Investigating the Structure Preserving Encryption of High Efficiency Video Coding (HEVC)

Zafar Shahid, William Puech

▶ **To cite this version:**

Zafar Shahid, William Puech. Investigating the Structure Preserving Encryption of High Efficiency Video Coding (HEVC). Electronic Imaging, Feb 2013, San Francisco, CA, United States. pp.86560N, 10.1117/12.2011933 . lirmm-00830968

HAL Id: lirmm-00830968

https://hal-lirmm.ccsd.cnrs.fr/lirmm-00830968

Submitted on 6 Jun 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Investigating the Structure Preserving Encryption of High Efficiency Video Coding (HEVC)

Zafar Shahid and William Puech

Laboratory LIRMM, UMR CNRS 5506, University of Montpellier II
161, rue Ada, 34392 MONTPELLIER Cedex 05, FRANCE

zafar.shahid@lirmm.fr, william.puech@lirmm.fr

ABSTRACT

This paper presents a novel method for the real-time protection of new emerging High Efficiency Video Coding (HEVC) standard. Structure preserving selective encryption is being performed in CABAC entropy coding module of HEVC, which is significantly different from CABAC entropy coding of H.264/AVC. In CABAC of HEVC, exponential Golomb coding is replaced by truncated Rice (TR) up to a specific value for binarization of transform coefficients. Selective encryption is performed using AES cipher in cipher feedback mode on a plaintext of *binstrings* in a context aware manner. The encrypted bitstream has exactly the same bit-rate and is format complaint. Experimental evaluation and security analysis of the proposed algorithm is performed on several benchmark video sequences containing different combinations of motion, texture and objects.

1. INTRODUCTION

HEVC[1] is the emerging video coding standard of ITU-T and ISO/IEC. HEVC achieves similar visual quality to its precedent H.264/AVC High Profile, with around 30% bit-rate reduction for low delay mode, and with around 20% bit-rate reduction for random access mode on average, but with lower complexity than H.264/AVC Baseline Profile.[2] HEVC performs better because of some additional tools as summarized in Table 1.

It is pertinent to analyze this standard regarding its protection and authentication. Selective encryption (SE) is used to restrict access of video data to only authenticated users. For huge video data with real time constraints, SE encrypts a small part of the whole bit-stream with minimal resource overhead and provides sufficient protection for most of the applications. In this work, we have analyzed the selective encryption of HEVC in its CABAC entropy coding module, while fulfilling real-time constraints. Different encryption techniques including permutation, the Data Encryption Standard (DES) and the Advanced Encryption Standard (AES)[3] have been used for the SE of video. SE of H.264/AVC has been studied in Ref. 4 wherein encryption has been carried out in some fields like intra-prediction mode, residual data, inter-prediction mode and motion vectors.

The use of general entropy coder as encryption cipher has been studied in the literature in Ref. 5. It encrypts non-zero coefficients by using different Huffman tables for each input symbols. The tables, as well as the order in which they are used, are kept secret. This technique is vulnerable to known plaintext attacks as explained in Ref. 6. Key-based interval splitting of arithmetic coding (KSAC) has used an approach[7] where in intervals are partitioned in each iteration of arithmetic coding. Secret key is used to decide how the interval will be partitioned. Number of sub intervals in which an interval is divided should be kept small as it increases the bit rate of bitstream. Randomized arithmetic coding[8] is aimed at arithmetic coding but instead of partitioning of intervals like in KSAC, secret key is used to scramble the order of intervals.

Yeung *et al.* proposed the perceptual video encryption at the transform stage by selecting one out of multiple unitary transforms.[9] The unitary transforms were significantly different from discrete cosine transform (DCT) or discrete sine transform (DST), and the resulted coding efficiency is exactly the same to what can be achieved by using DCT or just falls very slightly. In Ref. 10, a quality assessment is performed for perceptual video encryption, wherein a user can still obtain some visible video contents (but at a very annoying quality) even without knowing the encryption key, is becoming more and more interesting in video applications such as video-on-demand (VoD) and pay-TV. The main drawback of this scheme is that it needs a modification in the transform module of Codec, which is very unlikely for hardware codec chips and even for DSP codecs. Moreover, it is a challenge to keep all the transforms in the instruction cache, especially for embedded devices.

High Efficiency Configuration	Low Complexity Configuration
Coding Unit tree structure (8x8 up to 64x64 luma samples)	
Prediction Units	
Transform unit tree structure (3 level max.)	Transform unit tree structure (2 level max.)
Transform block size of 4x4 to 32x32 samples (always square)	
Angular Intra Prediction (34 directions max.)	
DCT-based interpolation filter for luma samples (1/4-sample, 8-tap)	
DCT-based interpolation filter for luma samples (1/8-sample, 4-tap)	
Coding Unit based Skip & Prediction Unit based merging	
Advanced motion vector prediction	
CABAC	
Internal bit-depth increase (4 bits)	X
X	Transform precision extension (4 bits)
Deblocking filter	
Adaptive loop filter	X

Table 1: Tools for high efficiency and low complexity setting of HEVC.

Encrypted bitstream (EB) compliance is a required feature for multimedia applications and these techniques make the bitstream non-compliant and hence, cannot be decoded by standard H.264/AVC decoder. In Ref. 11, we have presented a selective encryption scheme of H.264/AVC based on CAVLC and CABAC entropy coding modules which fulfill real-time constraints by keeping the same bit rate and by generating completely compliant bitstream. Selective encryption of scalable extension of H.264/AVC has been proposed in literature by other researchers.[12,13] In Ref. 14, Dubois *et al.* have proposed a format-compliant reduced selective encryption, wherein encryption ratio has been reduced while maintaining the minimum visual quality.

This paper is organized as follows: In Section 2, overview of CABAC entropy coding engine of HEVC is presented. It explains the working of CABAC of HEVC, its difference from CABAC of H.264/AVC regarding selective encryption. The proposed algorithm is presented in Section 3. Section 4 contains its experimental evaluation and performance analysis for benchmark video sequences and different QP values. In Section 5, we present the concluding remarks about the proposed scheme.

2. CABAC ENTROPY CODING OF HEVC

The development of HEVC was started with two entropy codings like H.264/AVC. First technique was based on variable length based entropy coding called low complexity entropy coding (LCEC). LCEC was aimed for low complexity, while having the same compression ratio as CAVLC. Second entropy coding technique was context adaptive binary arithmetic coding (CABAC), which is based on the principles of arithmetic coding. Later on, LCEC was removed from HEVC codec reference design. From HM-5.0 onward, LCEC entropy coding is removed from HEVC and CABAC is the only entropy coding technique available in HEVC. CABAC of HEVC has more sophisticated binarization stage as compared to CABAC of H.264/AVC. Moreover, binary arithmetic coding unit of CABAC of HEVC is implemented using tables to reduce the required processing power.

CABAC consists of *binarization* stage and binary arithmetic coding (BAC) stage as shown in Fig. 1. In the *binarization* step, non-binary syntax elements are converted to binary form called *binstrings* which are more amenable to compression by BAC. Binary representation for a non-binary syntax element is done in such a way that it is close to minimum redundancy code. In CABAC of HEVC, there are five basic code trees for *binarization* step, as compared to four basic binarization trees in CABAC of H.264/AVC as shown in Fig. 2. They are the *unary* code, the *truncated unary* code, the *truncated rice* code with context p (TRp), the *kth order Exp-Golomb* code (EGk) and the *fixed length* code.

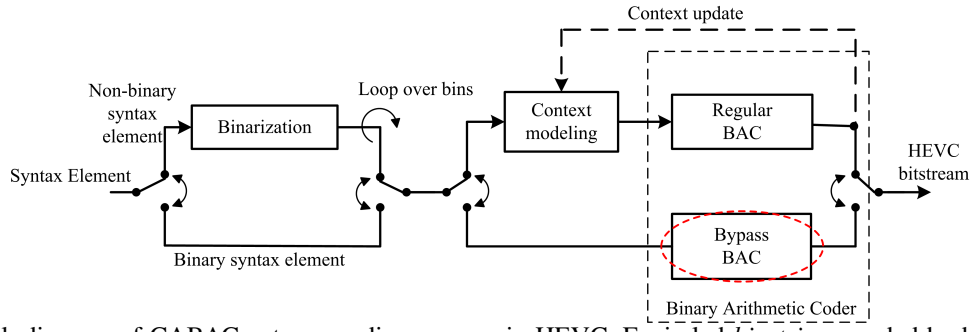


Figure 1: Block diagram of CABAC entropy coding process in HEVC. Encircled *binstrings*, coded by bypass-BAC are encrypt-able.

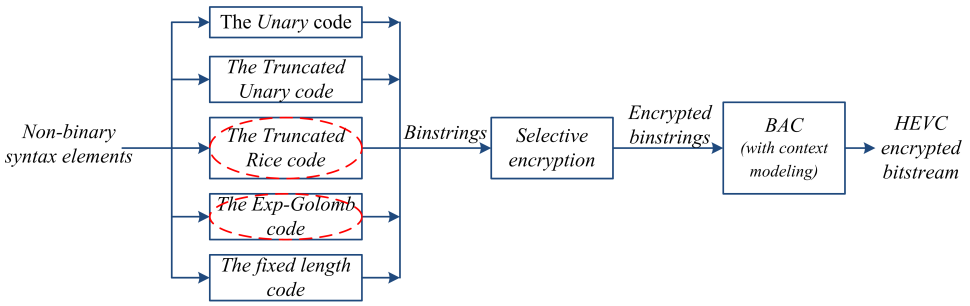


Figure 2: Selective encryption of CABAC-*binstrings* of HEVC. Encircled binarization techniques are encrypt-able.

For an unsigned integer value $x \geq 0$, the *unary* code consists of x 1's plus a terminating 0 bit. The *truncated unary* code (TU) is only defined for x with $0 \leq x \leq s$. For $x < s$, the code is given by the *unary* code, whereas for $x = s$ the terminating "0" bit is neglected.

The *truncated rice* code with context p (TRp) is the newly introduced basic binary tree in HEVC for the first time. A TRp *binarization* is a concatenation of quotient (q) and remainder (r) for a context p. For an unsigned integer value $x \geq 0$, the quotient q is given by $q = \lfloor \frac{x}{p} \rfloor$ and the remainder r is given by $r = x - qp$. For $p = 0$ the TR0 binarization is exactly the TU binarization.

The EGk code is also a concatenation of a prefix and a suffix. For a given unsigned integer value $x > 0$, the prefix part of the EGk *binstring* consists of a unary code corresponding to the length $l(x) = \left[log_2(\frac{x}{2^k} + 1)\right]$. The EGk suffix part is computed as the binary representation of $x + 2^k(1 - 2^{l(x)})$ using $k + l(x)$ significant bits. Consequently for EGk binarization, the code length is $2l(x) + k + 1$. When k = 0, the code length is $2l(x) + 1$. The *fixed length* code is applied to syntax elements with a nearly uniform distribution or to syntax elements, for which each bit in the *fixed length* code *binstring* represents a specific coding decision *e.g.*, *coded block flag*.

In HEVC, quantized transform coefficients (QTCs) are binarized by concatenation of the basic code trees. Binarization of absolute level of QTCs is done by concatenation of the *truncated rice* code (TRp) and $EG0$. Binarization and subsequent arithmetic coding process is applied to the syntax element $coeff\_abs\_value\_minus3 = abs\_level - 3$, since QTCs with zero magnitude are encoded using *significant map*.

3. THE PROPOSED ALGORITHM

Selectively encrypted HEVC bitstream will be format compliant and will fulfill real-time constraints provided the following conditions are fulfilled:

- **Bit-rate constraint:** the encrypted *binstring* must have the same length as the original *binstring*.

- **CABAC constraint:** the encrypted *binstring* must be valid and decode-able by entropy decoder.
- **Syntax element range constraint:** the decoded value of syntax element from encrypted *binstring* must stay in the valid range for that syntax element. This is important for syntax elements which are used for prediction of neighboring blocks.
- **Real-time constraint:** Encryption space is defined as the number of valid values for some syntax element. The candidate *binstrings* should have a dyadic encryption space, which can be represented by integer number of bits. This is mandatory for real-time selective encryption, wherein a number of *binstrings* are concatenated to prepare a plaintext for AES encryption in CFB mode.

Format compliant SE cannot be performed on CABAC bitstream. Rather it is performed on *binstrings* which are input to BAC as shown in Fig. 2. Among all the five *binarization* techniques, the *truncated rice* with context p (TRp) and EGk meet the conditions of selective encryption as highlighted in Fig. 2. The *unary* and *truncated unary* codes have different code lengths for each input value. They do not fulfill the first condition and their encryption will change the bit-rate of bitstream. In *fixed length*, different bits indicate different information regarding the header and is not viable for encryption. Suffix of EGk, and TRp can be encrypted while keeping the bit-rate unchanged and both of them are used for binarization of QTCs.

3.1 Encryption space

From CABAC entropy engine point of view, we can encrypt only those *binstrings* which use bypass-BAC mode and use fixed contexts as highlighted in Fig. 1. Contexts for *binstrings* coded by regular-BAC are adaptive and their encryption make the bitstream non-format compliant because of mismatch of contexts on encoder and decoder side. All the syntax elements, which use bypass-BAC, are not encrypt-able. There are other conditions to be fulfilled *e.g.* same bit-rate, dyadic encryption space and same context if a context is used during binarization. For example, context p is adaptive to QTC magnitude for TRp binarization. To keep the bitstream format compliant, it is not allowed to modify the context p for a specific QTC while performing selective encryption.

For binarization of QTCs, EGk code is used in H.264/AVC. In HEVC, it is replaced with TRp up to a specific value.[15] The main motivation for using TRp code is to increase the number of bypass bins by coding QTCs up to a specific maximum value with TRp. Moreover, EGk code is optimally fit for distribution of H.264/AVC residuals, which is geometrical. While distribution of HEVC residuals is different and TRp code performs better compression for HEVC residuals.

Fig. 3 shows the encrypt-able suffixes for QTCs. Let us denote the QTCs by *level* in this section. TRp codes are used for binarization of *levels* up to a threshold given by:

$$TR_{limit}[p] = \{7,\ 14,\ 26,\ 46,\ 78\}. \tag{1}$$

If *level* is smaller than $TR\_limit[p]$ for TRp, it will be coded using TRp codes and suffix of this TRp code will be encrypted as shown in Fig. 3. Context p for TRp code is adaptive to value of *level* in the following manner:

$$p_{next}[level] = \quad \{3,\ 5,\ 12,\ 24,\ \infty\} \tag{2}$$

For example, let the current context p is 0. It will remain unchanged if $level$ is 2, but if will be modified to 1 if current $level$ is 3. Context $p_{next}[level]$ must remain unchanged during encryption of QTCs. Table 2 shows the TRp code values for contexts 0 and 1. TRp codes which cannot be used for real-time encryption are colored gray in this table. In a TRp code with context p, we have p encrypt-able bits except two limitations. First, $TR0$ code is the same as truncated unary code and each *binstring* of $TR0$ has different length and is not encrypt-able because of violation of bit-rate constraint. Second, the length of last equal-length group of TRp codes is the same whether $EG0$ code is appended or not as shown in Table 2. So, if we encrypt the *binstrings* of this group, it may make the bitstream non-complaint. To fulfill this constraint we have to reduce the encrypt-able bits for last equal length TR codes to half. More precisely, we can encrypt only $(p-1)$ LSBs of the suffix **if MSB of the suffix is** 0. In Table 2, gray TRp codes are not encrypt-able because of this limitation.

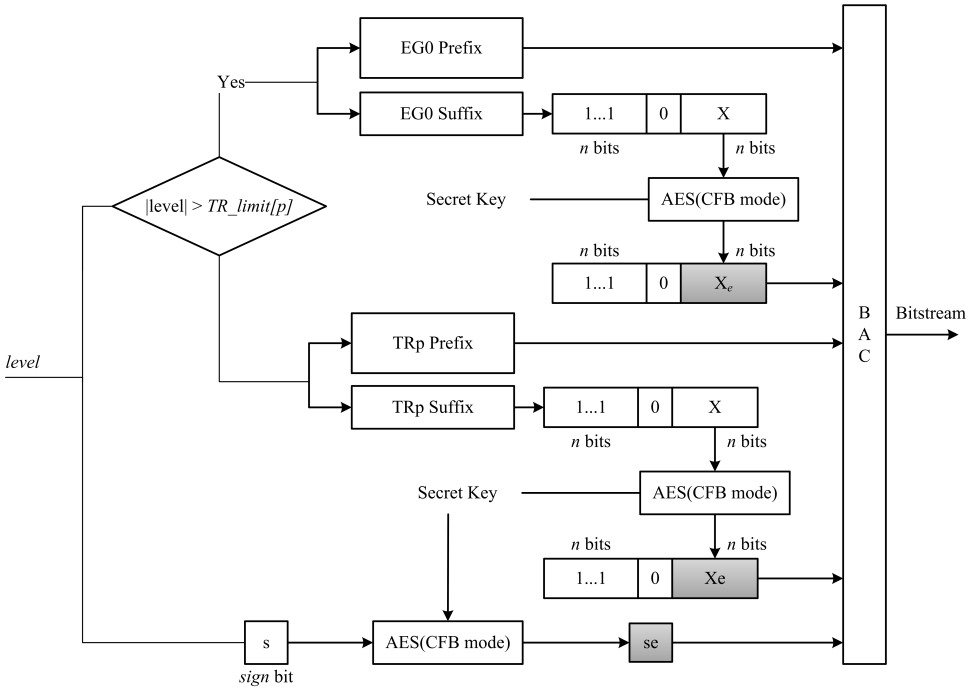


Figure 3: Encryption process for levels and their signs in CABAC of HEVC.

Symbol	TRcode[0]	Bits	TRcode[1]	Bits	...
0	0	1	0	2	
1	10	2	1	2	
2	110	3	100	3	
3	1110	4	101	3	
4	11110	5	1100	4	
5	111110	6	1101	4	
6	1111110	7	11100	5	
7	11111110	8	11101	5	
8	11111111,EG0	9	111100	6	
...					
18			11111111100	11	
19			11111111101	11	
20			11111111110	11	
21			11111111111,EG0	12	
...					

Table 2: TRp code with context p. Gray cells are not encrypt-able.

If *level* is higher than a $TR\_limit[p]$ limit, $EG0$ code is used for the binarization of $level - TR\_limit[p]$. In this case, suffix of $EG0$ will be encrypted as shown in Fig. 3. It is important to note that $p_{next}[level]$ constraint must be fulfilled even if $EG0$ code is appended.

Along with the first three constraints for selective encryption, it is mandatory to fulfill the real-time constraint *i.e.* to ensure the dyadic encryption space which can be represented by integer number of bits. In case of QTCs, the threshold for context switch of TRp code is not on a dyadic boundary as given in Eq. 1 and it may make the encryption space non-dyadic. Such type of encryption space can be encrypted separately, but it is not possible to put them in a plaintext for encryption by AES cipher in CFB mode.

3.2 SE of HEVC in the CABAC entropy coding stage

Let us consider $Y_i = X_i \oplus E_k(Y_{i-1})$ as the notation for the encryption of a n bit block X_i, using the secret key k with the AES cipher in CFB mode. We have chosen to use this mode in order to keep the original compression rate. Indeed, with the CFB mode for each block, the size of the encrypted data Y_i can be exactly the same one as the size of the plaintext X_i. In this mode, the code from the previously encrypted block is used to encrypt the current one.

HEVC has introduced the concept of entropy slices. Context models are reset at the start of every entropy slice. Moreover, entropy slices restrict definition for neighborhood. Hence it is pertinent to perform SE of HEVC on each entropy slice independently. SE is performed on *binstrings* before compression by BAC. Non-binary syntax elements qre transformed to *binstrings* through process of binarization and at the same time we fill the X_i with encrypt-able bits until either the vector X_i is completely filled or the slice boundary arrives. Let $L(X_i)$ be the length up to which vector X_i is filled. In case of slice boundary, if $L(X_i) < C$, we apply a padding function $p(j) = 0$, where $j \in \{L(X_i) + 1, \ldots, C\}$, to fill in the vector X_i with zeros up to C bits.

In the encryption step with AES in the CFB mode, the previous encrypted block Y_{i-1} is used as the input of the AES algorithm in order to create Z_i. Then, the current plaintext X_i is XORed with Z_i in order to generate the encrypted text Y_i.

With the CFB mode of the AES algorithm, the generation of the encrypted stream Z_i depends on the previous encrypted block Y_{i-1}. Consequently, if two plaintexts are identical $X_i = X_j$ in the CFB mode, then always the two corresponding encrypted blocks are different, $Y_i \neq Y_j$.

The decryption process in the CFB mode works in the same fashion except the fact that input is encrypted and output will be the plaintext. The decoded plaintext vector is split into segments in order to substitute the encrypted *binstrings* with original *binstrings*. The decoded *binstrings* substitute the encrypted *binstrings* in the bitstream. The bitstream is then further decoded to get the decrypted/original video frame using standard HEVC decoder steps.

4. EXPERIMENTAL RESULTS

In this section we analyze the results for the proposed selective encryption scheme for HEVC. We have used the reference implementation of HEVC HM 5.0 * for video sequences in HD resolution. For the experimental results, we have used the benchmark sequences of different resolution as shown in Table 3. This set of test material has been used in standardization process of HEVC.[16]

Class	Res.	FPS	Videos
A	2560×1600	30	Traffic (S01), PeopleOnStreet (S02)
B1	1920×1080	24	Kimono1 (S03), ParkScene (S04)
B2	1920×1080	50-60	Cactus (S05), BDrive (S06),BQTerrace (S07)
C	832×480	30-60	BasketballDrill (S08), BQMall (S09), PartyScene (S10), RaceHorses (S11)
D	416×240	30-60	BPass (S12), BQSquare (S13), BlowingBubbles (S14), RaceHorses (S15)
E	1280×720	60	Vidyo1 (S16), Vidyo2 (S17), Vidyo3 (S18)

*BDrive = BasketballDrive, BPass = BasketballPass

Table 3: Classification of benchmark video sequences used to evaluate the performance of the proposed SE technique.

To demonstrate the efficiency of our proposed scheme, we have compressed 50 frames with intra period of 10. We have applied simultaneously our SE and HEVC compression as described in Section 3, on all the benchmark video sequences. We have analyzed the available encryption space (ES) and PSNR value of encrypted bitstream. ES is defined as percentage of encrypt-able bits in a video bitstream.

4.1 Encryption space and PSNR analysis for benchmark video content

Table 4 compares the average ES and PSNR of 50 frames of all benchmark video sequences at QP value 18 without encryption and with SE. Average PSNR value of *luma* for all the sequences at QP value 18 is $9.67\ dB$ for SE. It confirms that this algorithm works well for various combinations of motion, texture and objects for intra and inter frames. Moreover, average PSNR values of U and V are $15.82\ dB$ and $17.23\ dB$ respectively, which are lower as compared to SE-CABAC

*https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/

of H.264/AVC values of 21.90 dB and 23.5 dB (Table IX of Ref. 11). Hence the proposed SE algorithm better protects the color information in HEVC video codec. Fig. 4 shows the visual quality of encrypted video frame for three different video sequences.

Table 4 also shows ES for SE for different benchmark video sequences for QP value 18, wherein ES varies from 14.83% to 17.76% based on the video content. Video sequences with static background (or with translational movement) like *Vidyo1* have less ES. *BQMall* has also a lesser ES despite its complex background. It is due to the fact that background is static and is coded by *skip* blocks and translational motion in the foreground is very efficiently predicted. On the other hand, video sequences which have complex motion and moving background have high ES. For example, *RaceHorses* contains walking horses, camera movement and high- texture grass in the background and it has higher ES.

(a) S06 (1920x1080_50) (b) S16 (1280x720_60) (c) S11 (832x480_30)

(d) SE (S06) (e) SE (S16) (f) SE (S11)

Figure 4: SE of different syntax elements for frame # 8 of several video sequences having different resolution and frame rate at QP value 18: a-c) Encoded frames, d-f) Encoded and encrypted frames.

4.2 Encryption space and PSNR analysis over whole range of QP values

Table 5 compares the average PSNR of 50 frames of *kimono1* over whole range of QP values without encryption and with SE. One can note PSNR values of *luma* and *chroma* components remains in the lower range for all values of QP and the proposed SE is effective over whole range of QP values. Table 5 also provides analysis of the effect of QP on the ES for *kimono1*. One can note that ES decreases from 18.24% to 13% with increase in QP value. It is due to the fact that data part of video bitstream decreases with increase in QP value. The decrease in ES with increase in the QP value is very less for HEVC as compared to SE of CABAC of H.264/AVC wherein ES decreases from 19.97% to 9.46% (Table II of Ref. 11). It is because of the introduction of TRp coding for binarization of QTCs.

4.3 Security Analysis

4.3.1 Histogram analysis

Histogram of the video frame gives the frequency distribution of the intensity levels or gray values. Here histograms of the original video frame and SE video frame are analyzed. Histogram graphs for original and SE video frame #8 of *kimono1* video sequence at QP value 18 are shown in Fig. 5. It is evident from Fig. 5.a and Fig. 5.b that both the histograms are entirely different. Moreover, histograms with 2 left and 2 right shifts in Fig. 5.c,d are also quite different.

4.3.2 Encryption quality analysis

The encryption quality (EQ) represents the average number of changes to each gray level L.[17] Let V and V' denote the original video frame and the encrypted video frame of a video sequence of resolution $m \times n$ with L intensity levels. $V(x,y)$, $V'(x,y) \in 0,..,L-1$ are the gray levels of video frames V and V' at position $(x,y)(0 \leq x \leq m-1, 0 \leq y \leq n-1)$. Let $H_L(V)$ denote the number of occurrences of each gray level L in the original frame V. Similarly, $H_L(V')$ denotes the number of occurrences of each gray level L in encrypted frameV'. The encryption quality is defined as:

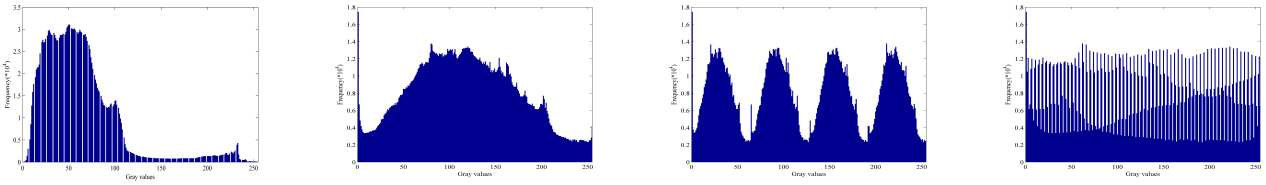
$$EQ = \frac{\sum_{L=0}^{255} H_L(V') - H_L(V)}{256}, \tag{3}$$

Seq	ES	PSNR (Y) (dB)		PSNR (U) (dB)		PSNR (V) (dB)	
	%	orig.	SE	orig.	SE	orig.	SE
S01	16.00	44.08	9.79	43.5	14.29	45.72	18.23
S02	15.88	43.88	8.23	46.36	15.28	45.82	17.67
S03	16.33	42.38	10.25	43.8	15.23	45.6	14.29
S04	15.75	43.41	9.55	44.84	12.78	46.82	17.27
S05	15.74	42.67	8.95	45.45	15.81	47.05	19.41
S06	17.59	43.42	7.89	43.62	14.59	45.28	17.26
S07	16.87	41.93	7.38	42.5	15.46	44.7	17.63
S08	15.40	43.61	11.35	45.32	13.64	46.51	11.84
S09	14.83	42.21	10.86	44.55	18.48	46.37	16.73
S10	16.98	43.04	8.99	43.86	16.13	44.55	20.23
S11	17.76	43.17	10.64	43.95	11.22	44.68	14.48
S12	15.69	44.66	13.76	46.61	23.46	46.66	14.39
S13	16.67	42.39	6.72	45.53	22.2	46.22	19.19
S14	15.98	41.92	12.15	43.39	12.04	45.21	22.22
S15	15.73	42.94	8.75	44.01	16.75	44.45	16.25
S16	15.42	45.35	10.32	47.68	15.23	48.78	18.95
S17	16.01	44.86	9.42	49.23	14.56	49.03	17.79
S18	15.45	45.35	8.98	49.34	17.54	49.61	16.39
avg.	16.11	43.4	9.67	45.2	15.82	46.28	17.23

Table 4: Encryption space and PSNR analysis of benchmark video sequences at QP value 18.

QP	ES	PSNR (Y) (dB)		PSNR (U) (dB)		PSNR (V) (dB)	
	(%)	orig.	SE	orig.	SE	orig.	SE
12	18.24	47.67	9.12	48.23	13.03	49.62	10.77
18	16.33	43.37	9.55	44.80	12.78	46.83	17.27
24	16.19	41.47	9.91	42.69	17.42	44.39	13.32
30	15.56	38.89	8.15	41.02	17.46	42.53	17.31
36	14.91	35.79	10.18	39.71	16.60	41.22	13.48
42	13.00	32.66	12.35	38.80	19.91	40.50	17.31

Table 5: Encryption space and PSNR analysis for *kimono1* video sequence over whole range of QP values.



(a) Original (b) SE frame (no shift) (c) SE frame (2 left shift) (d) SE frame (2 right shift)

Figure 5: Histogram analysis of original and SE video frame #8 of *kimono* video sequence at QP value 18.

The encryption quality for frame # 8 of *kimono* is shown for QP value of 18 in Table 6 for different number of shifts, wherein encryption quality remains in the higher range for all the shifts.

left bit shift	0	1	2	3	4	5	6	7
EQ	7002	6955	7100	6979	6879	7102	7199	6989

Table 6: Encryption quality for SE of *kimono* at QP value 18.

4.3.3 Sensitivity analysis

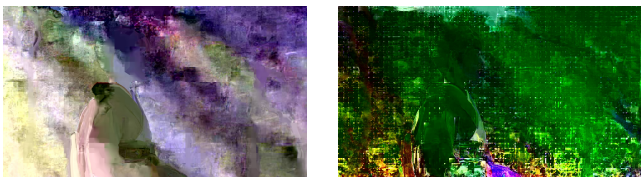
To provide an encryption algorithm with high security, cryptosystem should be highly sensitive towards the key to make any brute-force attack ineffective. The ciphertext should not be decrypted correctly although there is only a slight difference between encryption or decryption keys. For this purpose, a key sensitivity test is assumed where we pick one key and then applied the proposed technique for encryption and then make one bit change in the key and decode the bitstream. Numerical results show that the proposed technique is highly sensitive towards the key change, *i.e.*, the video frame decrypted with slightly different key is just another encrypted video frame, as shown in Fig. 6.

4.3.4 Approximation attack

If the syntax, context and statistical information is known a priori then the encrypted multimedia content may be approximately recovered even if the encrypted part is provably secure. To verify the ability of the proposed technique against approximation attack, it is assumed that an intruder knows the encrypted suffices of the *binstrings*, but not the bit values. These bit locations are then set to 0. An approximate copy of the original content is obtained by reconstructing the video frame with this partial information. Fig. 7 shows frame #0 of *kimono* video sequence with QP value 24 obtained with this partially assumed data. It is evident that quality of the decoded video frame is even worse than encrypted frame. For example, Fig. 7 shows that *luma* of attacked SE video frame has $PSNR = 5.45\ dB$ as shown in Fig. 7.

(a) (b)

Figure 6: Key sensitivity test for *kimono* #0. SE frame is decrypted with: a) Original key: PSNR (YUV)= $\{45.78,\ 47.04,\ 48.88\}$ dB, b) 1-bit different key: PSNR (YUV) = $\{9.45,\ 19.11,\ 13.63\}$ dB.



(a) (b)

Figure 7: Known plaintext attack for *kimono* #0: a) Encrypted frame: PSNR (YUV) = $\{10.12,\ 20.65,\ 14.23\}$ dB, b) Approximation attacked frame: PSNR (YUV) = $\{11.12,\ 20.19,\ 14.69\}$ dB.

5. CONCLUSION

In this paper, an efficient SE system has been proposed for HEVC video codec for its CABAC entropy coding. The security analysis verifies that the proposed scheme offer enough protection against crypt-analysis attacks. The SE is performed on the entropy slices independently in HEVC. In this way the proposed encryption method does not affect the parallelism of HEVC. Moreover, the proposed technique does not change bit-rate and the HEVC bitstream compliance. The SE is performed in CABAC *binstrings* such that they remain a valid *binstrings* having exactly the same length. The proposed method has the advantage of being suitable for streaming over heterogeneous networks because of no change in bit-rate. The experiments have shown that we can achieve the desired level of encryption under a minimal set of computational requirements.

REFERENCES

[1] HEVC, "High Efficiency Video Coding (HEVC) Text Specification Draft 6," tech. rep., ITU-T/ISO/IEC Joint Collaborative Team on Video Coding (JCT-VC), Doc. JCTVC-H1003, Geneva, CH (November 2011).

[2] Ugur, K., Andersson, K., and Fuldseth, A., "Video Coding Technology Proposal by Tandberg, Nokia, and Ericsson," tech. rep., Joint Video Team (JVT), JCTVC-A119, Dresden, Germany (April 2010).

[3] Uhl, A. and Pommer, A., [*Image and Video Encryption: From Digital Rights Management to Secured Personal Communication*], Springer (2005).

[4] Lian, S., Liu, Z., Ren, Z., and Wang, Z., "Selective Video Encryption Based on Advanced Video Coding," *Lecture notes in Computer Science, Springer-verlag* (3768), 281–290 (2005).

[5] Wu, C.-P. and Kuo, C.-C., "Design of Integrated Multimedia Compression and Encryption Systems," *IEEE Transactions on Multimedia* **7**, 828–839 (2005).

[6] Jakimoski, G. and Subbalakshmi, K., "Cryptanalysis of Some Multimedia Encryption Schemes," *IEEE Transactions on Multimedia* **10**, 330–338 (April 2008).

[7] Jiangtao, W., Hyungjin, K., and Villasenor, J., "Binary arithmetic coding with key-based interval splitting," *IEEE Signal Processing Letters* **13**, 69–72 (Feb. 2006).

[8] Grangetto, M., Magli, E., and Olmo, G., "Multimedia Selective Encryption by Means of Randomized Arithmetic Coding," *IEEE Transactions on Multimedia* **8**, 905–917 (October 2006).

[9] Yeung, S. K. A., Zhu, S., and Zeng, B., "Design of New Unitary Transforms for Perceptual Video Encryption," *IEEE Transactions on Circuits and Systems for Video Technology* **21**, 1341 –1345 (September 2011).

[10] Yeung, S. K. A., Zhu, S., and Zeng, B., "Quality Assessment for a Perceptual Video Encryption System," in [*Proc. IEEE International Conference on Wireless Communications, Networking and Information Security*], 102 –106 (June 2010).

[11] Shahid, Z., Chaumont, M., and Puech, W., "Fast Protection of H.264/AVC by Selective Encryption of CAVLC and CABAC for I & P Frame," *IEEE Transactions on Circuits and Systems for Video Technology* **21**, 565–576 (May 2010).

[12] Asghar, M. and Ghanbari, M., "An Efficient Security System for CABAC Bin-strings of H.264/SVC," *IEEE Transactions on Circuits and Systems for Video Technology* (99) (2012).

[13] Stutz, T. and Uhl, A., "A Survey of H.264 AVC/SVC Encryption," *IEEE Transactions on Circuits and Systems for Video Technology* **22**, 325 –339 (March 2012).

[14] Dubois, L., Puech, W., and Blanc-Talon, J., "Fast Protection of H.264/AVC by Reduced Selective Encryption of CAVLC," in [*Proc. European Signal Processing Conference*], 2185–2189 (2011).

[15] Nguyen, T., "CE11: Coding of transform coefficient levels with Golomb-Rice codes," tech. rep., ITU-T/ISO/IEC Joint Collaborative Team on Video Coding (JCT-VC), Doc. JCTVC-E253, Geneva, CH (March 16-23 2011).

[16] Baroncini, V., Ohm, J.-R., and Sullivan, G., "Report of Subjective Test Results of Responses to the Joint Call for Proposals (CfP) on Video Coding Technology for High Efficiency Video Coding (HEVC)," tech. rep., ITU-T/ISO/IEC Joint Collaborative Team on Video Coding (JCT-VC), Doc. JCTVC-A204, Geneva, CH (April 15-23 2010).

[17] Ahmed, H., Kalash, H., and Allah, O., "Encryption Efficiency Analysis and Security Evaluation of RC6 Block Cipher for Digital Images," in [*Proc. International Conference on Electrical Engineering*], 1 –7 (April 2007).