Hierarchical High Capacity Data Hiding in JPEG Crypto-compressed Images

Pauline Puteaux[1], Zichi Wang[2], Xinpeng Zhang[2] and William Puech[1]
[1]*LIRMM – Univ. Montpellier / CNRS*
Montpellier, France
[2]*School of Communication and Information Engineering – Shanghai University*
Shanghai, China

***Abstract*—With the fast development of cloud computing, exchanging JPEG compressed images in a secure way has significantly increased. Data hiding in encrypted images (DHEI) is an effective way to embed additional data directly into the encrypted domain. In recent state-of-the-art methods, almost all DHEI processes focused on uncompressed images. Recently, some schemes using data hiding (DH) in JPEG crypto-compressed images have been designed, but most of them are not fully JPEG format compliant. In this paper, we propose a hierarchical high capacity data hiding (HHCDH) approach for JPEG crypto-compressed images. After encrypting every non-null coefficients, they are processed from low to high frequencies. Sign bits that are specific to them are then substituted by bits of a secret message. During the decoding phase, correlations between neighboring blocks are exploited to hierarchically recover the original sign bit values. According to our experiments, we achieve to obtain a high payload value, while preserving a very good quality of the reconstructed JPEG image.**

***Index Terms*—Signal processing in the encrypted domain, data hiding, crypto-compression, JPEG compression.**

I. Introduction

During transfer or storage, bandwidth is limited and JPEG compressed images are often used [1]. Therefore, JPEG compressed image exchanges have significantly increased in the last years and their security can be exposed to privacy leaks.

For authentication, data enrichment or retrieval purposes in the confidential domain, methods of data hiding in encrypted images (DHEI) are effective. In these kinds of schemes, the owner of the image and the data hider are not necessary the same party, such as in a cloud scenario. In fact, encryption is completed by the content owner to protect image privacy and then, the encrypted image is sent over a network or uploaded on to a cloud server. Therefore, the server – considered as a data hider – has no access to the original image content and has to perform the secret message embedding directly on the encrypted data.

Even if most methods were designed for uncompressed images [2]–[4], some DHEI schemes focused on JPEG bitstreams specifically [5]–[8]. First, Qian *et al.* proposed to encode bits of a secret message with Error Correction Codes (ECC) [5]. In [6], the same authors suggested to form a new JPEG bitstream with some blocks from the original image. Unused blocks are also hidden in the JPEG header, using the same method as in JPEG XT [9]. Chang *et al.* proposed to reserve space for secret message embedding before bitstream encryption [7]. Recently, Qian *et al.* improved their previous scheme [6] using a combination of code mapping and ordered embedding [8]. None of these methods allows us to obtain a large payload, even using a high quality factor. Another issue lies in the fact that most of them are not fully JPEG format compliant.

In this paper, we propose a new framework of hierarchical high capacity data hiding (HHCDH) in JPEG crypto-compressed images. During the Huffman coding step, both AC and DC coefficients are encrypted. A secret message can be then embedded directly in the crypto-compressed domain. Adopting a chessboard configuration, diagonals from each Minimum Coded Unit (MCU) suitable for data embedding are processed, from low to high frequency coefficients. The sign bit of the largest coefficient of each diagonal is then substituted by a bit of the message. During the decoding phase, a hierarchical reconstruction is performed to predict the lost values. Finally, the reconstructed image quality is very good, even if a large amount of data has been embedded in the JPEG crypto-compressed image.

The rest of this paper is organized as follows. Section II describes our new framework for HHCDH in JPEG crypto-compressed images. Experimental results are presented in Section III. Finally, the conclusion is drawn in Section IV.

II. Proposed method

In this section we develop our proposed method of high capacity data hiding in JPEG crypto-compressed images. The encoding phase consists of crypto-compression and hierarchical data hiding, as presented in Fig. 1. After encrypting an original image, a user uploads it to a cloud platform. A data hider, can then process the crypto-compressed image in order to embed bits of an encrypted secret message.

A. Crypto-compression

From an original uncompressed image I, the first steps of JPEG compression are performed until the JPEG Huffman coding step to form MCU with 8×8 encoded frequency coefficients. These MCU are made up by pairs of head and amplitude parameters encoding the coefficient values.

Each block of the luminance Y component is then encrypted to obtain crypto-compressed blocks $\{F'_e(u,v)\}_{0\leq u,v<8}$. An

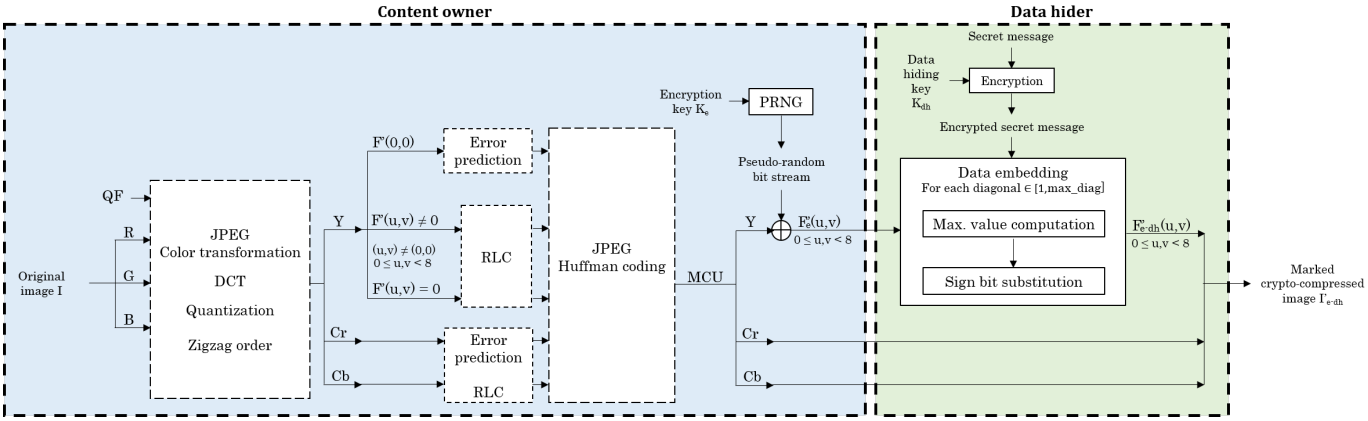

Fig. 1. Overview of the encoding phase of our HHCDH method in JPEG crypto-compressed images.

encryption key K_e is then used as the seed of a pseudo-random number generator (PRNG) to generate a pseudo-random bit stream. According to their amplitude size (in bits), each coefficient is encrypted by XORing its amplitude bits with its associated part in the generated pseudo-random bit stream, as illustrated in Fig. 1.

B. Data embedding

In the JPEG crypto-compressed bit stream, we propose using a chessboard configuration in order to perform the data embedding phase in only one encrypted MCU by two. Indeed, half of the MCU of the JPEG crypto-compressed bit stream are not modified so that they can be used for prediction during the reconstruction phase.

To embed bits of an encrypted secret message (using a key K_{dh}), each MCU with 8×8 encoded frequency coefficients is processed in a hierarchical way, from low to high coefficients, depending on the diagonal number in the original block before reordering and coding. A parameter *max_diag*, saved in the JFIF header, is fixed for each MCU, this indicates the number of the last marked diagonal. From the first to *max_diag*, each diagonal is analyzed and processed independently from each other. By analyzing the run-length value, it is possible to know if there is at least one non-null AC coefficient in the current diagonal. If there is, the maximum coefficient absolute value in the diagonal is considered. The associated coefficient is used to hold one encrypted bit $b(\alpha)$ of the secret message $\{b(\alpha)\}_{0 \leq \alpha < max\_diag}$ that can be embedded in the current MCU. Indeed, a marked encrypted coefficient $F'_{e\text{-}dh}(u,v)$ is obtained using sign bit substitution, according to Eq. (1) (we can see that the sign of a coefficient is indicated by the most significant bit (MSB) of its Huffman code):

$$F'_{e\text{-}dh}(u,v) = \begin{cases} F'_e(u,v), \text{ if } F'_e(u,v) < 0, \ b(\alpha) = 0, \\ F'_e(u,v), \text{ if } F'_e(u,v) > 0, \ b(\alpha) = 1, \\ -F'_e(u,v), \text{ else.} \end{cases} \tag{1}$$

The original sign bit value of the largest coefficient into each diagonal of the MCU suitable for data embedding is then lost. It must be predicted without error during the decoding phase. Therefore, if this value is modified due to bit substitution, sign bit values of all the other coefficients in the diagonal are also flipped in order to introduce noise and help the decoding step. Prediction performance during the decoding phase is then significantly increased. At the end, a marked JPEG crypto-compressed image $I'_{e\text{-}dh}$ is obtained.

C. Data extraction and image recovery

During the decoding phase, because the proposed method is separable, there are two possible outcomes, depending of the owned key.

On one hand, from the marked encrypted JPEG bitstream, if a recipient only has K_{dh}, they scan all marked encrypted MCU and collect the marked encrypted frequency coefficient values $\{F'_{e\text{-}dh}(u,v)\}_{0 \leq u,v < 8}$. According to *max_diag*, they process each diagonal $\#1$ to $\#max\_diag$, *i.e.* from low to high frequency coefficients. If there exists at least one non-null AC coefficient on this diagonal, they then focus on the frequency coefficient which has the largest absolute value and examine its sign. If the coefficient is negative, they extract 0 as the value of the expected embedded bit, else they extract 1. After processing each MCU, the encrypted secret message is then obtained and decrypted with the key K_{dh}. However, note that in this case the image content in the clear domain cannot be reconstructed.

On the other hand, if a recipient only has K_e, they also scan all marked encrypted MCU and they collect the marked encrypted frequency coefficient values $\{F'_{e\text{-}dh}(u,v)\}_{0 \leq u,v < 8}$. As an initialization step, the DC coefficient and all AC coefficients that are not on marked diagonals (*i.e.* whose the diagonal number is larger than *max_diag*), are directly decrypted using the encryption key K_e. Other frequency coefficients are initialized to zero. By this way, the recipient can then process each diagonal one by one, in order to hierarchically recover frequency coefficient values, from low to high frequencies. For each diagonal where there is at least one non-null frequency coefficient, two scenarios are considered:

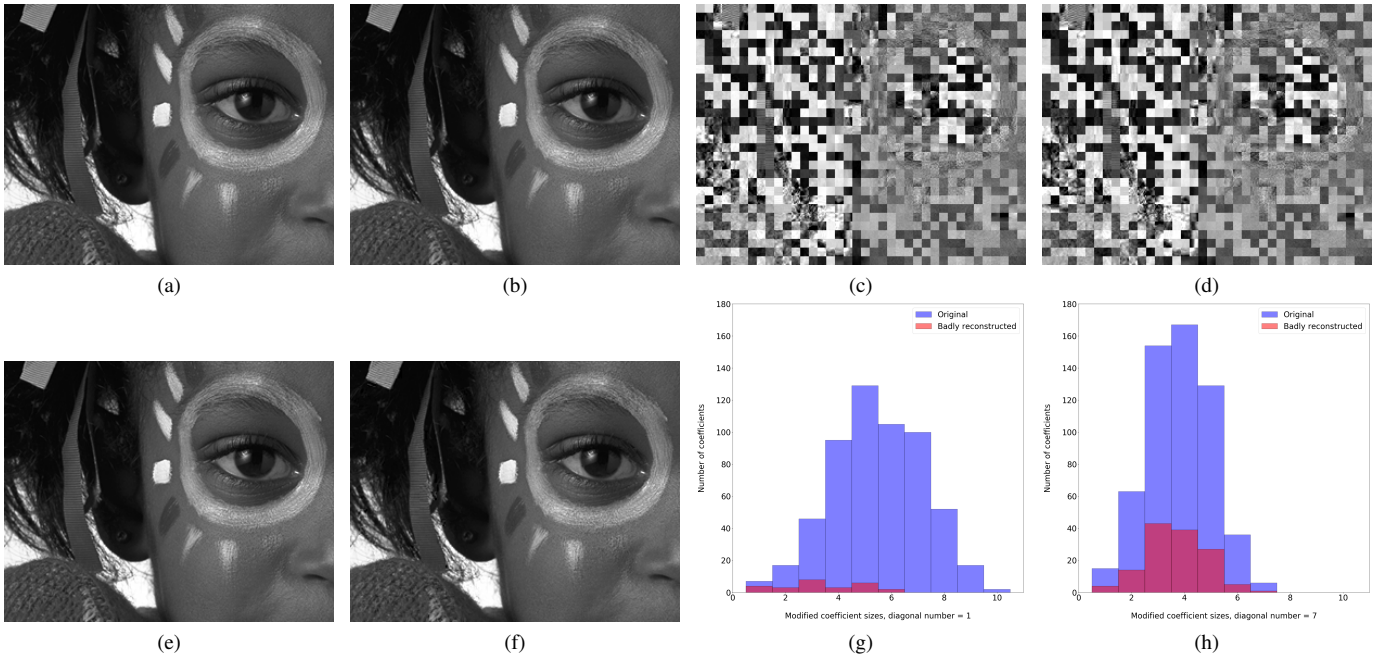

Fig. 2. Example of our proposed method: a) Original *Girl* image (304×240 pixels, 71.2 kB), b) JPEG image, QF $= 100\%$ (50.8 kB; with (a), PSNR $= 51.23$ dB, SSIM $= 0.9983$), c–d) Marked crypto-compressed JPEG images, embedding in the 1^{st} diagonal (payload of 570 bits), and in the 1^{st} to 7^{th} diagonals (payload of 3,988 bits) resp., e–f) Reconstructed JPEG images after correction from (c) (with (a), PSNR $= 47.87$ dB, SSIM $= 0.9972$; with (b), PSNR $= 51.46$ dB, SSIM $= 0.9989$) and from (d) (with (a), PSNR $= 34.19$ dB, SSIM $= 0.9407$; with (b), PSNR $= 34.26$ dB, SSIM $= 0.9417$) resp., g)–h) Coefficient size distributions according to (e) (1^{st} diagonal only) and to (f) (7^{th} diagonal only) resp.

1) All encrypted coefficient values on the diagonal have not been modified.
2) Sign bits of all encrypted coefficients on the diagonal were flipped during the data embedding step.

Therefore, the two associated MCU are computed. JPEG entropy decoding, inverse quantization and inverse-DCT (IDCT) are then applied to the two decrypted MCU, to obtain two possible pixel block configurations in the clear domain.

Correlation between neighboring blocks is thus exploited by computing a similarity score between each of these two configurations and neighboring blocks (in the clear domain in the chessboard configuration). As a similarity score, we chose to evaluate the differences between the current block edges and neighboring block edges. The block configuration that minimizes the difference is considered as the expected value in the clear domain. Indeed, the modification of only one coefficient bit sign in the frequency domain may significantly impact all pixel values in the current block, especially when the coefficient amplitude size is large. After recovering all coefficients of one diagonal, we hierarchically repeat the same steps on the next diagonal as soon as the diagonal number does not exceed *max_diag*. At the end of this process, all the coefficients are reconstructed and can be decoded to obtain the original image content, with the same quality factor QF as used during crypto-compression.

III. Experimental results

In this section, we present experimental results obtained by applying our proposed method of hierarchical RDH in JPEG crypto-compressed images. Note that even if our experiments have been done on grey-level images. They can be easily extended to color images, by considering the luminance component only for data embedding.

In Fig. 2, we have applied our proposed method on the *Girl* image (71.2 kB) illustrated in Fig. 2.a. Fig. 2.b corresponds to the image obtained with a standard JPEG compression in the clear domain using a quality factor QF $= 100\%$. Although the image size has decreased (50.8 kB), high quality has been preserved, as indicated by a PSNR of 51.23 dB and a SSIM of 0.9983. To perform the encryption step of our method, we propose to crypto-compress the JPEG image displayed in Fig. 2.a using a quality factor QF $= 100\%$. Non-null AC and DC coefficients are all encrypted. Note that this encryption method is format-compliant and size-preserving. Moreover, Fig. 2.c and Fig. 2.d are marked crypto-compressed JPEG images obtained by embedding information in the first diagonal and the first to seventh diagonals in the crypto-compressed image respectively. Using only the first diagonal, it is possible to embed 570 bits of the secret message, which means that half of the blocks of the image contain one bit of information. Using the 1^{st} to 7^{th} diagonals, a high payload value is achieved ($3,988$ bits, *i.e.* 0.054 bpp). Even with this large amount of embedded information, the visual confidentiality of the original content is not altered. Moreover, the file size is preserved: the marked crypto-compressed JPEG images (Fig. 2.c and Fig. 2.d) have exactly the same size as the compressed JPEG image in the clear domain. Fig. 2.e and Fig. 2.f are the reconstructed images after correction using the

TABLE I
PAYLOAD (IN BITS) ON 1,000 IMAGES USING A SIZE OF 512×512 PIXELS FROM [10], ACCORDING TO QF.

Average results	QF=100%		QF=90%		QF=80%		QF=50%	
	Diag. 1	Diag. 1 to 7	Diag. 1	Diag. 1 to 7	Diag. 1	Diag. 1 to 7	Diag. 1	Diag. 1 to 7
Payload (bits)	2,006	13,913	1,959	10,401	1,819	8,581	1,522	6,198
PSNR (dB)	51.34	37.75	50.84	37.62	50.18	37.43	48.10	36.71
SSIM	0.9984	0.9678	0.9981	0.9645	0.9978	0.9616	0.9965	0.9554

differences between neighboring blocks edges as a similarity score. Then, we compare these reconstructed images with the original uncompressed *Girl* image (Fig. 2.a) and with the reference image obtained after a standard JPEG compression in the clear domain using a quality factor QF $= 100\%$ (Fig. 2.b). Fig. 2.g and Fig. 2.h are the coefficient size distributions of the first and seventh diagonals respectively (obtained according to Fig. 2.e and Fig. 2.f). Using only the first diagonal for data embedding, we can see that almost all coefficients are correctly reconstructed, as indicated by the PSNR and the SSIM values with the original uncompressed image comparable to the values obtained between this image and the reference one. Even using the first to seventh diagonals, the reconstructed image is very similar to the original uncompressed version and the reference image (PSNR $\approx$ 35 dB, SSIM $\approx$ 0.95). Moreover, if we observe using the naked eye the original and reconstructed images, we cannot notice a difference, due to the fact that, even in the case of bad reconstruction, a wrong sign bit coefficient has not necessary an impact on the reconstructed pixel values. This is especially true in homogeneous areas, where frequency coefficients have very small sizes, even the low ones. According to the coefficient size distributions, we can see that most of the large size coefficients are correctly reconstructed. Indeed, they are easier to reconstruct because they are significant in the IDCT computation.

We have applied our method on the *Girl* image illustrated in Fig. 2.a, with four different quality factors (QF $= 100\%$, QF $= 90\%$, QF $= 80\%$ and QF $= 50\%$) and embedded variable amounts of bits of the secret message. Fig. 3 illustrates the quality of the reconstructed image with respect to the expected compressed JPEG image, in terms of PSNR. First of all, we can see that there is a real trade-off between the payload and the reconstructed image quality: the larger the embedding rate is, the lower the reconstructed image quality. This is explained by the fact that more bits of the JPEG bitstream are replaced by bits of the secret message and therefore, have to be predicted. Moreover, we can see that the quality factor QF also impacts the efficiency of the reconstruction. Indeed, with a small QF, it is more difficult to obtain a significant similarity score according to the neighboring values and therefore, to discriminate the correct configuration from the badly reconstructed one, in particular due to the natural block effect with JPEG compression.

In order to test the efficiency of our method for various image contents, we have applied our scheme to 1,000 randomly selected images using a size of 512×512 pixels from the BOSSbase dataset [10]. Table I presents the average results in

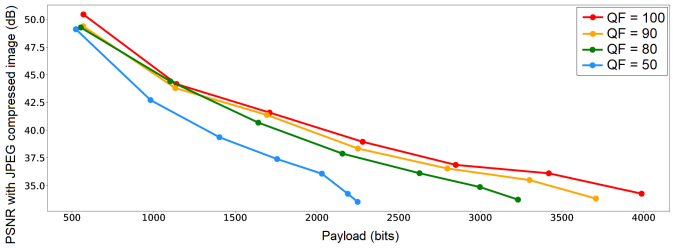

Fig. 3. PSNR (in dB) as a function of the payload (in bits) and QF with our method applied to the *Girl* image in Fig. 2.a.

terms of payload (in bits), PSNR (in dB) and SSIM (between each image of the database and their relative compressed JPEG image), according to different quality factors when only the first diagonal and when the first to seventh diagonals are used for data embedding. First, we can see that the lower the QF is, the lower the payload is. Indeed, if the QF is small, frequency coefficients are strongly quantized during the quantization step of JPEG. Consequently, most of them become equal to zero and are not suitable for data embedding. Nonetheless, we can remark that the first diagonal, composed of the lower two frequency coefficients, is almost always marked, even with a small QF. In addition, it is important to stress that the compression rate with a low quality factor (QF$= 50\%$) is equal to 0.35 bpp on average. Therefore, the image size (in kB) is much more smaller than using a high quality factor. Then, the ratio between the image size (in kB) and the payload is very attractive using QF$= 50\%$. If we observe the reconstructed image quality, whatever the used quality factor, when only the first diagonal is used for data embedding, results are very good (PSNR $\approx$ 50 dB and SSIM $\approx$ 1). By reconstructing in a hierarchical way each diagonal from the first to the seventh, the probability of a perfect reconstruction decreases, because high frequency coefficients are less predictable. Nevertheless, results remain interesting and the reconstructed image content is close to the expected compressed JPEG image content (PSNR $>$ 36 dB and SSIM $>$ 0.9555). Note that it is not recommended to use the next diagonals after the seventh) for data embedding because high frequency coefficients are not easily predictable.

In Table II, we compare our proposed method with previous approaches of Qian *et al.* in [5], [6] and [8] and of Chang *et al.* in [7], on five images with QF $= 80\%$. Even while embedding information only in the first diagonal using a chessboard configuration, we note that we can achieve a larger payload than methods [5], [6] and [7], but the payload remains smaller than using method [8]. Finally, if we use the first to the seventh

TABLE II
PAYLOAD COMPARISONS WITH CURRENT STATE-OF-THE-ART METHODS (QF $= 80\%$, IMAGE SIZE = 512×512 PIXELS).

Images	[5]	[6]	[7]	[8]	Diag. 1	Diag. 1 to 7
Lena	750	1,364	798	3,667	2,010	10,051
Baboon	750	768	1,555	7,447	2,044	13,785
Man	750	1,368	1,809	4,856	2,022	11,925
Peppers	750	1,026	960	4,253	2,013	10,436
Sailboat	750	1,023	1,032	4,964	2,006	11,675

diagonals to embed bits of the secret message, we can see that our proposed scheme can achieve a much larger payload than current state-of-the-art methods.

IV. CONCLUSION

In this paper, we proposed a new method of hierarchical high capacity data hiding in JPEG crypto-compressed images. In the crypto-compressed domain, we suggested embedding bits of a secret message on non-zero AC coefficients. Into each MCU suitable for data embedding, the sign bit of the largest coefficient of each diagonal is substituted by a bit of the hidden message. During the decoding phase, a hierarchical reconstruction is performed to predict the lost values, from low to high frequency coefficients. Indeed, significant coefficients, especially with a large amplitude size, can be correctly reconstructed with a high level of confidence. Experimental results show that our proposed method achieves a larger embedding rate when compared to other current state-of-the-art methods, while preserving the JPEG structure, the original image size and high visual quality. In future work, we are interested in computing similarity scores in the frequency domain to predict original sign bit values.

REFERENCES

[1] G. K. Wallace, "The JPEG still picture compression standard," *IEEE Transactions on Consumer Electronics*, vol. 38, no. 1, pp. XVIII–XXXIV, 1992.

[2] X. Cao, L. Du, X. Wei, D. Meng, and X. Guo, "High capacity reversible data hiding in encrypted images by patch-level sparse representation," *IEEE Transactions on Cybernetics*, vol. 46, no. 5, pp. 1132–1143, 2016.

[3] X. Zhang, J. Long, Z. Wang, and H. Cheng, "Lossless and reversible data hiding in encrypted images with public-key cryptography," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 9, pp. 1622–1631, 2016.

[4] P. Puteaux and W. Puech, "An efficient MSB prediction-based method for high-capacity reversible data hiding in encrypted images," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 7, pp. 1670–1681, 2018.

[5] Z. Qian, X. Zhang, and S. Wang, "Reversible data hiding in encrypted JPEG bitstream," *IEEE Transactions on Multimedia*, vol. 16, no. 5, pp. 1486–1491, 2014.

[6] Z. Qian, H. Zhou, X. Zhang, and W. Zhang, "Separable reversible data hiding in encrypted JPEG bitstreams," *IEEE Transactions on Dependable and Secure Computing*, 2016.

[7] J.-C. Chang, Y.-Z. Lu, and H.-L. Wu, "A separable reversible data hiding scheme for encrypted JPEG bitstreams," *Signal Processing*, vol. 133, no. C, pp. 135–143, 2017.

[8] Z. Qian, H. Xu, X. Luo, and X. Zhang, "New framework of reversible data hiding in encrypted JPEG bitstreams," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 2, pp. 351–362, 2019.

[9] T. Richter, A. Artusi, and T. Ebrahimi, "JPEG XT: A new family of JPEG backward-compatible standards," *IEEE Transactions on Multimedia*, vol. 23, no. 3, pp. 80–88, 2016.

[10] T. Pevný, T. Filler, and P. Bas, "BOSSBase dataset," http://dde.binghamton.edu/download/ImageDB/BOSS base_1.01.zip, Accessed on 15.10.2019.