Efficient and secure image availability and content protection

Hassan Noura, Mohamad Noura, Ola Salman, Raphael Couturier, Ali Chehab

Noname manuscript No.
(will be inserted by the editor)

Efficient & Secure Image Availability and Content Protection

Hassan Noura[1,2], Mohamad Noura[3], Ola Salman[2], Raphaël Couturier[3], and Ali Chehab[2]

[1]Arab Open University, Department of Computer Sciences, Beirut, Lebanon
[2]American University of Beirut, Electrical and Computer Engineering
[3]FEMTO-ST Institute, Univ. Bourgogne Franche-Comté (UBFC), France

the date of receipt and acceptance should be inserted later

Abstract Digital images are among the most communicated multimedia data types. Many of these images include private data that require a high level of security. The traditional image security schemes rely on cryptographic solutions to ensure the confidentiality or the authentication of image contents, and to ensure that the encryption key is not compromised. However, the continuous evolution of the attackers' capabilities is making it harder than ever to achieve the goal of safeguarding the private data against breaches. Moreover, the centralization aspect of images' storage makes them prone to availability attacks. In this paper, we propose a distributed and secure storage scheme for images, based on the Modified Information Dispersal Algorithm (MIDA), and taking into consideration the trade-off between the high security level and the associated computational overhead. The proposed solution applies block permutation on the image to ensure data confidentiality and then, divides it into k fragments that are encoded using the proposed parallel modified IDA. The output consists of n encoded fragments, instead of k, to ensure data availability. Next, each encoded fragment is authenticated using a lightweight Message Authentication Algorithm (MAA) to ensure data integrity with source authentication. Finally, the encoded fragments are distributed over n storage nodes (or multi-cloud providers). The resilience degree of such redundancy is $(n-k)$, since only k fragments are required to reconstruct the original images. All the cryptographic steps such as permutation, IDA encoding and MAA consist of simple operations and they are based on a dynamic key. This ensures a high level of security since in each session, a new key is used to produce different cryptographic primitives as well as the update primitives, which are used to update the permutation and selection tables. The implementation results show that the proposed scheme meets the desired cryptographic properties to guard against different attacks. Finally, the performance tests show that the proposed scheme is lightweight with low overhead in terms of computations, communication and storage.

Keywords: Lightweight data protection, Dynamic Key-Dependent cryptographic algorithms; Data availability, integrity and confidentiality; security and performance analysis.

1 Introduction

Data security has always been a major concern within the networking domain, and more so with the emergence of new types of applications in current and future networks, whereby tremendous amounts of critical data will be stored/shared in a digital form (e.g. multimedia contents). Accordingly, security concerns such as confidentiality, integrity, and availability have become more crucial than ever before. On the other hand, Several solutions have been proposed to ensure the security and privacy of sensitive data and to protect it from unauthorized access.

Typically, data confidentiality is ensured by using symmetric key encryption, which can be applied at the block or stream level. However, the encryption of multimedia contents such as images is different than that of plain texts due to the specific data characteristics of images [1–3]. Thus, traditional block cipher schemes such as the Advanced Encryption Standard (AES) [4] have been optimized towards making them suitable for securing images, but these solutions are based on applying a round function over multiple rounds. In every round, several substitution and diffusion operations are performed, which is computationally expensive. Within this context, recent image cipher schemes have been proposed with a low number of rounds (1 or 2) to overcome the multi-round

Address(es) of author(s) should be given

computational complexity [2,3,5–7]. These schemes are based on the dynamic key approach where the structure of all cipher primitives changes depending on the dynamic key. These primitives can be updated for each new input image, and this takes place at a low computational cost. However, these cipher schemes require hardware optimization similar to that of AES to achieve a better reduction when compared to optimized AES.

On the other hand, data availability is key to preventing data loss since new attacks suck as ransomwares (e.g. WannaCry) have shown their potential to compromise data systems. Typically, distributed storage systems can provide a reliable access to data through redundancy, where data is distributed among a group of nodes, and each node holds the whole data in encrypted or plaintext form. However, a better solution is to fragment the data and distribute the fragments among a set of nodes such that an individual node holds only an encrypted part of the original data.

As such, dispersing data fragments over multiple locations limits the risk of an attacker gaining access to the whole data, in contrast to the traditional centralized approach. The data fragments are normally generated using secret sharing such as Shamir secret sharing, information dispersal algorithms, or data shredding [8,9]. In this paper, a set of existing secret sharing schemes are presented, and they are described briefly in Section 2. These techniques provide secure data redundancy whereby each entity holds only a portion of the secret.

1.1 Related Work

Data can be fragmented in various ways such as the case in perfect secret sharing [10], computational secret sharing [11,12], information dispersal [13], and data shredding [14,15]. Since Shamir [10] and Blakley [16] presented their secret sharing schemes in 1979, the issue of secret sharing has been investigated widely in the last decades. In [17], Naor and Shamir applied the secret sharing concept to images in what they called visual cryptography. Also, the concept of (k, n)-threshold Secret Image Sharing (SIS) has been further extended in [18–21]. The SIS schemes can be divided into two classes, either based on polynomials or on Boolean-operations.

In general, a secret image is encoded as several shadow images (so-called shares or sub-images) using a Visual Secret Sharing (VSS) scheme [22,19]. This turns an image into n different noise-like shares. The secret image can be reconstructed, by the human visual system, by superimposing any k shares, where $n \geq k$. No information about the secret image could be revealed by collecting less than k shares. This is called the k-out-of-n scenario. The VSS scheme was first applied to gray images [23,18,24]. Then, an k-out-n VSS scheme was presented for colored images [25,26]. In this context, the pixel expansion approach was applied to constitute the n shares [27, 28]. However, the first VSS schemes, relying on the human visual system, and not on cryptographic methods for reconstructing the original data, were lossy. Thus, there were many proposals that considered cryptographic methods for lossless recovery of the original images [29–32]. In addition, many cryptographic algorithms have been considered for encrypting the data shares: bit-level encryption [33], standard encryption algorithms (AES [34], homomorphic encryption [35], etc. It is worth noting that homomorphic encryption introduced a drastic computational and storage overhead. In [18], the authors presented a k-out-of-n polynomial-based secret image sharing (PSIS) scheme to reconstruct loss-less secret images by applying the Lagrange interpolation technique. Later on, many extensions to PSIS have been proposed to meet different goals such as authentication [36], progressive secret image recovery [37], and essential shadows [38].

On the other hand, IDA was presented and described in [13]; an input file of size $|M|$ is divided into k pieces (fragments) with each piece having a size of $\frac{|M|}{k}$. Then, these k pieces are encoded by multiplying them by a static integer matrix to produce n encoded data fragments, and stored at n different storage devices. The main property of the selected matrix is that any k rows of this matrix should form a square invertible matrix. In addition, each encoded fragment is a linear combination (according to the employed specific row matrix) of all data fragments and matrix elements. The recovery is only possible when different k fragments are gathered. Recently, in [39], an IDA-based solution was proposed for images; it uses optimal Asymmetric Encryption Padding (OAEP) in addition to Information Dispersal Algorithms (IDA), which exhibits computational overhead since an asymmetric cryptographic algorithm is used instead of a symmetric one. Moreover, steganography is used to transmit the secret images within other non-secret images, which includes a high bandwidth overhead.

Thus, the proposed methods suffer from high computational complexity since they rely on the use of cryptographic algorithms with multi-operations and multiple rounds, and high storage overhead with secret sharing given that the share size is equal to the original image size. Therefore, the tremendous growth in the size of

communicated multimedia data, especially images, calls for new techniques that ensure an appropriate trade-off between the security level and computational and storage overhead, which is our main target.

1.2 Problem Formulation

A new scheme, intermediary between secret sharing and information dispersal, was recently sketched out in [40]. However, this approach suffers from high error propagation, and high computational overhead. The choice of the most appropriate method depends on the particularity of the use-case: perfect secret sharing may be highly secure, but it is slow and very costly in terms of memory and storage, however, the original IDA (threshold secret sharing scheme) can be resilient and relatively fast, yet it is not highly secure. In fact, Rabin's IDA has several advantages: it adds resiliency to data, produces almost ($n - k$ fragments) storage overhead, and uses simple arithmetic operations. However, this scheme only guarantees an incremental confidentiality level [9].

Towards overcoming the security issue of IDA, a new variant, All-Or-Nothing Transform, with Reed-Solomon coding "AONT-RS", was presented in [11]; it combines symmetric encryption (AES) with the original IDA. This solution is more secure compared to the original IDA and it is scalable, but in some scenarios, it might be less efficient than the information dispersal since the additional encryption process introduces computational and storage overhead.

Typically, any data protection solution needs to consider the challenging trade-off between cost effectiveness (including memory usage, processing, and computational complexity), and the required security level. Thus, there is a need for an efficient lightweight cryptographic scheme that can achieve strong data protection (data confidentiality, data integrity, and data availability) with the minimal possible processing overhead when applied to multimedia contents such as images, or any large data volume.

The existing secure variant of IDA (AONT) introduces processing and memory overhead, which hinders its adoption. Our main goal is to design a scheme that ensures a high security level without negatively impacting the system performance. Note that, designing an efficient AONT-RS variant that avoids complex operations is not a straightforward task.

1.3 Motivations and Contributions

In this paper, the original IDA scheme is modified to allow for parallel implementation (sub-matrix level) by using a set of IDA matrices instead of just one matrix to strike a good balance between efficiency and security. Moreover, we propose an encryption algorithm that consists of a single round and only one operation. Thus, the proposed scheme exhibits high efficiency in terms of computational complexity and resources, and a high security level compared to AONT. In addition, any lightweight message authentication algorithm [41] can be used to ensure the fragments integrity and authentication.

The proposed solution is suitable for the scenarios of limited processing and memory resources. It is based on a dynamic key-dependent IDA and dynamic cipher operations. The security performance results show that the proposed approach can prevent security attacks without degrading the system performance in terms of latency and energy consumption.

As shown in Fig. 1, the proposed model consists of a source node that aims at securely storing multimedia contents, such as images, by protecting and distributing it over n storage nodes. Overall, the proposed scheme has been designed to achieve the following goals:

1. **Robustness against attacks**: The proposed cryptographic solution is robust since it is based on 1) the dynamic key approach to generate and update the IDA matrices, 2) a permutation cipher table, and 3) two pseudo-random selection tables. Consequently, different dynamic cryptographic primitives are actually used for each input data. Moreover, the encoded fragments exhibit a high level of independence since the difference between the original and encoded fragments (encrypted) is close to $\frac{1}{2}$. Additionally, key sensitivity is ensured since any slight modification in the nonce or in the secret key leads to different dynamic key and consequently different cryptographic primitives. This translates to a probability of difference among encoded fragments close to $\frac{1}{2}$. Different cryptographic primitives are used for each input data, which limits the capabilities of existing cryptanalysis techniques to break it [3]. Also, the secret key size is flexible and can be set to either

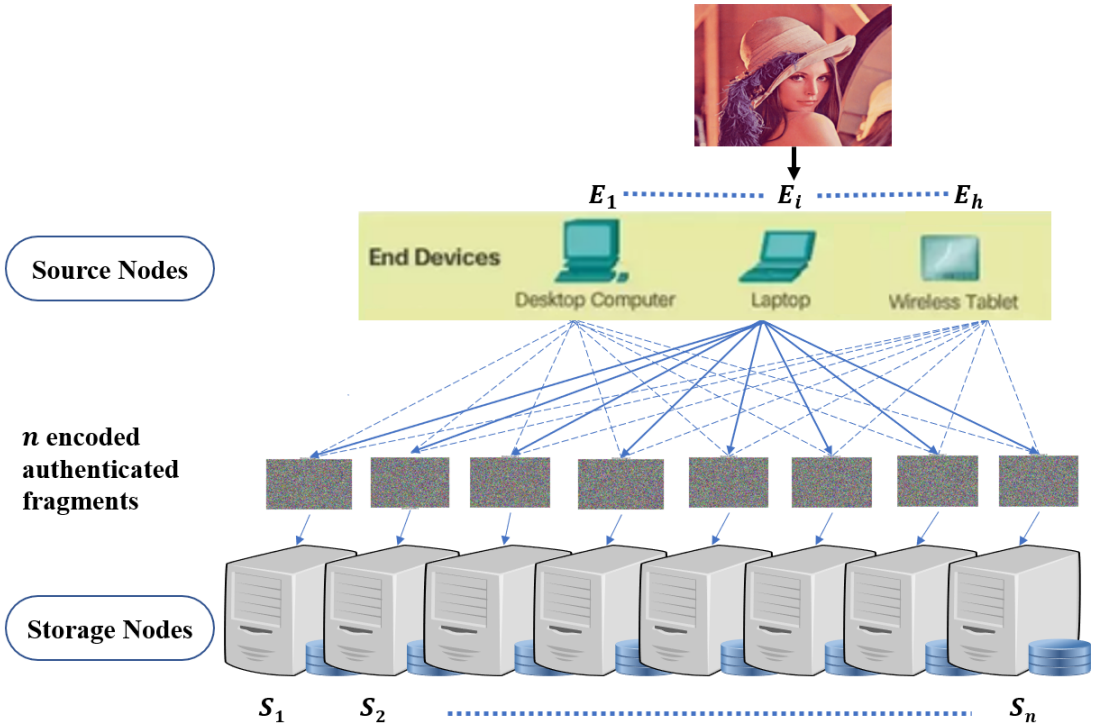


Fig. 1: Distributed Storage Model

128, 196 or 256, similarly to AES, and the size of the dynamic key is 512 bits. This makes the proposed solution immune against brute force attacks [42].

2. **Fast response time**: The use of a one-round cipher with a single operation (permutation) results in a low computational complexity and a fast encryption algorithm when compared to the existing cipher algorithms. In addition, the fragmentation/de-fragmentation algorithm can be implemented in parallel for different blocks, which reduces the latency by a factor of q, where q represents the number of threads.
3. **Adaptable to constrained devices:** The low execution time, simple implementation, and low memory requirements are all mandatory conditions to have an efficient model that accounts for the short battery life and low memory and processing resources, especially in the case when there is a need to store private images on personal devices.
4. **Low propagation error**: Since the proposed approach uses the dynamic key-dependent approach, the proposed algorithm can be considered as an Electronic Code Book (ECB) mode since no chaining operation among blocks is required in encryption or in IDA to avoid error propagation. In case the same image is repeated, it will be encrypted and encoded with different cryptographic primitives and this will result in different encoded fragments, which resolves the typical weakness issue of ECB. Moreover, the proposed modified IDA approach can be realized at the block level instead of the sub-matrix level to limit the error propagation to only one block. This is preferable in the case of noisy wireless communication channels.
5. **Simple implementation**: The proposed algorithm is based on simple operations such as look-up tables for the permutation, and selection and update primitives. Moreover, it uses addition and modulo multiplication operations, where the latter can be optimized using look-up tables for multiplication.

1.4 Organization

The rest of this paper is organized as follows: Section 2 presents existing secret sharing schemes, especially the IDA ones. Then, Section 3 presents the proposed generation process of dynamic keys and the techniques used to construct the cryptographic primitives and their update primitives such as the two pseudo-random selection tables, the permutation table, and the set of IDA matrices. Next, Section 4 presents the proposed dynamic key-dependent data protection cryptographic solution. In Section 5, a security analysis to assess its robustness is conducted, and a cryptanalysis discussion is provided in Section 6 to validate its immunity against well known

attacks. The performance evaluation of the proposed solution is presented in Section 7 to prove its efficiency. Finally, we conclude and present directions for future work in Section 8.

2 Secret Sharing Schemes

This section presents the most relevant schemes related to secret sharing, information dispersal, data shredding, and the all-or-nothing transform. Later in this paper, we show a comparison in terms of security and performance of these algorithms against our proposed solution.

2.1 Shamir's Secret Sharing

Shamir's perfect secret sharing scheme (SSS) [10] takes as input the message data M and divides it into n encoded fragments $EF_1, \ldots, EF_n$, of which at least k encoded fragments are needed for initial data recovery. This algorithm is based on the fact that given k unequal points $x_1, \ldots, x_k$ and arbitrary values $EF_1, \ldots, EF_k$, there is at most one polynomial $y(x)$, of degree less or equal to $k-1$, such that $y(x_i) = EF_i,\ i = 1, \ldots, k$. The SSS algorithm provides high level of confidentiality, but has quadratic complexity as a function of k and it exhibits a high memory overhead since the size of each fragment is as large as the size of the initial data. Therefore, SSS is usually applied to ensure the protection of small or critical data like encryption keys. In such a case, the drawbacks of the SSS scheme are acceptable and negligible, but for larger data, they present a major issue due to the high overhead in terms of computations, communication and storage.

2.2 Information Dispersal Algorithms

An Information Dispersal Algorithm (IDA) [13] divides input data M of size $|M|$ into a matrix F of size $k \times \frac{|M|}{k}$. This means M is divided into k rows (fragments), where each one has a size of $\frac{|M|}{k}$. The output of the IDA processing phase is n Encoded Fragments (EF) of size ($n \times \frac{|M|}{k}$), that are obtained by multiplying the non-singular IDA generator matrix G by X.

$$EF = G \oplus F \tag{1}$$

Information dispersal adds $(n-k)$ redundancy fragments that represent the IDA storage overhead but also its resilience degree. Recovery of the original data M consists of multiplying any different k encoded fragments of EF (EF_k) by the inverse of a $(k \times k)$ matrix built from k rows of the generator matrix G_k.

$$F = G_k^{-1} \oplus EF_k \tag{2}$$

In [43], Li analyzed the confidentiality of IDA. For instance, Rabin's IDA proposal was found to have high data confidentiality since the original data cannot be explicitly reconstructed from a number of fragments less than k. However, some information about its content could be leaked since the data patterns are preserved inside the fragments when the same matrix is reused to encode different data fragments. A similar problem occurs when using symmetric encryption with the Electronic Code Book (ECB) mode of operation [44].

2.3 Krawczyk's IDA Variant

The Krawczyk's variant [12] combines symmetric encryption with IDA for the protection of big data. the input data is encrypted using a symmetric encryption algorithm, then fragmented using IDA. The encryption key is fragmented using a perfect secret sharing scheme and is dispersed within data fragments. Accordingly, the solution does not require an explicit key management, and the storage overhead does not depend on the data size, but it is equal to the size of the key per data fragment. The performance of this technique depends on the details of the chosen encryption and IDA technique.

2.4 AONT-RS

The AONT-RS technique [11] is similar to Krawczyk's; it combines symmetric encryption with IDA. It applies an all-or-nothing transform (AONT) [45] to create n encoded fragments. First, the input data is encrypted using a strong block cipher such as AES. Then, the encrypted data is reshaped into $(k-1)$ fragments and an additional fragment is generated by *xor*-ing hashes of these data fragments with the encryption key. The last encoded fragment helps to validate the authentication and integrity of these fragments, which is not possible with the Krawczyk's IDA variant. Furthermore, this approach does not require an explicit key management. Data availability is ensured by producing $(n-k)$ additional fragments, which are obtained by using a systematic Reed-Solomon error correction code.

Table 1: Comparison of secret sharing variants ($h(M)$ represents the authentication of message M)

	Communication & Storage Overhead	**Key Management**	**Operation Base**	**Security Services**	**Additional Operations**
Secret Sharing Shamir	$(n-1) \times \|M\|$	Keyless	Polynomial	DA and weak DC	None
IDA	$(n-k) \times \frac{\|M\|}{k}$	Keyless	Matrix	DA and weak DC	None
AONT-RS	$(n-k) \times \frac{\|M\|}{k} + k \times h(\frac{\|M\|}{k})$	Xored with data	Systematic IDA	DA, DC, DI and SA	Encryption (block cipher) + PRNG + Hash
IDA + Over Encryption	$(n-k) \times \frac{\|M\|}{k}$	NA	IDA	DA and DC	Partial Encryption

According to Table 1, Shamir's original secret sharing suffers from high communication and storage overhead. IDA is presented as a solution to reduce this overhead. Recent AONT-IDA variants were presented to reduce the computational overhead in addition to offering more security services such as data confidentiality and integrity with source authentication.

The proposed cryptographic solution variant, which is described in the following section, follows this logic and uses dynamic key-dependent permutation operation (one round and also one operation), which requires minimum execution time as an encryption algorithm. This makes the proposed solution suitable for constrained devices and real-time applications with large data volume.

3 Proposed Key Derivation Scheme

In this section, we explain the generation process of the dynamic key(s) and the associated sub-keys that are used in the construction of the different cryptographic and update primitives. Fig. 3 illustrates the key derivation function, which takes as input a secret session key SK that should be unique for every new session (specified by the underlying application) to increase the security level. An efficient technique to generate the session keys is done via any Deterministic Pseudo-Random Number Generator (DRBG) [44] and by using a master key as a seed. The proposed technique to produce a new session key and a nonce is illustrated in Fig. 2. The master secret key MK is xor-ed with session meta-data (SM) and additional information such as a counter (CR). The output is hashed to produce a new dynamic session key $SK = h(MK \oplus SM \oplus CR)$, where h is a secure cryptographic hash function such as SHA-512. This operation ensures the sensitivity of the session key.

Note that, the secure hash function ($SHA-512$) is selected because it offers the desirable cryptographic hash properties such as the high resistance against collision attacks. The produced session key SK has 512 bits and it is renewed for every session and consequently, different cryptographic updates and primitives are produced, which guarantees the randomness of the proposed scheme. Employing dynamic cryptographic primitives (updated for

each new input image) provides high immunity against existing and modern attacks. Next, SK is divided into four sub-keys that form the seeds for the different cipher primitives, as described next.

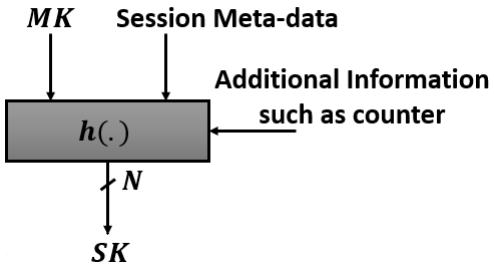


Fig. 2: Proposed session key generation process

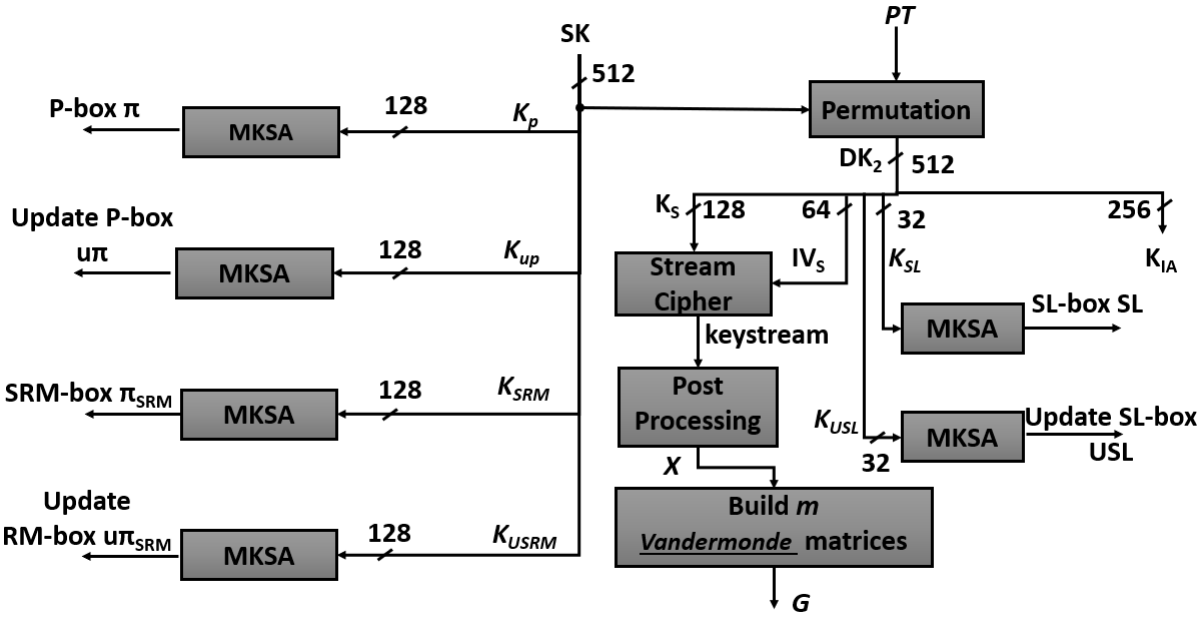


Fig. 3: Proposed key derivation function and its corresponding cipher and update primitives generation process

3.1 Derivation of Session Key & Sub-keys

SK can be changed as frequently as needed by the user or by the application by modifying the session time. SK has a size of 512 bits (64 bytes) and it will be split into four sub-keys, each with a size of 128 bits (16 bytes), $\{K_p,\ K_{up},\ K_{SRM},\ K_{USRM}\}$; each sub-key will be used for a different purpose within the algorithm (see Fig. 3).

1) **Permutation sub-key** K_p: it consists of the most significant 16 bytes of SK.
2) **Update permutation sub-key** K_{up}: it consists of the next most significant 16 bytes of SK.
3) **Selection matrices sub-key** K_{SRM}: this consists of the third most significant 16 bytes.
4) **Update selection matrices sub-key** K_{USRMP}: the least significant 16 bytes of SK.

In parallel, another dynamic session key SK_2 is obtained by permuting SK using the initial permutation table of DES, PT [46,42]. Then, SK_2 will be split into five sub-keys where two (K_SL and K_USL) have a size of

32 bits, the third one IV_S has a size of 64 bits, the fourth one K_S has a size of 128 bits (16 bytes) and the last one has a size of 256 bits. Each of these sub-keys will be used for a different purpose within the algorithm (see Fig. 3).

1) **Seed sub-key** K_S: it consists of the most significant 16 bytes of SK_2.
2) **Initial Vector sub-key** IV_S: it consists of the next most significant 8 bytes of SK_2.
3) **Update Selection sub-key** K_{USL}: it consists of the next most significant 4 bytes of SK_2.
4) **Selection sub-key** K_{SL}: it consists of the next most significant 4 bytes of SK_2.
5) **Message Integrity-Authentication sub-key** K_{IA}: the least significant 32 bytes of SK_2.

In summary, the size of the dynamic sub-keys (K_p, K_{up}, K_{SRM}, and K_{USRM}) is set to $L = 16$ bytes, while K_{SL} and K_{USL} are set to 4 bytes. The derived dynamic key is renewed for each session and any bit change will lead to a completely different set of dynamic sub-keys and thus, different cryptographic updates and primitives will be generated. Consequently, we obtain different cryptographic primitives for each new multimedia content. In the next section, we describe the construction of the proposed cryptographic updates and primitives that are based on these produced dynamic sub-keys. Note that Table 2 summarizes all the notations used in this paper.

3.2 Construction of Cryptographic Primitives

The main goal is to design a simple, yet very effective and efficient lightweight AONT-RS variant, which modifies IDA to allow an implementation at the block level (or sub-matrix level) and by using a set of m invertible matrices. In addition, the proposed solution uses a cipher algorithm that requires only one round and only one permutation operation per round. The permutation operation is selected to preserve the homomorphic properties, which is not possible if a substitution operation is used. The proposed solution must be suitable for constrained devices and for real-time applications, to preserve the privacy and data analysis at the storage node.

Fortunately, the dynamic key approach provides a high security level against existing and future powerful attacks [2,7,5,47]. This is ensured since all used sub-keys depend on the generated dynamic key and consequently, the cryptographic primitives become variable, preventing attackers from recovering information from the collected/chosen/known set of original/encoded (encrypted) fragments. More importantly, the dynamic key approach helps in reducing the required number of iterations [7,47]. Additionally, it reduces the required processing and memory resources and latency, which are key for preserving the main functionality of real-time applications. Moreover, to optimize IDA and to enhance its security level, we modify IDA by applying it at the block (k bytes) or at the sub-matrix level ($k \times k$ bytes). In fact, K_p, K_{up}, K_{SRM}, K_{USRM}, K_{SL} and K_{USL} are used in the proposed modified KSA algorithm to produce the permutation and update permutation tables π, $u\pi$, π_{SRM}, $u\pi_{SRM}$, SL, and USL, respectively. Next, we describe the technique for the construction of the key-dependent cipher primitives.

3.2.1 Dynamic Permutation Primitives

The one-round key-dependent encryption scheme is based on the permutation operation, either at the byte or at the block level (less computations compared to the byte level). The scheme generates a dynamic permutation table and it achieves high performance according to [48] since it requires only one operation with linear computational complexity. The permutation table π is updated for each new input image by using the update permutation table ($u\pi$). In this paper, the modified KSA of $RC4$, presented in [3], is used to produce the permutation tables. The modified KSA algorithm ($MKSA$) is described in Algorithm 1, where an input key of L bytes is used to produce a dynamic permutation table P with len elements.

The decryption process requires the use of the inverse permutation table. Since the produced π is bijective, the inverse of π, π^{-1}, can be easily obtained by: $\pi^{-1}[\pi(i)] = i$, where $\pi(i)$ represents the value of π at the i^{th} index and $1 \leq \pi(i) \leq |M|$. The process of permutation uses a swap function, where (i) and $(\pi(i))$ are the original and permuted byte/block positions of the vector image $|M|$.

3.2.2 Dynamic Selection Table of Sub-matrices

K_{SRM} is used as a seed for the proposed modified KSA algorithm to build a flexible key-dependent selection IDA matrix table SRM with ns elements (sub-fragment, each fragment can be a column or a sub-matrix of size $k \times k$), where $ns = \frac{|M|}{k}$. Moreover, the i^{th} IDA matrix used for encoding the i^{th} block (k permuted bytes) is

Table 2: Summary of notations

Notation	Definition
MK	Master key
IV_S	Initial vector sub-key
SK	Dynamic session key
SK_2	Second session key and it is obtained by permuting SK using the initial permutation table of DES.
K_p	Permutation sub-key used to construct the permutation table π
K_{up}	Update permutation sub-key used to construct the update permutation table $u\pi$, which is used to update the permutation table for each new input image file
K_{SRM}	Selection matrices sub-key used to construct the selection matrix table π_{SRM} that is used to select which one of the IDA matrices will be used during each input block
K_{USRM}	Update selection matrices sub-key used to construct the update permutation table $u\pi_{SRM}$, which is used to update the selection table π_{SRM} for each new input image file
K_S	Seed sub-key used as seed of any stream cipher that produces a keystream, which is post-processing to form m IDA matrices
K_{USL}	Update selection sub-key used to construct the update selection table π_{USL}
K_{SL}	Selection sub-key used to construct the update selection table π_{SL}
K_{IA}	Message integrity-authentication sub-key
π	A Dynamic produced permutation table (P-box)
π^{-1}	The inverse corresponding to the permutation table (P-box)
$\pi(i)$	The corresponding permuted value at the i index of π
X	The produced filtered keystream
x_i	The i^{th} block of the produced filtered keystream X and it is used to construct the G_i IDA vandermode matrix
G	A set of m dynamic IDA matrices
$G(i)$	The i^{th} dynamic IDA matrix
$G_k(i)$ and $G_k^{-1}(i)$	$k \times k$ of the i^{th} dynamic IDA matrix ($G(i)$) and its corresponding inverse matrix.
M	Original image file
$\|M\|$	The number of bytes in an input image file I after reshaped it to a vector
k	Number of fragments required for data recovery
ns	Number of sub-fragments inside initial data
$n, n \geq k$	Total number of fragments
SF_i	i^{th} original sub-fragment that has k bytes of length (column)
ESF_i	i^{th} encoded sub-fragment, a set of n bytes of encoded fragment
F_i	i^{th} original fragment that has ns bytes length
EF_i	i^{th} encoded fragment, which has ns bytes length
C	Columns number
R	Rows number
P	Plane number (for gray-scale is equal to 1 and 3 for RGB)

selected according to $\pi_{SRM}(i)^{th}$. In addition, $\pi_{SRM}(i)$ represents the value of the π_{SRM} at the i^{th} index and $1 \leq \pi_{SRM}(i) \leq m$, where m represents the produced IDA matrices and $m \leq ns$.

For the decryption process, the same operations are performed, but with the inverse IDA matrix $G_k^{-1}(\pi_{SRM}(i))$. Accordingly, π_{SRM} is used to control the modified IDA and its inverse processes. Another permutation table is required to control the IDA matrices for the fragmentation process (π_{SRM}), which is updated for each new input image via the update permutation table $u\pi_{SRM}$ to produce different encoded fragments.

Algorithm 1 Proposed Modified KSA (MKSA)

```
procedure MKSA(K = {k_1, k_2, ..., k_L}, L, len)
    for i ← 0 to 255 do
        π [i] ← i
    end for
    j ← 0
    for i ← 0 to len do
        j ← (j + π[i] + k[j mod L]) mod len
        swap(π[i], π[j])
    end for
    return π
end procedure
```

3.2.3 Dynamic Fragments Distribution Table based on π_{SL}

We use another permutation table, SL, Where $SL(i)$ represents the value of the SL at the i^{th} index and $1 \leq SL(i) \leq n$. SL can be considered as a permutation table with n elements, and it is used to control the distribution of the fragments to n different storage entities. SL is updated after each new input file by using the update permutation table USL. As such, the fragments of each image are distributed differently compared to the previous or next images. This increases the security level since if a storage entity is compromised, it would contain different fragments for different images, which is better than having a specific index fragment of all images at one entity.

3.2.4 Dynamic Key-Dependent Pseudo Random IDA Matrices

The IDA sub-key is divided into two parts (K_S; IV_S). The modified IDA scheme requires m IDA matrices that are chosen in a dynamic manner to ensure high randomness properties and to remove any existing pattern from the permuted message blocks. Each IDA matrix requires only a row of n unique and non zero bytes. To satisfy this condition, the $RC4$ stream cipher is used and it is iterated with a dynamic seed K_S and a dynamic initial vector IV_S towards producing the required key-stream of length $(m \times n)$ bytes. This key-stream is used to form a bank of m invertible matrices, where each column is used to produce one of the m invertible matrices.

The key-stream is post-processed and reshaped into m rows, each with n bytes such that each row does not contain any repeated or zero values. If these n bytes have zero bytes or repeated value(s), the stream cipher is re-iterated to replace them until no repeated or zero values are found in this block. These conditions are necessary to preserve the invertibility property of the obtained matrices (to recover the initial data). Each filtered row is used to construct an $(n \times k)$ IDA Vandermonde matrix. The Vandermonde matrices are used during the fragmentation process. Any k rows of any IDA matrix form an invertible $(k \times k)$ matrix. In fact, the Vandermode matrix form produces invertible matrices, only if the input block does not contain repeated or zero values ($x_i \neq x_j$ for $i = j = 1, 2, \ldots, n$). The Vandermode matrix form is expressed by the following equation:

$$G = \begin{bmatrix} 1 & x_1 & x_1^2 & \ldots & x_1^{k-1} \\ 1 & x_2 & x_2^2 & \ldots & x_2^{k-1} \\ 1 & x_3 & x_3^2 & \ldots & x_3^{k-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \ldots & x_n^{k-1} \end{bmatrix} \tag{3}$$

On the other hand, the dynamicity of the stream cipher avoids any possible security weakness and ensures a high level of randomness, uniformity, and periodicity. Note that any efficient secure stream cipher can be used, and the choice of RC4 is due to its simple software and hardware implementations and its ability to generate

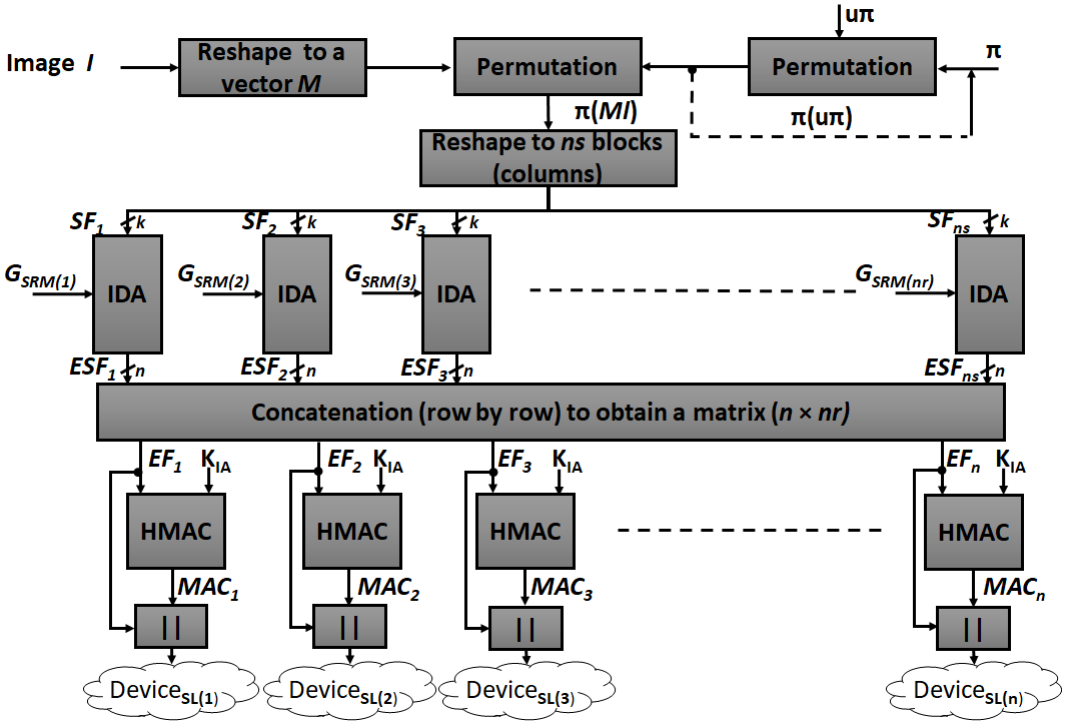


Fig. 4: Proposed cryptographic solution

permutation primitives with good cryptographic performance. In this paper, the RC4 stream cipher [49] is used only to produce the IDA matrices (cryptographic primitives) and not for the encryption/decryption process.

4 Proposed Cryptographic Solution

This section describes the steps of the proposed image (multimedia) cryptographic scheme, which are illustrated in Fig. 4. The first step is the encryption process by applying the permutation process, then the permuted image is reshaped to k original fragments, to which the modified IDA process is applied to produce n encoded fragments. Next, for each fragment, a message authentication process is applied to ensure data integrity and source authentication. In fact, the permutation-based encryption can be applied at the byte-level (size of sub-matrix is (1×1)) or for a set of bytes in a sub-matrix form such as (8×8) or (16×16) as shown in Fig. 6. In Fig. 5, we give examples of this sub-matrix size: 1 byte, 64 bytes, and 256 bytes. The encrypted image is reshaped into a matrix of size $(k \times \frac{|M|}{k})$. Each row of this matrix represents an encrypted fragment, and it has a length of $\frac{|M|}{k}$ bytes. Then, each column of this matrix (or a set of columns) is considered as a sub-fragment, which is encoded using modified IDA. In this example, we set $k = 4$ and $n = 8$. This means that the encoded sub-fragment consists of n bytes compared to k before IDA encoding. All columns are encoded and the output is concatenated to form a matrix of size $(n \times \frac{|M|}{k})$ bytes. Each row of this matrix forms an encrypted encoded fragment, which is authenticated using HMAC and the corresponding MAC value is appended to it.

The final output consists of n encoded authenticated fragments that are dispersed over n storage devices, which are selected in a dynamic manner. The second part of Fig. 5 illustrates the recovery process, where any k collected authenticated encoded fragments can be used to recover the encrypted permuted image vector. Then, the inverse permutation process, using the corresponding permutation table, is applied to recover the original vector, which will be reshaped to the original image size. The inverse cryptographic solution is not presented in details because it consists of the same operations (with the use of the inverse permutation table and inverse IDA matrices) in the reverse order.
The proposed algorithm is symmetric and is based on a session secret key SK, which is used to produce several sub-keys used in the construction of the cryptographic and update cryptographic primitives. These steps are detailed next.

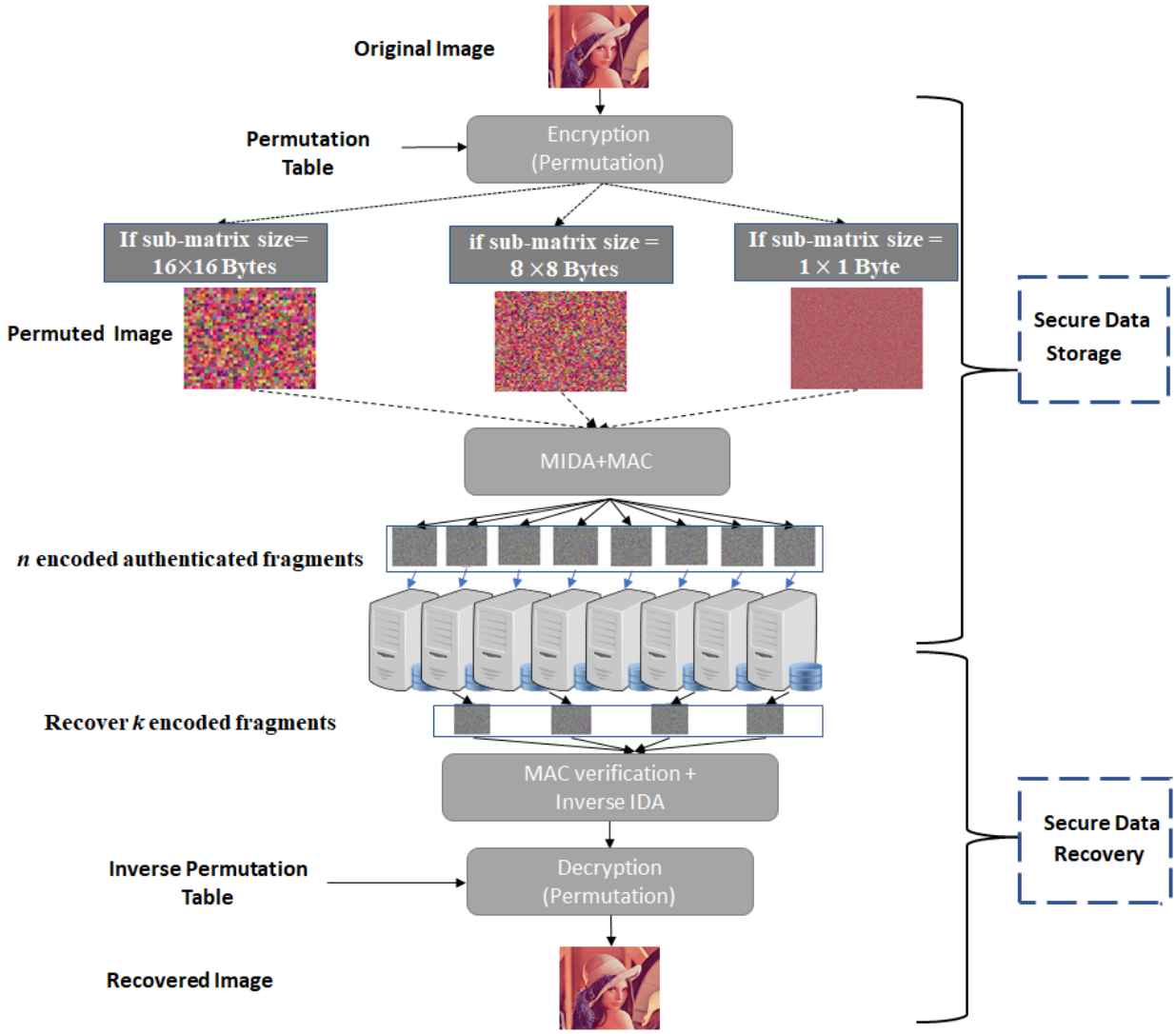


Fig. 5: An example of the proposed cryptographic solution

4.1 Encryption Permutation Process

The encryption process is based only on **a single round and one permutation operation**. First, an input image M is reshaped to a vector of size $|M| = (C \times R \times P)$, where C, R and P correspond to the number of columns, rows and plane in M, respectively. Then, The permutation is applied on M using the dynamic permutation table π.

Let the employed permutation table π of dimension α be defined by: $\pi=[p_i]_{1\leq i\leq \alpha}$. A plain-text M of length α is given by: $M = [M_i]_{1\leq i\leq \alpha}$. After permutation, the result is given by: $\pi(M) = [M_{p_i}]_{1\leq i\leq \alpha}$. For each input image, the permutation table is updated using the update table and thus, each image is permuted differently compared to the previous or next image. An example of the proposed key-dependent permutation image encryption scheme is presented in Fig. 6 for the original gray and color Lenna images for different permutation sub-matrix sizes of (1×1), (8×8) and (16×16), respectively.

A trade-off between the size of the permutation sub-matrix TB and the randomness degree is visually clear as shown in Fig. 6, where permuted gray and Lenna images are presented for different sizes. In Fig. 7-a, the results indicate that increasing the value of TB leads to an increase in the correlation between the adjacent pixels. From Fig. 7-b, showing the results of the fragmented encrypted images (permutation and IDA steps), we can see that for any sub-matrix size, a low correlation between the adjacent pixels is obtained. Consequently, the proposed IDA algorithm decreases noticeably the correlation between adjacent pixels and removes the spacial redundancy.

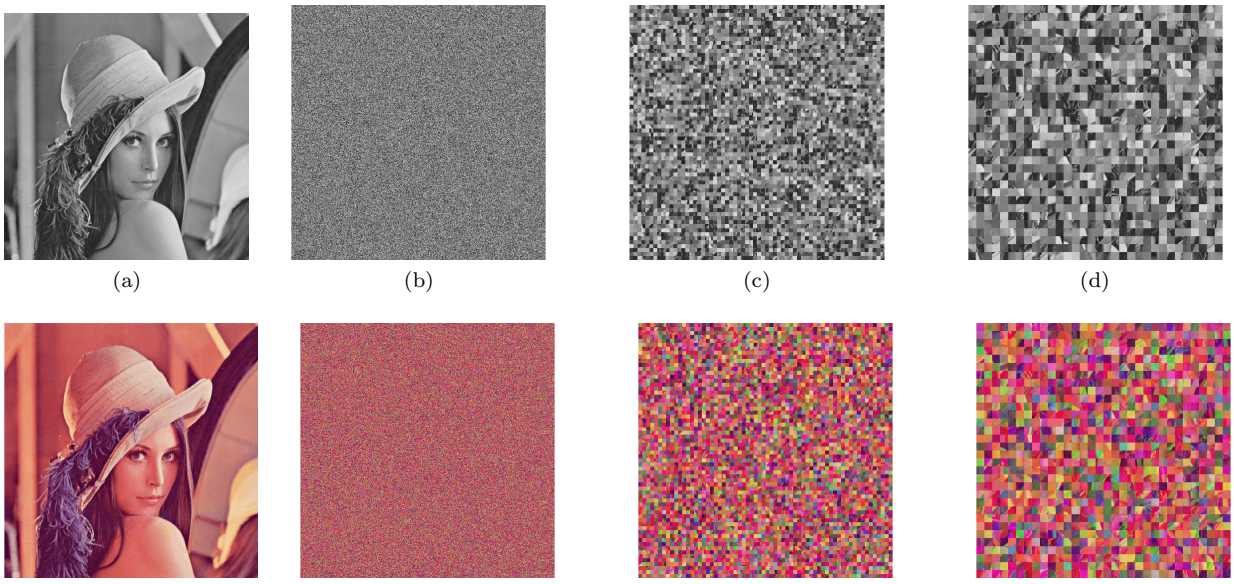


Fig. 6: (a) Original Gray Lenna image, (b)-(d) the corresponding permutations, (e) colored Lenna image, and (f)-(h) the corresponding permutations; for sub-matrix size (1× 1), (8×8) and (16×16), respectively

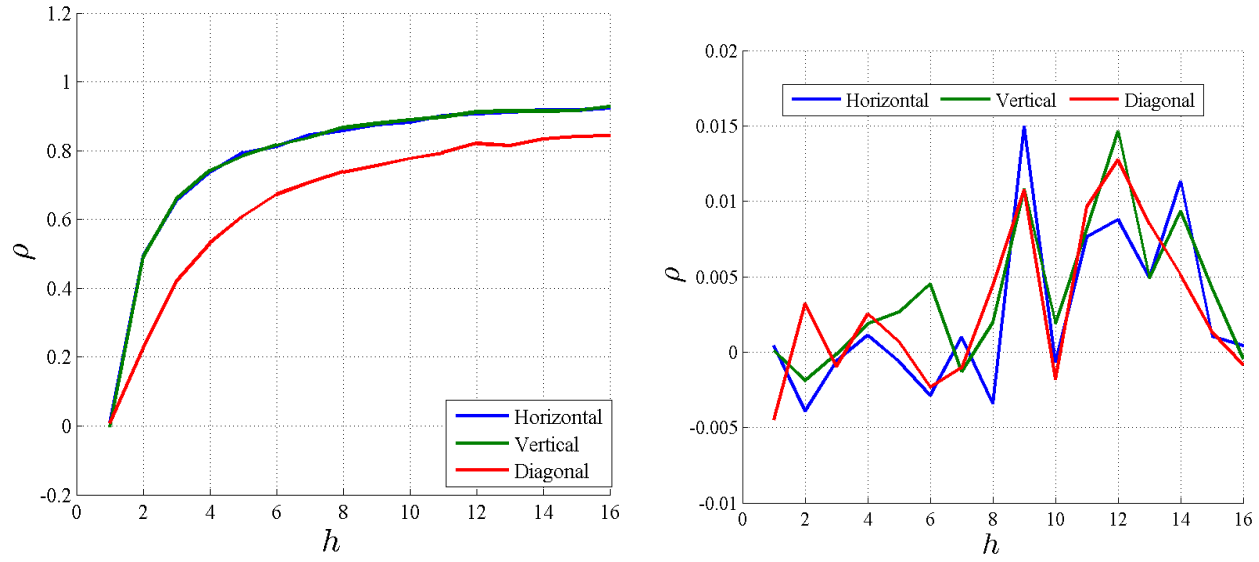


Fig. 7: (a) Correlation between adjacent pixels of encrypted image (permutation only) and (b), applying permutation with the proposed modified IDA algorithm on Lenna image versus h

4.2 Modified IDA Algorithm

The permuted encrypted vector is reshaped into k fragments, where each one has $\frac{M}{k}$ bytes. In addition, $|M|$ is equal to $\frac{C \times R \times P}{k}$, where k represents the threshold, and $M = \{m_1, m_2, \ldots, m_{R \times C \times P}\}$. The IDA algorithm can be considered as a channel encoding step and it is based on matrix multiplication. The IDA algorithm is shown in Algorithm 2 and it is modified in this paper to be applied at a block level (k bytes). The input is a permuted image vector M that is padded (if the number of bytes of the permuted message is not a multiple of k) and reshaped to form ns data sub-fragments $SF_1, \ldots, SF_{ns}$, where $ns = \lceil \frac{|Ml|}{k} \rceil$ and the length of each data fragment is k

bytes. These data sub-fragments are encoded one by one into ns data shares $ESF_1, \ldots, ESF_{ns}$ and the length of each data encoded sub-fragment is n bytes. For convenient processing, the sub-fragments are regrouped into ns columns (blocks) of k elements, where $SF_i(j)$ is the j^{th} byte in the i^{th} data sub-fragment block. The i^{th} data sub-fragment SF_i (a block of k bytes) is encoded using one of the m IDA matrices (see Fig. 8). The selection of the dynamic encoding IDA matrix depends on the selection permutation table, $\pi_{SRM} = [\pi_{SRM}(i)]_{1 \leq i \leq ns}$ and it has the same length of data sub-fragments, ns. Moreover, $\pi_{SRM}(i)$ is an integer value between 1 and m. The selection of the corresponding encoding matrix is controlled by the permutation table π_{SRM} that has a length of ns and contains elements varying from 1 to m, where m is the number of possible IDA matrices. $G = GS(\pi_{SRM}(i)$ indicates that $\pi_{SRM}(i)$ IDA matrix is used. Additionally, ns encoded sub-fragments are concatenated to form n fragments. Finally, n encoded fragments are distributed into n stored devices, where each row represents a fragment. This means that the i^{th} output row represents the i^{th} encoded fragment.

Even though k is flexible, however, increasing it leads to an increase of the security level and a decrease of the resiliency degree. In the rest of this paper, we fix k to 8. Note that k and n can be changed according to the possible number of storage devices and the required resilience/availability level. Any k fragments are needed for image recovery. Note that the enhanced IDA is built without altering some homomorphic properties (addition and multiplication by scalar) in contrast to AONT-RS where the homomorphic properties are lost (uses AES that relies on a substitution operation).

Algorithm 2 Proposed Modified IDA algorithm outline

1: $SF = SF_1, SF_2, , \ldots, SF_{ns}$
2: $ESF = ESF_1, ESF_2, \ldots, ESF_{ns}$
3: **for** $i = 1 \rightarrow ns$ **do**
4: $\quad G_i \leftarrow G(\pi_{SRM}[i])$
5: $\quad ESF_i \leftarrow (G_i \odot SF_i)$
6: **end for**
7: $EF \leftarrow ESF_1 || ESF_2 || \ldots || ESF_{ns}$
8: **for** $i = 1 \rightarrow n$ **do**
9: $\quad Distribute\ EF_i \rightarrow SL_i$
10: **end for**

4.3 Data Origin Authentication Scheme

The output of the modified IDA process is n encoded fragments $EF = \{EF_1, EF_2, \ldots, EF_n\}$. Each fragment EF_i is authenticated using a lightweight message authentication function or the standard cryptographic keyed hash function such as the HMAC presented in [50] with K_{IA} as a authentication-secret key. K_{IA} is used to authenticate each fragment EF_i. MAC_i is the i^{th} Message Authentication Code (MAC_i) of EF_i, which is concatenated at the end of its corresponding fragment ($EF_i || MAC_i$) and sent to one of the n storage devices.

4.4 Inverse Cryptographic Solution

The inverse process of the proposed cryptographic solution uses the same operations in reverse order. Having SK, all the secret cryptographic primitives can be generated and consequently, their corresponding inverse ones. The decryption process is based on the following steps:

1. First, we generate the dynamic secret key and construct the cryptographic and update primitives. Then, the required inverse cryptographic primitives are computed.
2. **Collection of authenticated k fragments received from k different storage devices:** The data integrity and source authentication of each received fragment are verified by applying the same keyed hash function with the specific origin authentication key K_IA. Each fragment is validated if the received MAC is equal to the computed one.
3. **Inverse IDA decoding (de-fragmentation)**: The verified k fragments are grouped into a matrix, where each column represents k bytes of the encoded sub-fragments. Then, the inverse modified IDA process is performed by using the inverse IDA matrices of the received k encoded fragments. The receiving end host can generate the same secret IDA matrices and the same dynamic matrix selection permutation algorithm by using the corresponding dynamic key. The inverse modified IDA decoding algorithm is done per block and by

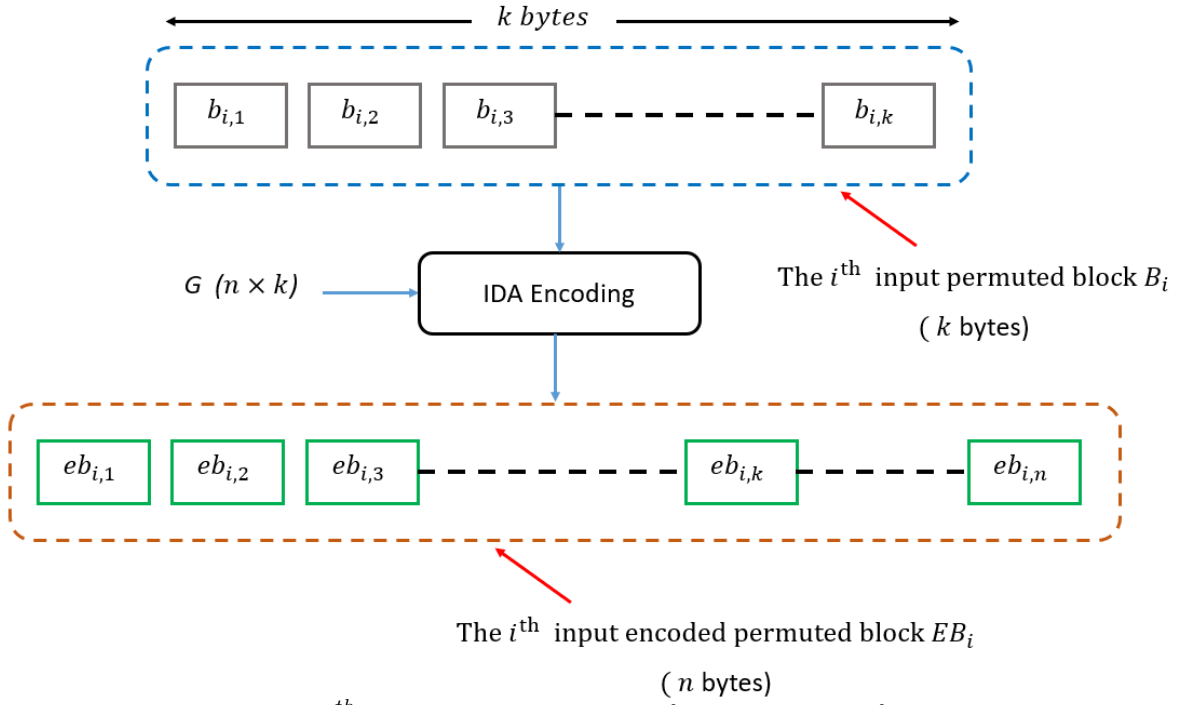


Fig. 8: Encoding example: the i^{th} permuted data block $B_i = \{b_{i,1},\ b_{i,2},\ \ldots,\ b_{i,k}\}$, representing SF_i, is transformed into the i^{th} encoding permuted block $EB_i = \{eb_{i,1},\ eb_{i,2},\ \ldots,\ eb_{i,n}\}$, representing ESF_i in this paper and $i = 1,\ 2,\ \ldots,\ ns$

multiplying with the inverse matrices G_k^{-1} (according to the selected k fragments). Accordingly, the matrix multiplication of each encoded sub-fragment and its corresponding inverse matrix is performed to recover the permuted sub-fragment. Next, all permuted sub-matrices are stored (column by row) to form a matrix of $(ns \times K)$, that will be reshaped to form the permuted image vector.

4. **Decryption (inverse permutation using the inverse permutation table π^{-1})**: At this stage, the inverse corresponding dynamic key-dependent permutation table π^{-1} is applied to the permuted image vector to recover the original image. In fact, the inverse permutation table can be obtained from the original permutation table by: $\pi^{-1}[\pi[i]] = i$. Finally, the image is reshaped to the original size $(L \times C \times P)$.

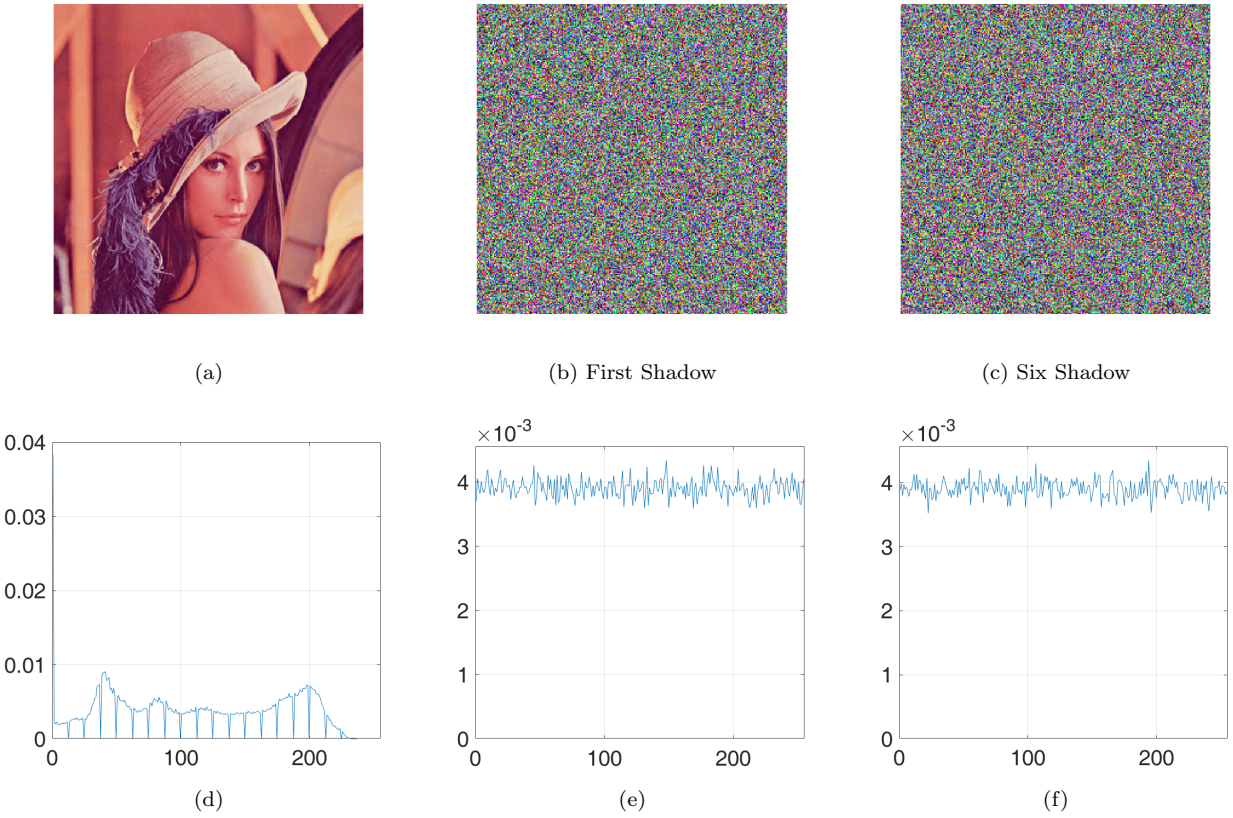
5 Security Analysis

The security analysis of the proposed scheme is based on the methodology presented in [7,51]. In fact, an efficient cryptographic solution should protect data against the most known attack types of confidentiality, integrity and availability. On the other hand, the employed message authentication should be secure and efficient. This part focuses on analyzing the produced encoded fragments to prove resistance against statistical, differential, chosen/known plain-text, and brute-force attacks [3]. Extensive experiments are performed to demonstrate the robustness of the proposed scheme against these attacks. In the following, randomness tests are analyzed in details such as correlation between original and encoded fragmented images and uniformity tests such as entropy and Probability Density Function (PDF) tests, which were applied at the fragment level instead of the image level. In addition, the key sensitivity is also tested by computing the difference among the encoded (encrypted) fragments.

All tests were performed using Matlab. The standard original images such as Lenna and peppers with (512×512) pixels are used. In this test, $k = 4$ and $n = 8$. Hence, 8 shadow (encoded fragment in a matrix form) images are produced for each new input image. In addition, the size of each shadow image is $(128 \times 128 \times 3)$ since the size of the original image is $(512 \times 512 \times 3)$, and $k = 4$. To recover the original image, any k shadow images can be used.

5.1 Randomness Tests

The original images have a high spacial redundancy, which should be removed after the encryption-encoding step, if an efficient cryptographic scheme [5,7] is used. Two original plain-images and their corresponding fragmented images (Lenna and Pepper) are shown in Fig. 9, and Fig. 10, respectively.



(a) (b) First Shadow (c) Six Shadow

(d) (e) (f)

Fig. 9: (a) Original Lenna, (b) first shadow of Lenna image, (c) six Shadow of Lenna image, and (d)-(f) its corresponding PDF, respectively

To measure the randomness introduced by the proposed solution, we used the correlation test. This test randomly takes $N = 4,000$ pairs of adjacent pixels from the two plain images and their corresponding fragmented images. The correlation is measured in the horizontal, vertical, and diagonal directions. The correlation coefficient r_{xy} is calculated using the following equations:

$$\rho_{x,y} = \frac{cov(x,y)}{\sqrt{D(x) \times D(y)}} \tag{4}$$

$$\text{where } cov(x,y) = E[\{x - E(x)\}\{y - E(y)\}];$$

$$E(x) = \frac{1}{n} \times \sum_{k=1}^{n} x_i$$

$$\text{and } D(x) = \frac{1}{n} \times \sum_{k=1}^{n} \{x_i - E[x]\}$$

The obtained results for two standard images (Lenna and peppers) at their first shadow images level are presented in Fig. 11, 12 and 13 for 100 random secret session keys. Similar results are obtained for other fragments. The results clearly indicate the high correlation between adjacent pixels in the original images (correlation

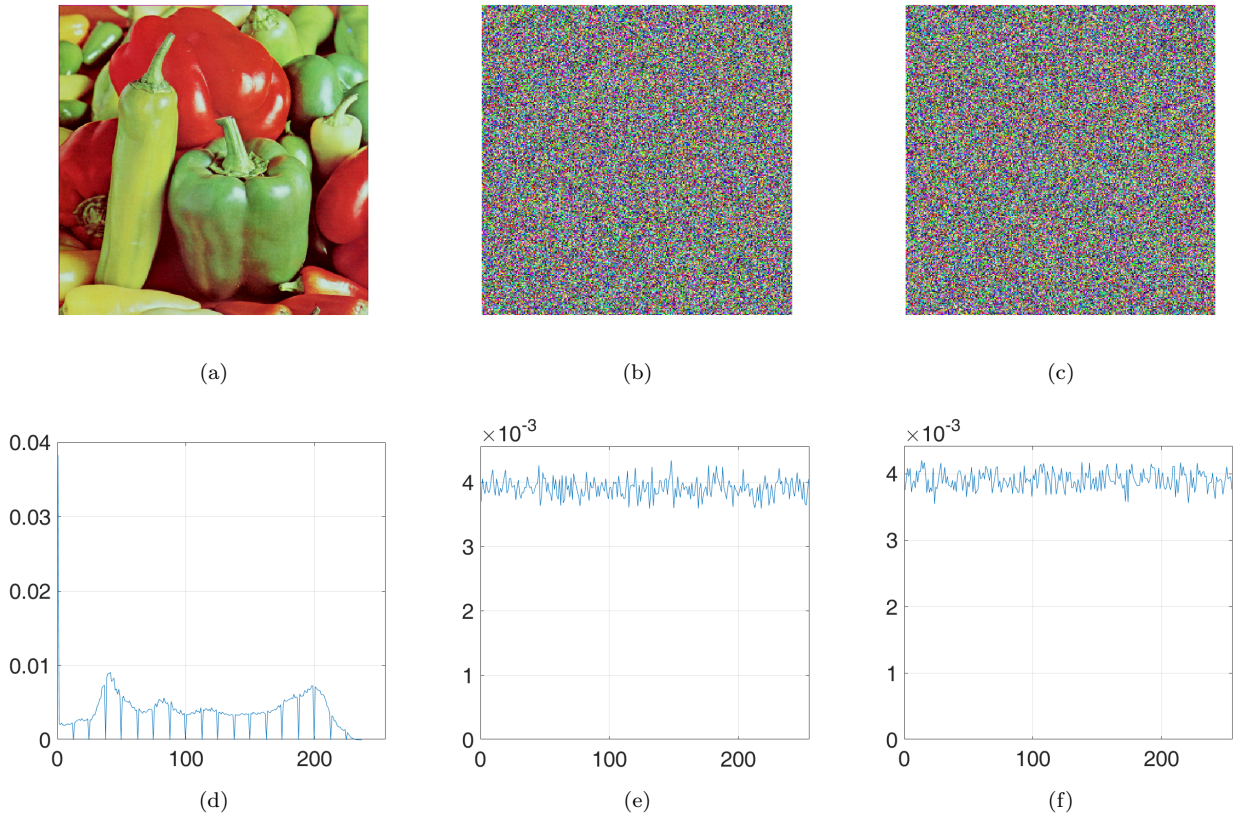


Fig. 10: (a) Original pepper, (b) first shadow of pepper image, (c) six Shadow of pepper image, and (d)-(f) its corresponding PDF, respectively

coefficient close to 1). As for the encoded fragmented shadow images, the correlation coefficient is very low (close to 0), which clearly indicates that the proposed cryptographic scheme reduces noticeably the spatial redundancy. The statistical results are presented in Table 3.

According to the obtained results, the proposed encryption-IDA algorithm ensures a low correlation since the obtained coefficient correlation of adjacent pixels is always close to zero for the different directions. This confirms that no detectable correlation exists in the adjacent pixels of the encoded fragmented parts. Therefore, the proposed scheme results in encrypted-fragmented images with a high level of randomness. In contrast to Rabin's IDA scheme that suffers from the problem of data patterns appearing in the fragmented data [8], the proposed scheme does not preserve distances between encoded data parts (see Fig. 11 and 12), since the dynamic key-dependent approach is used in the different cryptographic processes such as encryption, modified IDA (a set of m IDA matrices instead of static ones), and message authentication.

5.2 **Uniformity Analysis**

To measure the uniformity of encrypted encoded fragments, both the probability density function (PDF) and entropy tests are applied. To resist the common statistical attacks, the encoded fragmented image should have a uniform PDF in addition to a high level of randomness. The uniformity means that each symbol has an occurrence probability close to $\frac{1}{Q}$, where Q is the number of symbols. The PDF of two original plain-images and their corresponding encrypted-fragmented shadow images are shown in figures 9, 10. It can be observed that the PDF of the produced encrypted encoded fragments is close to a uniform distribution, which is $\frac{1}{256}$ at the byte level.

To validate this result at the sub-matrix level with a dimension of $(t \times t)$, an entropy test is performed as indicated in [3]. Each sub-matrix can be considered as a random source with a uniform distribution, if it has an entropy equal or close to $log_2(t^2)$, which is the desired value if $t^2 \leq Q$. The entropy analysis for the sub-matrices

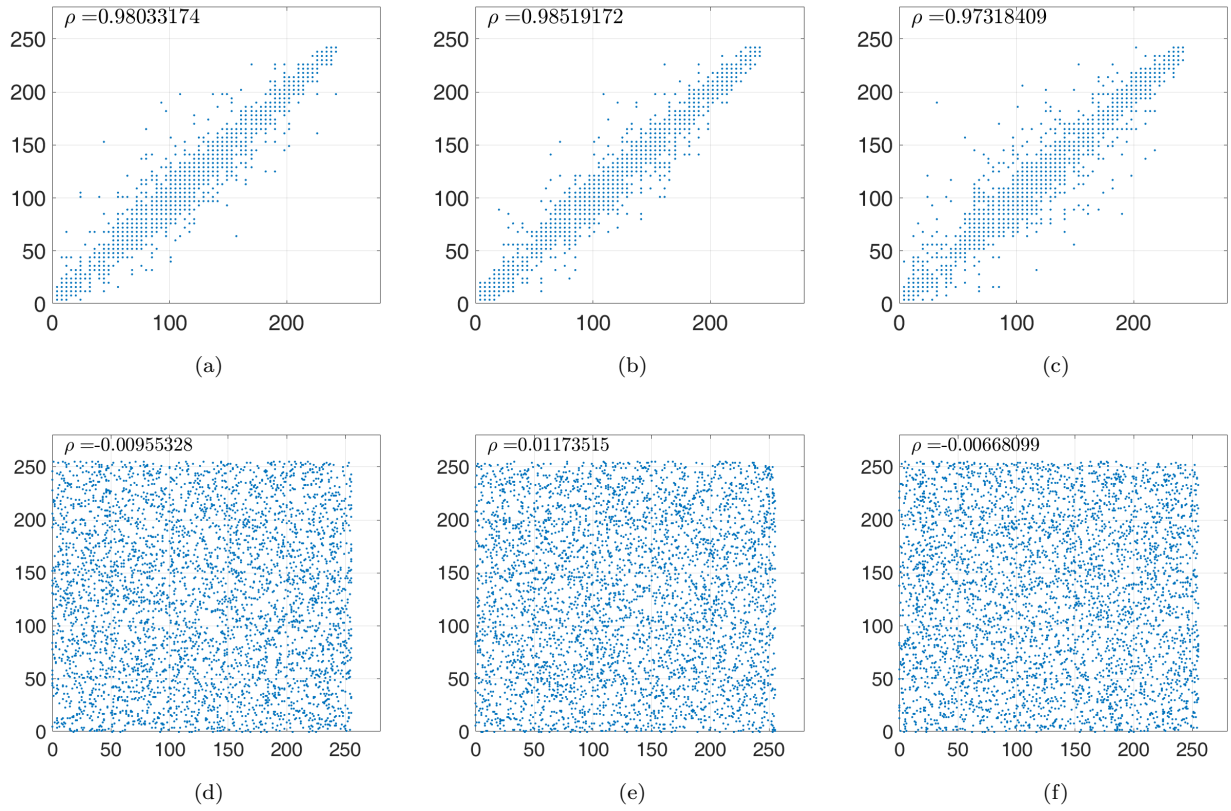


Fig. 11: Correlation in adjacent pixels in original Lenna: (a) horizontally, (b) vertically and (c) diagonally and correlation in adjacent pixels in fragmented Lenna:(d) horizontally, (e) vertically and (f) diagonally

of the original and encoded fragmented Lenna images for $t = 16$ are shown in Fig. 14-a) for the first fragment and its corresponding average versus 100 random keys. Similar results are obtained for other encoded fragments.

The results indicate that the encrypted encoded fragmented sub-matrices have an entropy always close to the desired value, which is 8 in case of $t = 16$. Table 4 shows the values obtained from the entropy analysis for t =4, 8 and 16, respectively. According to the results, the proposed encryption-fragmentation algorithm is sufficiently secure against statistical attacks.

5.3 Independence

Fragmented-encrypted images should be very different from the original ones, and their inter-correlation should be very low. In the following, we present the independence tests results among shadow images (encoded fragments) and the difference between original and encoded fragments.

5.3.1 ***Independence among Shadow Images***

In Table 6, the correlation coefficient among n (here $n = 8$) different encoded fragments is presented in table-view. The results demonstrate a low correlation between the different encoded fragments, which reveals the dissimilarity between the encoded shadow images.

5.3.2 Difference Test

This test is performed to calculate the percentage difference at the bit level between the original and the encoded fragments. A secure cryptographic algorithm should ensure a difference percentage at the bit level close to 50%

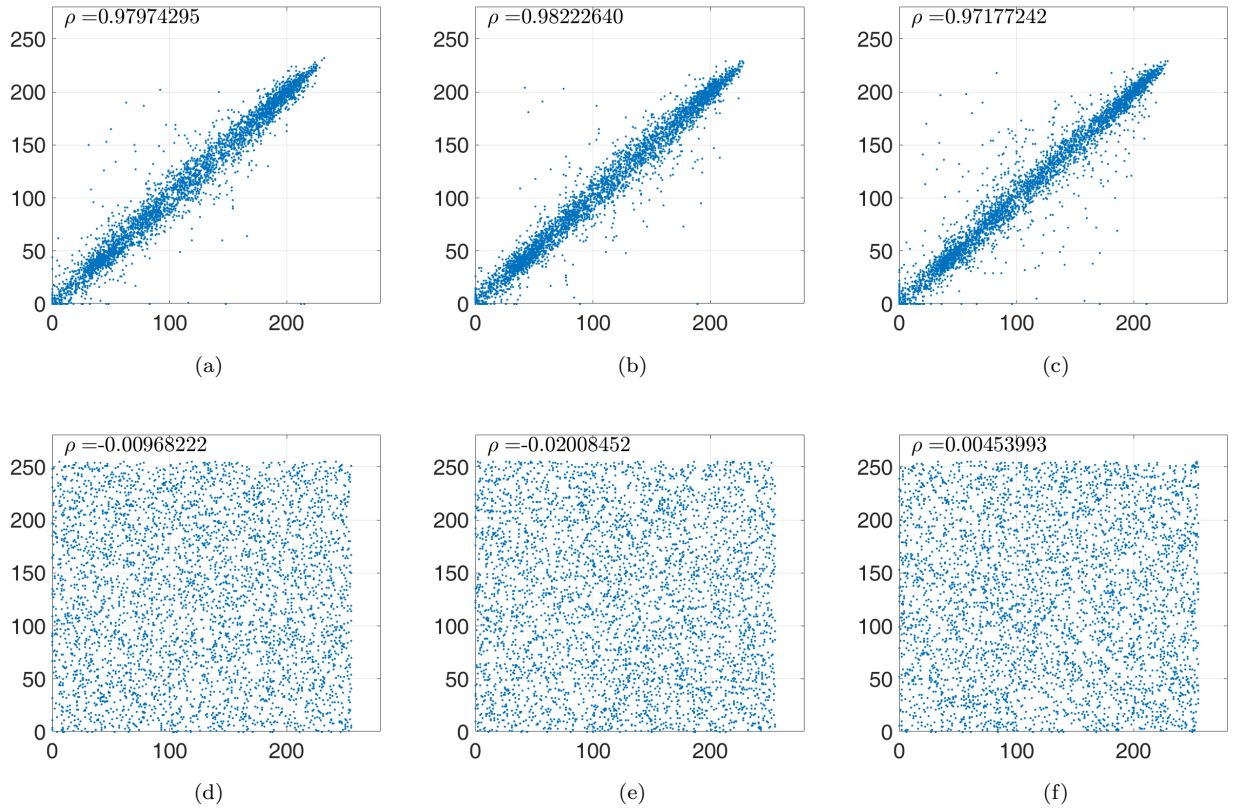
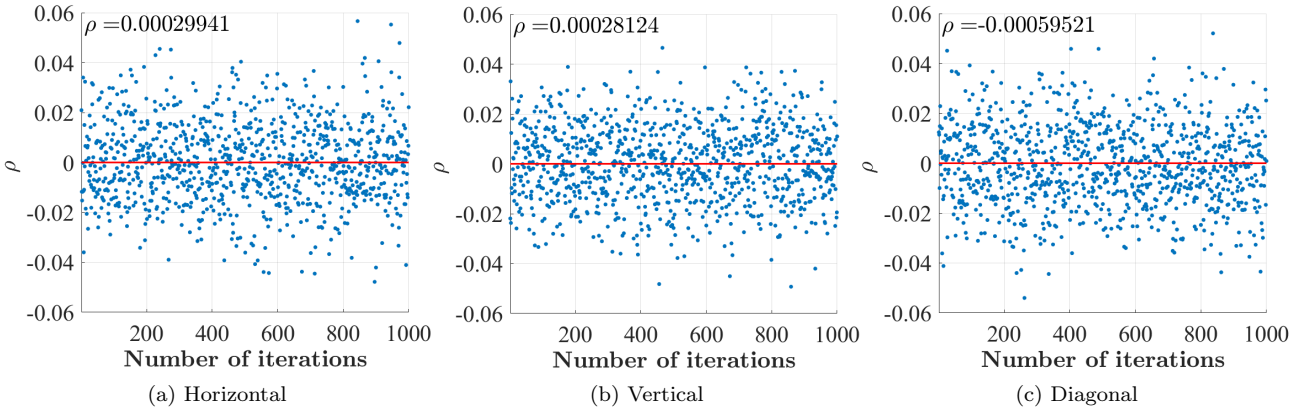


Fig. 12: Correlation in adjacent pixels in original Pepper: (a) horizontally, (b) vertically and (c) diagonally. and correlation in adjacent pixels in one fragmented Pepper image: (d) horizontally, (e) vertically and (f) diagonally.



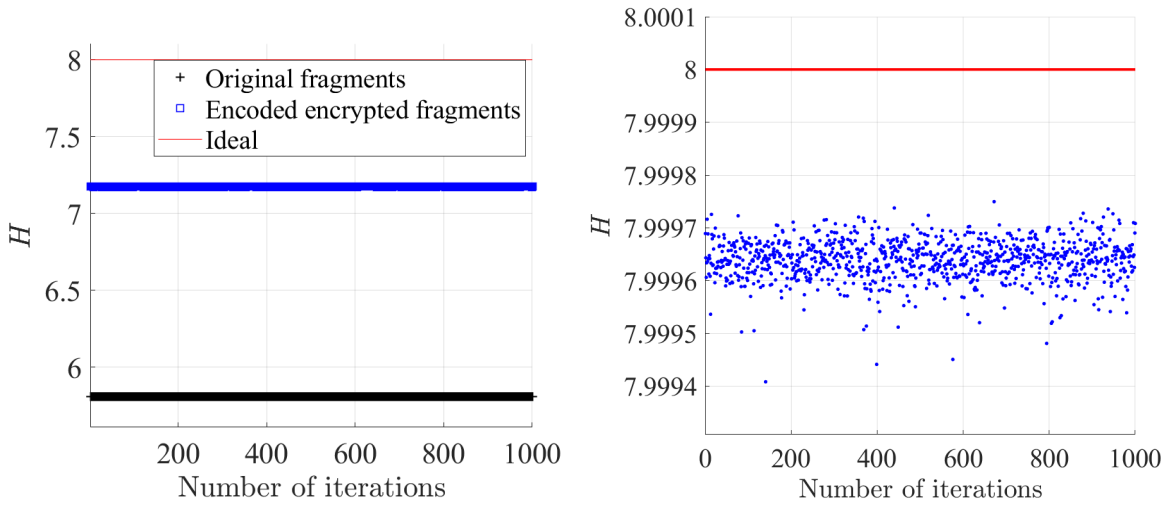
(a) Horizontal (b) Vertical (c) Diagonal

Fig. 13: Variation of the correlation coefficient in adjacent pixels in one fragmented pepper image: (a) horizontally, (b) vertically and (c) diagonally

between the encoded fragments and the original ones. More importantly, the difference test should be applied also among the encoded fragments. We can see in Fig. 15-a) that the proposed scheme achieves 49.998% as average difference between the original and the encoded fragmented data for 1,000 random session keys. Table 5 shows an example of the percentage difference between the k original OIF and the n encoded shadow images (EF) for Lenna image and with a random dynamic key. Similarly, it is required to have up to 50% difference in bits among the encoded fragments. The average difference (without the diagonal part) at the bit level between each couple of

Table 3: The average correlation coefficient r_{xy} of the encrypted fragmented for different standard images under the proposed approach

Encrypted Images	**Average Correlation Coefficient**		
	Horizontal	**Vertical**	**Diagonal**
Lena	0.0029	0.0014	-0.0017
Pepper	-0.0290	-0.0254	-0.0094
Baboon	-0.0134	0.0348	-0.0091
Boat	0.0280	0.0083	-0.0001
Cameraman	-0.0205	-0.0232	0.0030
Fruits	-0.0209	0.0042	-0.0008
Goldhill	-0.0111	0.0147	-0.0122



(a) Sub-matrix level

(b) Fragment level

Fig. 14: The average of entropy at the sub-matrix level (for $t = 16$) of original and first shadow Lena image a) under the use of a random dynamic key and b) the average of entropy of encrypted fragments versus 1000 random keys

the encoded fragments is calculated for 1,000 times and shown in Fig. 15-b). A careful examination of the results indicates that the obtained value is close to the ideal one, and the mean value is 49.93%, which is close to 50% in addition to a low standard deviation of 0.3095. In addition, Table 7 shows an example of the difference percentage for the various couples of the encoded fragmented images for a random session key and the obtained values are very close to 50%. Consequently, the proposed encryption-IDA encoding algorithm ensures the required level of difference between the original and the encoded fragments and among the encoded fragments.

Table 4: Entropy Statistical Tests versus $t.m$ and C represents the original and fragment sub-matrices, respectively

		Min	**Mean**	**Max**	**Std**
t=4	$H(m)$	0	3.1625	4.0000	0.5014
	$H(C)$	3.3750	3.9421	4	0.0828
t=8	$H(m)$	2.1823	4.2014	5.7813	0.7991
	$H(C)$	5.4452	5.7653	5.9688	0.0754
t=16	$H(m)$	2.7235	4.9910	6.8398	0.9624
	$H(C)$	7.0386	7.1754	7.3299	0.0514

Table 5: The percentage difference between the k original F and the n encoded shadow images (EF) for Lenna image and with a random dynamic key (k =4 and n =8)

	EF_1	EF_2	EF_3	EF_4	EF_5	EF_6	EF_7	EF_8
F_1	50.24	49.92	50.07	50.04	49.97	49.87	50.14	50.11
F_2	50.10	49.96	50.08	50.08	49.88	49.92	49.94	50.12
F_3	50.01	49.95	49.99	50.01	50.03	49.91	50.04	50.01
F_4	50.14	49.99	50.03	49.98	49.93	50.01	49.93	50.06

5.4 **Sensitivity Test**

Differential attacks are based on studying the relation between encoded fragmented images obtained with a slight change in the encryption key. Usually, a change of one bit in the original dynamic key should produce different cryptographic and update primitives. A sensitivity test shows how much a slight change in the dynamic key (or after updating cryptographic primitives) will affect the resulted encoded encrypted fragments. In other words, the higher the change in encoding fragments for a slight change of the dynamic key or of the cryptographic primitives, the better the sensitivity of the proposed cryptographic algorithm. Below we analyze different types of sensitivity.

The **plain-text sensitivity** is not considered since the cryptographic primitives are re-generated for each new session and they are updated for each new input image. Consequently, this results in totally different encoded fragment (shadow) images for the same original image. Therefore, the proposed solution successfully satisfies the message avalanche effect, but in a different manner based on the use of the dynamic key-dependent encryption-IDA approach.

Concerning the **key sensitivity** test, it is one of the most important tests and it quantifies the sensitivity against any slight change(s) to the key. In this test, we compute the percentage of change in the encoded fragmented images due to a slight change in the secret session key or after updating the cryptographic primitives. The results should ensure a percentage of sensitivity close to 50% to be considered secure. To study the sensitivity, two dynamic secret keys are used : K_1 and K_2 that differ in only one random bit.

Table 6: Variation of the coefficient correlation among $n = 8$ Lenna shadow images with a random dynamic key

	EF_1	EF_2	EF_3	EF_4	EF_5	EF_6	EF_7	EF_8
EF_1	1	-		-	-	-	-	-
EF_2	0.0021	1	-	-	-	-	-	-
EF_3	0.0047	0.0028	1	-	-	-	-	-
EF_4	-0.0047	0.0024	0.0013	1	-	-	-	-
EF_5	-0.0045	0.0018	0.0068	0.0005	1	-	-	-
EF_6	0.0043	-0.0009	-0.0031	0.0025	-0.0002	1	-	-
EF_7	0.0004	0.0033	0.0024	0.0018	-0.0071	-0.0038	1	-
EF_8	-0.0089	0.0080	0.0014	-0.0017	-0.0101	-0.0002	0.0007	1

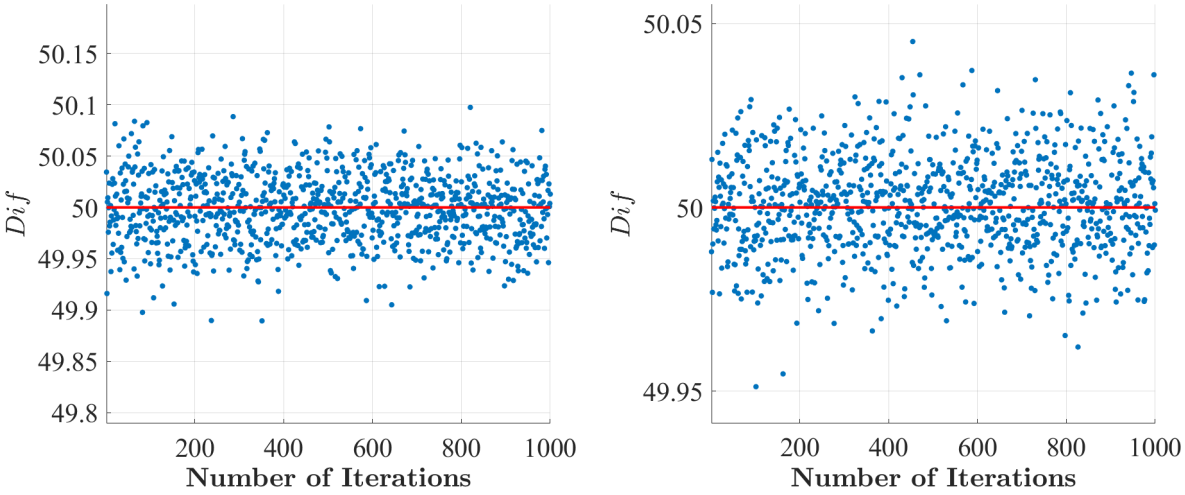
Table 7: The percentage bit difference among Lenna shadows images with k =4 and n =8 with a random dynamic key

	EF_1	EF_2	EF_3	EF_4	EF_5	EF_6	EF_7	EF_8
EF_1	0	-	-	-	-	-	-	-
EF_2	50.02	0	-	-	-	-	-	-
EF_3	50.06	49.91	0	-	-	-	-	-
EF_4	50.06	50.11	49.92	0	-	-	-	-
EF_5	50.10	49.96	49.91	49.91	0	-	-	-
EF_6	49.96	50.02	50.05	49.97	49.99	0	-	-
EF_7	50.02	50.02	49.89	49.94	50.1	50.018	0	-
EF_8	50.09	49.92	49.97	49.99	50.07	50.06	49.85	0

The sensitivity test for the w^{th} key ($K_w^{'}$) is calculated as follows:

$$KS_w = \frac{\sum_{k=1}^{T} FE_{SK_w} \oplus FE_{SK'_w}}{l} \times 100\%, w = 1,\ 2,\ \ldots,\ 1,000. \tag{5}$$

where all the elements of $SK_w^{'}$ are equal to those of SK_w, except for a random Least Significant Bit (LSB) of a random byte. FE represents the proposed cryptographic scheme and $\frac{8\times n\times |M|}{k}$ is the length of the fragmented



(a) Difference between original and fragment parts

(b) Difference between Shadow fragments

Fig. 15: Difference between plain Lenna and shadow fragment Lena (a) and between shadow fragments(b) for 1000 random keys

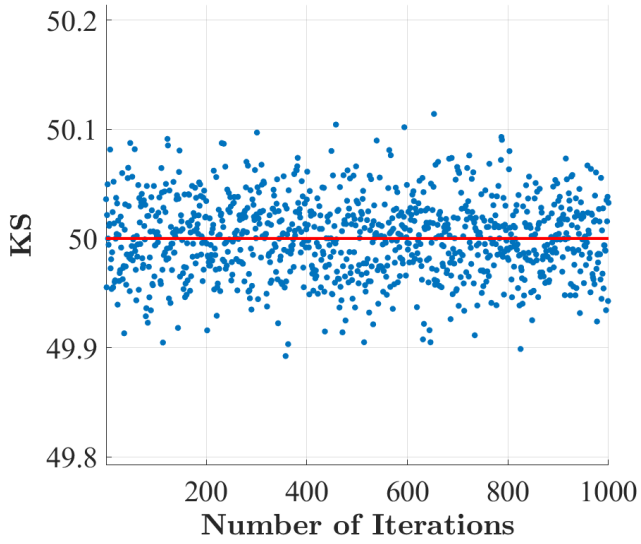


Fig. 16: key sensitivity variation against 1000 random dynamic keys by using Lenna image

data (in bits). In Fig. 16, the KS test is done for 1,000 iterations; the mean value is 49.99%, very close to 50% and the standard deviation equals to 0.3128, which is low. In addition, a numerical result of Key sensitivity test between two sets of shadow images, obtained for Lenna image with two slightly different dynamic keys with k =4 and n =8, is presented in Table 8. The results confirm that the proposed cryptographic solution achieves the required key sensitivity level, ensuring a high resistance degree against different attack types.

The statistical results are presented in Table 9 to prove that the proposed solution ensures the independence between original and shadow images and also the independence among shadow images in addition to the high key sensitivity. The mean of the difference (independence) and the key sensitivity test results are close to the ideal value of 50%. The obtained standard deviation values are close to 0.

Table 8: Key sensitivity test between two obtained shadow images sets obtained for Lenna image but with two slightly different keys (Dk and DK') with k =4 and n =8

	EF_1	EF_2	EF_3	EF_4	EF_5	EF_6	EF_7	EF_8
EF'_1	50.096	49.91	49.91	49.98	50.04	50.02	50.02	49.94
EF'_2	50.06	50.08	49.95	49.97	50.06	50.05	49.96	49.97
EF'_3	49.88	50.03	50.05	49.97	49.94	49.92	50.08	49.98
EF'_4	49.99	50.02	50.02	49.92	49.96	50.01	50.07	49.98
EF'_5	49.92	50.04	50.038	49.95	49.91	50.0578	50.01	50.0492
EF'_6	50.01	50.04	50.04	50.05	50.09	49.90	49.99	49.99
EF'_7	49.99	49.99	50.01	50.13	50.01	50.01	50.01	50.03
EF'_8	49.98	50.03	49.96	50.06	49.99	50.01	50.06	50.10

Table 9: Statistical results of sensitivity tests

	Min	**Mean**	**Max**	**Std**
Dif between original and shadow fragments	49.86	49.97	50.11	0.036
Dif between among shadow fragments	49.95	50.01	50.05	0.013
KS	49.87	49.97	50.10	0.0338

5.5 **Visual Degradation**

The degradation of the original image must be done in a way that the visual content presented in the shadow images (encoded fragments) cannot be recognized. One well known parameter is considered to measure the fragmentation's visual quality, the Structural Similarity index (SSIM) [53], [54]. The perceived quality of the image by the human eye is highly dependent on the loss of structural information in the image. The SSIM value lies in the interval [0,1]. A value of 0 means that there is no correlation between the original and the fragmented image [5], while a value close to 1 means that the two images are approximately the same.

We computed the $SSIM$ index for the original and fragmented Lena image for $1,000$ pseudo-random seeds and the results are presented in Fig. 17 and Table 11. Table 10 shows the $SSIM$ variation between the k original and the n shadow fragments for the Lenna image with a random dynamic key (k =4 and n =8). As shown, the $SSIM$ value has a maximum value of 0.0414, indicating that a high and hard visual distortion is achieved using the proposed cryptographic scheme. This validates the big difference between the original and the shadow images, and that no useful information could be revealed about the original image from the fragmented ones.

In summary, the security analysis results show that the proposed solution ensures a high level of randomness as presented in Section 5.1. The uniformity and independence of the encoded fragments as compared to the

Table 10: $SSIM$ variation between the k original OIF and the n shadow images (EF) for Lenna image with a random dynamic key (k =4 and n =8)

	EF_1	EF_2	EF_3	EF_4	EF_5	EF_6	EF_7	EF_8
F_1	0.0005	-0.0003	-0.0001	-0.0001	0.0001	0.0005	-0.0014	-0.0002
F_2	-0.0001	-0.0003	-0.0009	-0.0001	0.0005	0.0005	-0.0015	0.0005
F_3	0.0003	-0.0015	-0.0010	0.0004	0.0011	-0.0005	-0.0016	0.0007
F_4	0.0011	-0.0016	0.0001	0.0002	-0.0001	-0.0015	-0.0008	0.0002

Table 11: Statistical results of visual degradation tests for k =4 and $n = 8$ for 1000 times

	Min	**Mean**	**Max**	**Std**
PSNR	7.8744	8.315	8.4241	0.0421
SSIM	-0.0013	0.0002	0.0017	0.0005

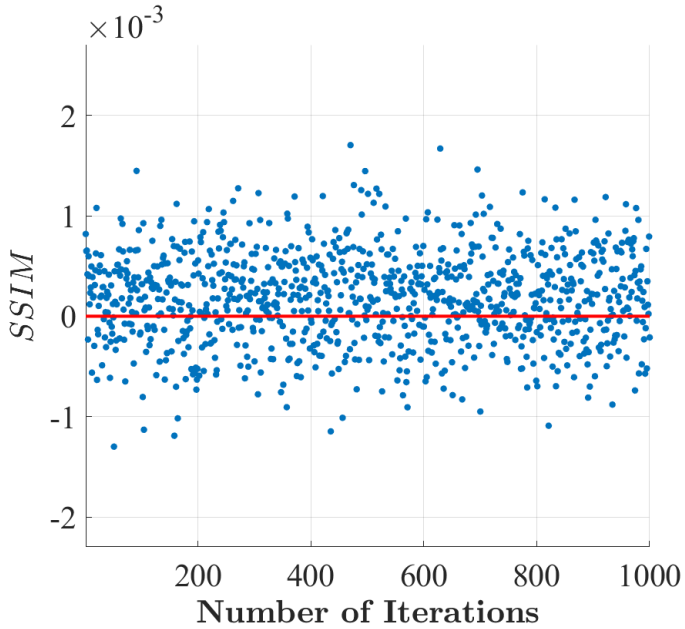


Fig. 17: The variation of $SSIM$ average between the original and the shadow Lenna image versus 1000 random key with $n = 8$

original ones are shown in Section 5.2 and 5.3, respectively. Moreover, in Section 5.4, the sensitivity test of the proposed scheme is performed and the results indicate that the solution exhibits a high level of key sensitivity.

6 Cryptanalysis Discussion

The cryptographic strength of the proposed solution relies on the use of variable cryptographic primitives in a pseudo-random manner. A complete knowledge of the encoded fragments of a certain image does not permit the recovery of the previous or next image since for each input image, new cryptographic primitives are used. Also, for each session, a new dynamic key is produced using a one-way cryptographic hash function, which prevents the recovery of the master key. This guarantees the backward secrecy and forward security.

Moreover, the permutation cipher algorithm is applied before the fragmentation process to increase the randomness of the fragments. The modified IDA applies the encoding process at the column level (k bytes) and not on the whole matrix ($k \times \frac{|M|}{k}$), which makes parallel processing possible and efficient. Also, the modified IDA scheme uses a set of dynamic matrices instead of just one, and for each block (column), one of the produced IDA matrices is selected according to a dynamic selection table.

In the considered scenario, n encoded fragments are authenticated and then dispersed over n different storage entities. Thus, the original data protection relies first on the difficulty of collecting k encoded fragments, out of the n dispersed ones, and finding the correct session key, which has sufficient length to make it unfeasible. An attacker needs to know the location of the fragments before accessing them. Also, the attacker must first guess or obtain the right order of fragments, the bank of matrices along with their corresponding order, and the encryption permutation table in order to recover the original image. This means that the knowledge of k encoded fragments alone does not reveal any useful information to the attacker.

In the following, some of the most known confidentiality attacks (statistical, differential, chosen/known plain-test, and brute-force) are discussed in a situation where k encoded fragments have been revealed to an attacker. Also, the proposed cryptographic scheme is considered to be known to the attacker.

6.1 Statistical Attacks:

This category of attacks exploits the fact that the encoded fragments may reveal some statistical properties. Therefore, in an ideal situation, the frequency analysis of data within a fragment should be indistinguishable from the output of a random generator. Previous statistical tests (entropy analysis, probability density function, correlation tests) have confirmed the robustness of the scheme against statistical attacks.

6.2 Brute-force Attacks:

The size of the master secret key can be 128, 196 or 256 bits, while the size of the dynamic key is 512 bits. Therefore, the size of the master secret and dynamic session keys are large enough to make brute force attacks unfeasible.

6.3 Linear and Differential Attacks: Known and Chosen Plain/Cipher Text Attacks

A different set of cryptographic primitives are used for each input image in order to protect the data against the known or chosen plain-text attack types. In fact, the dynamic cryptographic primitives are updated for each new input image, which eliminates any similarity among the obtained shadow images for the same original image. Section 5.4 shows clearly that a high key sensitivity is reached with the proposed solution; a single bit change in the secret key or update cryptographic primitives generates different cryptographic primitives and consequently, different shadows images.

The sensitivity analysis confirms the efficiency of the proposed cryptographic algorithm against key-related attacks [55] since a key derivation function is used to produce dynamic cryptographic primitives and the corresponding update primitives. These results show that no useful information could be detected from the fragmented shadow images where all pixels of the fragmented image are changed.

We performed cryptanalytic tests (available in the literature) to analyze the capability of the proposed solution against several cryptanalytic attacks. The proposed fragmentation-encryption algorithm is considered to be

public, and the cryptanalyst has complete knowledge of all operations, but has no knowledge about the secret key and the nonce. The proposed scheme is based on a variable dynamic key and update mechanism for the cryptographic primitives, for every input image. Accordingly, the problems associated with single image failure and accidental seed disclosure are avoided by this scheme.

On the other hand, the value of $(n-k)$ is selected according to the required availability level, which depends on the importance of the data. A higher value of $(n-k)$ is required for highly sensitive images to provide a high availability level. However, this introduces a cost in terms of communication and storage overhead $(n-k \times \frac{|M|}{k})$.

Finally, the resistance against data integrity and message authentication is related to the employed message authentication algorithm. The use of variable cryptographic primitives produces different encoded fragments for the same input image and different associated MACs.

7 Performance Analysis

The proposed approach is based on dividing the encrypted image into k fragments where the size of k and n have to meet the trade-off between performance and security levels. On the other hand, the principal advantage of the proposed approach is that the modified IDA scheme can be performed in parallel, which leads to a reduction in the execution time, if parallel computing is possible.

Note that, a low value of $(n-k)$ requires a low execution time, but it offers a low availability level. On the other hand, a high value of $(n-k)$ increases both, the execution time and the availability degree. Hence, one has to select the value of $(n-k)$ according to the application data classification level. In Fig. 18, different values of n and k were used for a file of size 256 MBytes. The variation of execution time versus different values of n is expressed as a linear function, and increasing k leads to an increase in the execution time.

7.1 Theoretical Performance

The required computational complexity and storage overhead of the proposed scheme are compared against other relevant state-of-the-art works such as Shamir's secret sharing, IDAs, and AONT-RS, as shown in Table 12. A precise evaluation is hard due to the variety of the implementations. The cost of an IDA is equivalent to the multiplication of the data by a $Matrix$ of size $(n \times k)$. The performance of AONT depends on the chosen encryption and hash algorithms, as well as the data size.

Moreover, the proposed encryption scheme is based on the dynamic key-dependent approach and it requires a single round and a single permutation operation, instead of multiple rounds and operations in the case of standard ciphers such as AES. Similarly, the selected keyed hash function can be also based on a single round [41]. Consequently, the required computational complexity for the data confidentiality and integrity, in addition to source authentication, is lower than the one required for AONT using standard cryptographic algorithms.

The data is fragmented into n fragments and to $(\frac{M}{k \times TB})$ blocks. A matrix multiplication operation is required for each data fragment (block). Therefore, the proposed modified IDA can strongly benefit from this parallelization since each data fragment can be encoded or recovered independently from the others. The fragmentation (defragmentation) process ensures better parallelization using a different thread for each computation compared to the original IDA.

7.2 Storage/Communication Overhead

The size of the produced fragments is close to the optimal value of $(|M|/k)$. Therefore, the fragmentation procedure presented in Fig. 4 does not incur any data overhead and preserves the benefits of the original IDA. The storage overhead $[(n-k) \times \frac{|M|}{k}]$ is due only to the redundant fragments, which is inevitable for preventing data loss in case of damage or alteration.

Table 12: *Running time and storage requirements*

Scheme	**Running time**	**Storage Requirement**
SSS	Poly(n, k, $\|M\|$)	$n \times \|M\|$
IDA	Matrix(n, k)	$\frac{n \times \|M\|}{k}$
AONT-RS	AONT($\|M\|$) + RS($n-k$, k,d)	$\frac{n \times (\|M\|+w)}{k}$
Our proposal	$ns\times$ Matrix (1, k)/number of Threads	$n \times \frac{\|M\|}{k}$

7.3 Propagation of Errors

An important criteria for any cryptographic solution is the low error propagation property. The interference and noise in the transmission channel (or in the storage system) are the main cause of errors. A bit error refers to the substitution of a '0' bit by a '1' bit and vice versa. This error may propagate and result into the destruction of data; this is a big challenge since there is a trade-off between the avalanche effect and error propagation [56, 7]. In this proposal, if a bit error takes place in any of the encoded sub-fragmented shadow image, and it will affect only the corresponding decoded sub-fragment (block of k bytes). Therefore, error propagation is limited to the sub-fragment level.

7.4 **Execution Times**

The following experiments were performed using a C code on an Intel(R) Xeon(R) Silver 4110 CPU @ 2.10 GHz machine with 64GB of RAM. In Table 13, the execution times are reported for two different situations. In the first column, the size of the file is given. In columns 2 to 4, parameters (k, n) and the execution times are presented such that the number of fragments (n) is fixed and k varies (4, 8, 12). It can be observed that the execution time does not vary significantly when k varies. In columns 5 to 7, we consider the scenario where n varies (8, 16, 24) and the number of fragments for the recovery (k) is equal to ($\frac{n+1}{4}$). It can be seen that when k increases, the execution time also increases.

In Fig. 18, the execution times are reported for a file of size 256MB. It can be seen that when n increases, the execution time also increases. Moreover, k is not so significant since when it doubles, the execution time slightly increases. In both experiments, the algorithm proves to be very efficient; it performs the splitting of files into fragments very quickly. Consequently, the solution is ready to be used in real implementations.

8 Conclusion and Future Work

In this paper, a dynamic key-dependent cryptographic solution is presented for data confidentiality, integrity, availability, and message authentication. The robustness of the solution is based on the dynamic key-dependent cryptographic approach such that different cryptographic primitives are changed for each new input image, and not just for each new session. The efficiency is ensured by using simple operations with a low number of rounds, leading to a fast execution time for encryption and authentication of fragments. Further optimization of the data availability scheme can be achieved using look-up tables instead of the matrix multiplication operation. In addition, the authentication-encryption scheme can be realized in parallel. Several security analysis tests were conducted to prove the high security level. As a conclusion, due to its flexibility, high security, low computational complexity, the proposed solution can be considered as a competitive cryptographic solution for securing image contents.

Table 13: Execution times for different sizes of messages, total number of fragments, and number of fragments for recovery

Size of messages	k	n	Execution Time (s)	k	n	Execution Time (s)
16KB	4	16	0.00086	3	8	0.00068
16KB	8	16	0.00096	5	16	0.00086
16KB	12	16	0.0011	7	24	0.0011
64KB	4	16	0.003	3	8	0.0019
64KB	8	16	0.003	5	16	0.003
64KB	12	16	0.0036	7	24	0.0038
256KB	4	16	0.0093	3	8	0.0047
256KB	8	16	0.011	5	16	0.0098
256KB	12	16	0.012	7	24	0.012
1MB	4	16	0.019	3	8	0.01
1MB	8	16	0.023	5	16	0.022
1MB	12	16	0.027	7	24	0.028
4MB	4	16	0.061	3	8	0.038
4MB	8	16	0.066	5	16	0.062
4MB	12	16	0.073	7	24	0.086
16MB	4	16	0.23	3	8	0.13
16MB	8	16	0.24	5	16	0.22
16MB	12	16	0.27	7	24	0.31
64MB	4	16	0.87	3	8	0.51
64MB	8	16	0.98	5	16	0.9
64MB	12	16	1.07	7	24	1.21
256GB	4	16	3.48	3	8	2.02
256GB	8	16	4.01	5	16	3.5
256GB	12	16	4.3	7	24	4.85

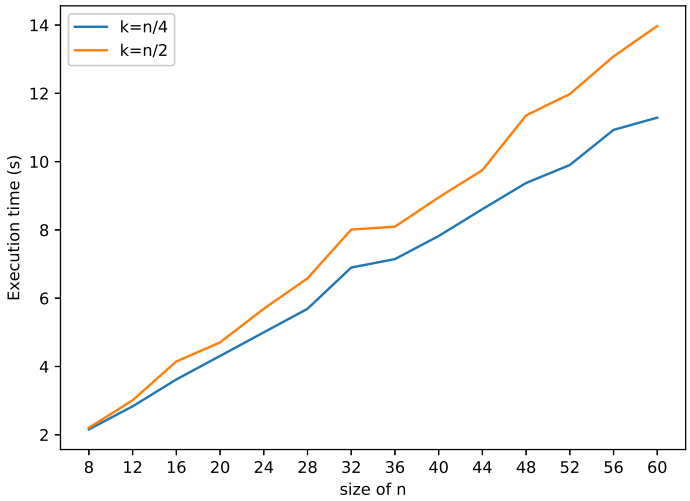


Fig. 18: Executions times for different number of fragments n for a file of size 256MB

Acknowledgement

This paper is partially supported with funds from the Maroun Semaan Faculty of Engineering and Architecture at the American University of Beirut and also from the EIPHI Graduate School (contract "ANR-17-EURE-0002"). We also thank the supercomputer facilities of the Mésocentre de calcul de Franche-Comté.

References

1. Ahmed Mostefaoui, Hassan Noura, and Zeinab Fawaz. An integrated multimedia data reduction and content confidentiality approach for limited networked devices. *Ad Hoc Networks*, 32:81–97, 2015.

2. Hassan Noura, Lama Sleem, Mohamad Noura, Mohammad M Mansour, Ali Chehab, and Raphaël Couturier. A new efficient lightweight and secure image cipher scheme. *Multimedia Tools and Applications*, pages 1–28, 2017.
3. Hassan Noura, Ali Chehab, Lama Sleem, Mohamad Noura, Raphaël Couturier, and Mohammad M Mansour. One round cipher algorithm for multimedia iot devices. *Multimedia Tools and Applications*, pages 1–31, 2018.
4. Joan Daemen and Vincent Rijmen. *The design of Rijndael: AES-the advanced encryption standard*. Springer Science & Business Media, 2013.
5. Hassan N Noura, Mohamad Noura, Ali Chehab, Mohammad M Mansour, and Raphaël Couturier. Efficient and secure cipher scheme for multimedia contents. *Multimedia Tools and Applications*, pages 1–30, 2018.
6. Hassan Noura, Reem Melki, Ali Chehab, Mohammad M Mansour, and Steven Martin. Efficient and secure physical encryption scheme for low-power wireless m2m devices. In *2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC)*, pages 1267–1272. IEEE, 2018.
7. Hassan Noura, Ali Chehab, Mohamad Noura, Raphaël Couturier, and Mohammad M Mansour. Lightweight, dynamic and efficient image encryption scheme. *Multimedia Tools and Applications*, pages 1–35, 2018.
8. Katarzyna Kapusta, Gerard Memmi, and Hassan Noura. Secure and resilient scheme for data protection in unattended wireless sensor networks. In *Cyber Security in Networking Conference (CSNet), 2017 1st*, pages 1–8. IEEE, 2017.
9. Katarzyna Kapusta, Gerard Memmi, and Hassan Noura. Poster: A keyless efficient algorithm for data protection by means of fragmentation. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1745–1747. ACM, 2016.
10. Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.
11. Jason K. Resch and James S. Plank. Aont-rs: Blending security and performance in dispersed storage systems. In *Proceedings of the 9th USENIX Conference on File and Stroage Technologies*, FAST'11, pages 14–14, Berkeley, CA, USA, 2011. USENIX Association.
12. Hugo Krawczyk. Secret sharing made short. In *Proceedings of the 13th Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '93, pages 136–146, London, UK, 1994. Springer-Verlag.
13. Michael O. Rabin. Efficient dispersal of information for security, load balancing, and fault tolerance. *J. ACM*, 36(2):335–348, April 1989.
14. P. Cincilla, A. Boudguiga, M. Hadji, and A. Kaiser. Light blind: Why encrypt if you can share? In *2015 12th International Joint Conference on e-Business and Telecommunications (ICETE)*, volume 04, pages 361–368, July 2015.
15. Jean-Charles Fabre, Yves Deswarte, and Brian Randell. *Designing secure and reliable applications using fragmentation-redundancy-scattering: an object-oriented approach*, pages 21–38. Springer Berlin Heidelberg, Berlin, Heidelberg, 1994.
16. GR Blakley. Safeguarding cryptographic keys. In *afips*, page 313. IEEE, 1899.
17. Moni Naor and Adi Shamir. Visual cryptography. In *Workshop on the Theory and Application of of Cryptographic Techniques*, pages 1–12. Springer, 1994.
18. Chih-Ching Thien and Ja-Chen Lin. Secret image sharing. *Computers & Graphics*, 26(5):765–770, 2002.
19. Stelvio Cimato and Ching-Nung Yang. *Visual cryptography and secret image sharing*. CRC press, 2017.
20. Kai-Hui Lee and Pei-Ling Chiu. Digital image sharing by diverse image media. *IEEE transactions on information forensics and security*, 9(1):88–98, 2014.
21. Shih-Chieh Wei, Young-Chang Hou, and Yen-Chun Lu. A technique for sharing a digital image. *Computer Standards & Interfaces*, 40:53–61, 2015.
22. M Naor and A Shamir. Visual cryptography [j/ol]. *Lecture Notes in Computer Science*, 950(1):1–12, 2017.
23. Chang-Chou Lin and Wen-Hsiang Tsai. Visual cryptography for gray-level images by dithering techniques. *Pattern Recognition Letters*, 24(1-3):349–358, 2003.
24. Eric R Verheul and Henk CA Van Tilborg. Constructions and properties of k out of n visual secret sharing schemes. *Designs, Codes and Cryptography*, 11(2):179–196, 1997.
25. Young-Chang Hou. Visual cryptography for color images. *Pattern recognition*, 36(7):1619–1629, 2003.
26. Shyong Jian Shyu. Efficient visual secret sharing scheme for color images. *Pattern Recognition*, 39(5):866–880, 2006.
27. Ching-Nung Yang and Chi-Sung Laih. New colored visual secret sharing schemes. *Designs, Codes and cryptography*, 20(3):325–336, 2000.
28. Carlo Blundo, Annalisa De Bonis, and Alfredo De Santis. Improved schemes for visual cryptography. *Designs, Codes and Cryptography*, 24(3):255–278, 2001.
29. Long Bao, Shuang Yi, and Yicong Zhou. Combination of sharing matrix and image encryption for lossless (k, n)-secret image sharing. *IEEE Transactions on Image Processing*, 26(12):5618–5631, 2017.
30. Xiaotian Wu, Jian Weng, and WeiQi Yan. Adopting secret sharing for reversible data hiding in encrypted images. *Signal Processing*, 143:269–281, 2018.
31. Xuehu Yan, Shen Wang, Ahmed A Abd El-Latif, and Xiamu Niu. Visual secret sharing based on random grids with abilities of and and xor lossless recovery. *Multimedia Tools and Applications*, 74(9):3231–3252, 2015.
32. Ching-Nung Yang and Sin-Ming Huang. Constructions and properties of k out of n scalable secret image sharing. *Optics Communications*, 283(9):1750–1762, 2010.
33. Rastislav Lukac and Konstantinos N Plataniotis. Bit-level based secret sharing for image encryption. *Pattern recognition*, 38(5):767–772, 2005.
34. K Shankar, Mohamed Elhoseny, R Satheesh Kumar, SK Lakshmanaprabu, and Xiaohui Yuan. Secret image sharing scheme with encrypted shadow images using optimal homomorphic encryption technique. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–13, 2018.
35. Runhua Shi, Hong Zhong, Liusheng Huang, and Yonglong Luo. A (t, n) secret sharing scheme for image encryption. In *2008 Congress on Image and Signal Processing*, volume 3, pages 3–6. IEEE, 2008.
36. Guzin Ulutas, Mustafa Ulutas, and Vasif V Nabiyev. Secret image sharing scheme with adaptive authentication strength. *Pattern Recognition Letters*, 34(3):283–291, 2013.
37. Cheng Guo, Chin-Chen Chang, and Chuan Qin. A hierarchical threshold secret image sharing. *Pattern Recognition Letters*, 33(1):83–91, 2012.
38. Peng Li, Ching-Nung Yang, Chih-Cheng Wu, Qian Kong, and Yanpeng Ma. Essential secret image sharing scheme with different importance of shadows. *Journal of Visual Communication and Image Representation*, 24(7):1106–1114, 2013.

39. Amir M Ahmadian and Maryam Amirmazlaghani. A novel secret image sharing with steganography scheme utilizing optimal asymmetric encryption padding and information dispersal algorithms. *Signal Processing: Image Communication*, 74:78–88, 2019.
40. Anonymized. Poster: A keyless efficient algorithm for data protection by means of fragmentation. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pages 1745–1747, New York, NY, USA, 2016. ACM.
41. Hassan Noura, Soran Hussein, Steven Martin, Lila Boukhatem, and Khaldoun Al Agha. Erdia: An efficient and robust data integrity algorithm for mobile and wireless networks. In *2015 IEEE wireless communications and networking conference (WCNC)*, pages 2103–2108. IEEE, 2015.
42. Christof Paar and Jan Pelzl. *Understanding Cryptography: A Textbook for Students and Practitioners*. Springer Publishing Company, Incorporated, 1st edition, 2009.
43. Mingqiang Li. On the confidentiality of information dispersal algorithms and their erasure codes. *CoRR*, abs/1206.4123, 2012.
44. Elaine B Barker and John Michael Kelsey. *Recommendation for random number generation using deterministic random bit generators (revised)*. US Department of Commerce, Technology Administration, National Institute of Standards and Technology, Computer Security Division, Information Technology Laboratory, 2011.
45. Ronald L. Rivest. All-or-nothing encryption and the package transform. In *In Fast Software Encryption, LNCS*, pages 210–218. Springer-Verlag, 1997.
46. William Stallings. *Cryptography and network security: principles and practice*. Pearson Upper Saddle River, NJ, 2017.
47. Hassan N Noura, Ali Chehab, and Raphael Couturier. Efficient & secure cipher scheme with dynamic key-dependent mode of operation. *Signal Processing: Image Communication*, 78:448–464, 2019.
48. Hassan Noura and Damien Couroussé. Lightweight, dynamic, and flexible cipher scheme for wireless and mobile networks. In *International Conference on Ad Hoc Networks*, pages 225–236. Springer, 2015.
49. Goutam Paul and Subhamoy Maitra. *RC4 stream cipher and its variants*. CRC press, 2011.
50. H. Noura, S. Hussein, S. Martin, L. Boukhatem, and K. A. Agha. Erdia: An efficient and robust data integrity algorithm for mobile and wireless networks. In *2015 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 2103–2108, March 2015.
51. Hassan Noura, Steven Martin, Khaldoun Al Agha, and Khaled Chahine. Erss-rlnc: Efficient and robust secure scheme for random linear network coding. *Computer Networks*, 75, Part A:99 – 112, 2014.
52. Katarzyna Kapusta, Gerard Memmi, and Hassan Noura. Secure and resilient scheme for data protection in unattended wireless sensor networks. In *1st Cyber Security in Networking Conference, CSNet 2017, Rio de Janeiro, Brazil, October 18-20, 2017*, pages 1–8, 2017.
53. Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *Image Processing, IEEE Transactions on*, 13(4):600–612, 2004.
54. Alain Hore and Djemel Ziou. Image quality metrics: Psnr vs. ssim. In *Pattern recognition (icpr), 2010 20th international conference on*, pages 2366–2369. IEEE, 2010.
55. Ashutosh Dhar Dwivedi, Paweł Morawiecki, Rajani Singh, and Shalini Dhar. Differential-linear and related key cryptanalysis of round-reduced scream. *Information Processing Letters*, 136:5–8, 2018.
56. Ayoub Massoudi, Frédéric Lefebvre, Christophe De Vleeschouwer, Benoit Macq, and J-J Quisquater. Overview on selective encryption of image and video: challenges and perspectives. *Eurasip Journal on information security*, 2008(1):179290, 2008.