream corresponding to this block will be "*dct_dc_size*, *dc_dct_differential*, EOB". Since EOB and *dc_dct_differential* are both known, it is easy to determine the VLC encoded *dct_dc_size*. Given 7 luminance blocks with different values of s, all the

7 encrypted VLC codewords in Table B-12 can be reconstructed. Similarly, given 6 chrominance blocks, Table B-13 can be completely reconstructed. According to the value of $chroma\_format$, the maximal numbers of required chosen intra MBs for reconstructing Tables B-12 and 13 are 2 and 3, respectively. Since an MB can include both luminance and chrominance blocks, at most 3 intra MBs are needed to break the two secret Huffman tables.

Reconstructing Table B-15. After reconstructing Tables B-12 and B-13, one can break Table B-15 by choosing some intra-blocks, in the same way of reconstructing Table B-14. The 16 encrypted VLC codewords in Table B-15 and the RLE codewords used as locators can not be included in a single block, but two blocks in one intra MB are enough to reconstruct all the encrypted VLC codewords.

As a whole, to completely reconstruct all the secret Huffman table, at most 4 intra MBs in an I-picture and 3 non-intra MBs in a P-picture (or 2 non-intra MBs in a B-picture) are needed. While two chosen pictures are needed to break all the secret Huffman tables, note that not all the five Huffman tables are used for a single picture. For example, for an I-picture, Table B-10 is not used, and for a P- or B-picture, most blocks are non-intra coded without Tables B-12 and B-13. This implies that only one chosen picture is enough to recover all secret Huffman tables needed for decoding the same type of pictures (and part of other types of pictures). So the MPEG-video encryption scheme is very weak against chosen-plaintext attack.

4 More Discussions

In this section, a brief discussion is given on how to improve the security of the MPEG video encryption scheme under study. A simple measure is to change the secret Huffman tables frequently. In [15], it was suggested to reshuffle them after certain number of frames. Generally speaking, these reshuffling operations might be enough to provide an acceptable resistance against ciphertext-only attack. However, even reshuffling these Huffman tables frame by frame is generally not sufficient for the security against the above chosen-plaintext attack, since a few number of MBs may be enough to break the secret Huffman tables. From the most conservative point of view, one has to reshuffle the Huffman tables for each VLC codeword. Such a heavy reshuffling process will dramatically reduce the speed of the whole system and become impractical in many real applications.

Another possible solution is to use multiple Huffman tables (MHT) as suggested in [7, 16, 17, 18]. While some configurations of MHT encryption have been known insecure [19, 20], the following one remains secure: a stream cipher (or a secure PRNG) is adopted to determine the secret Huffman table from multiple candidate tables for each VLC codeword. However, as is well known in cryptology, a stream cipher is not secure against plaintext attacks if the key is reused to encrypt more than two plain messages. Thus, in real applications, to guarantee the security against plaintext attacks, some practical measures must

be adopted to avoid reuse of the same key for different plaintexts, such as using a key-management system to assign a different key for different encryption session.

Though using a stream cipher with MHT might be able to ensure the security against plaintext attack in practice, its performance could be worse than simply using the same stream cipher to mask the video bitstream. Assume that the number of different Huffman tables is 256 and the output of the stream cipher is a sequence of 8-bit integers. In this case, because the average bit size of VLC codewords is less than 8, more than one iterations of the stream cipher are required for encryption of each plain-byte. As a comparison, if the stream cipher's output is directly used to mask the video bitstream, only one iteration of the stream cipher is needed for each plain-byte. It needs more further research to see if there exist some other possibilities to enhance the performance of MHT-based secure Huffman coding algorithms.

5 Conclusions

This paper has re-analyzed the security of an MPEG-video encryption scheme based on secret Huffman tables. It is found that the scheme is not sufficiently secure against divide-and-conquer ciphertext-only attack, and is very weak against the chosen-plaintext attack. A brief discussion has also been given on how to improve the security of the MPEG-video encryption scheme.

Acknowledgments

Shujun Li was supported by a fellowship from the Zukunftskolleg, Universität Konstanz under the support of the "Exzellentinitiative" program of the German Research Foundation (Deutsche Forschungsgemeinschaft – DFG), and also by a research fellowship from the Alexander von Humboldt Foundation, Germany. K.-T. Lo was supported by the Research Grants Council of the Hong Kong SAR Government under Project no. 523206 (PolyU 5232/06E).

References

1. Qiao, L., Nahrsted, K.: Comparison of MPEG encryption algorithms. Comput. Graph. 22(4), 437–448 (1998)
2. Bhargava, B., Shi, C., Wang, S.Y.: MPEG video encryption algorithms. Multimedia Tools Appl. 24(1), 57–79 (2004)
3. Furht, B., Kirovski, D. (eds.): Multimedia Security Handbook. CRC Press, LLC (2004)
4. Furht, B., Muharemagic, E., Socek, D. (eds.): Multimedia Encryption and Watermarking. Springer, Heidelberg (2005)
5. Uhl, A., Pommer, A.: Image and Video Encryption: From Digital Rights Management to Secured Personal Communication. Springer, Heidelberg (2005)
6. Zeng, W., Yu, H., Lin, C.Y. (eds.): Multimedia Security Technologies for Digital Rights Management. Academic Press, London (2006)

7. Wu, C.P., Kuo, C.C.J.: Design of integrated multimedia compression and encryption systems. IEEE Trans. Multimedia 7(5), 828–839 (2005)
8. Wen, J., Severa, M., Zeng, W., Luttrell, M.H., Jin, W.: A format-compliant configurable encryption framework for access control of video. IEEE Trans. Circuits and Systems for Video Technology 12(6), 545–557 (2002)
9. Zeng, W., Lei, S.: Efficient frequency domain selective scrambling of digital video. IEEE Trans. Multimedia 5(1), 118–129 (2003)
10. Mao, Y., Wu, M.: A joint signal processing and cryptographic approach to multimedia encryption. IEEE Trans. Image Processing 15(7), 2061–2075 (2006)
11. ISO/IEC: Information technology – coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s – Part 2: Video. MPEG-1 standard: ISO/IEC 11172-2 (1993)
12. ISO/IEC, ITU-T: Information technology – generic coding of moving pictures and associated audio information: Video. MPEG-2 standard: ISO/IEC 13818-2 and ITU-T Rec. H.262 (2000)
13. ISO/IEC: Information technology – coding of audio-visual objects – Part 2: Visual. ISO/IEC 14496-2, MPEG-4 (2004)
14. Shi, C., Bhargava, B.: Light-weight MPEG video encryption algorithm. In: Shaping the Future: Proc. Int. Conference on Multimedia (Multimedia 1998), pp. 55–61 (1998)
15. Kankanhalli, M.S., Guan, T.T.: Compressed-domain scrambler/descrambler for digital video. IEEE Trans. Consumer Electronics 48(2), 356–365 (2002)
16. Wu, C.P., Kuo, C.C.J.: Fast encryption methods for audiovisual data confidentiality. In: Multimedia Systems and Applications III. Proc. SPIE, vol. 4209, pp. 284–295 (2001)
17. Wu, C.P., Kuo, C.C.J.: Efficient multimedia encryption via entropy codec design. In: Security and Watermarking of Multimedia Contents III. Proc. SPIE, vol. 4314, pp. 128–138 (2001)
18. Xie, D., Kuo, C.C.J.: An enhanced MHT encryption scheme for chosen plaintext attack. In: Internet Multimedia Management Systems IV. Proc. SPIE, vol. 5242, pp. 175–183 (2003)
19. Zhou, J., Liang, Z., Chen, Y., Au, O.C.: Security analysis of multimedia encryption schemes based on multiple Huffman table. IEEE Signal Processing Letters 14(3), 201–204 (2007)
20. Jakimoski, G., Subbalakshmi, K.P.: Cryptanalysis of some multimedia encryption schemes. IEEE Trans. Multimedia 10(3), 330–338 (2008)
21. Schneier, B.: Applied Cryptography – Protocols, Algorithms, and Souce Code in C, 2nd edn. John Wiley & Sons, Inc., New York (1996)