SPCBC: A Secure Parallel Cipher Block Chaining Mode of Operation based on logistic Chaotic Map

Aly M. El-Semary[1,2,*], Mohamed Mostafa A. Azim[1,3] and Hossam Diab[1,4]
[1] College of Computer Science and Engineering, Taibah University - KSA
[e-mail: {aelsemary, mzayed, hdiab }@taibahu.edu.sa]
[2] Al-Azhar University, Faculty of Engineering, Cairo - Egypt
[alyelsemary@{azhar.edu.eg, ieee.org}]
[3] Beni-Suef University, Faculty of industrial Education, Beni-Suef - Egypt
[e-mail: mmazim@ieee.org]
[4] Menoufia University, Faculty Of Science, Menoufia - Egypt
[e-mail: dr.hosamdiab@gmail.com]
*Corresponding author: Aly M. El-semary

Received November 2, 2016; revised March 16, 2017; accepted April 17, 2017; published July 31, 2017

Abstract

Several block cipher modes of operation have been proposed in the literature to protect sensitive information. However, different security analysis models have been presented for attacking them. The analysis indicated that most of the current modes of operation are vulnerable to several attacks such as known plaintext and chosen plaintext/cipher-text attacks. Therefore, this paper proposes a secure block cipher mode of operation to thwart such attacks. In general, the proposed mode combines one-time chain keys with each plaintext before its encryption. The challenge of the proposed mode is the generation of the chain keys. The proposed mode employs the logistic map together with a *nonce* to dynamically generate a unique set of chain keys for every plaintext. Utilizing the logistic map assures the dynamic behavior while employing the *nonce* guarantees the uniqueness of the chain keys even if the same message is encrypted again. In this way, the proposed mode called SPCBC can resist the most powerful attacks including the known plaintext and chosen plaintext/cipher-text attacks. In addition, the SPCBC mode improves encryption time performance through supporting parallelized implementation. Finally, the security analysis and experimental results demonstrate that the proposed mode is robust compared to the current modes of operation.

Keywords: Block cipher modes of operation, cryptography, computer security, network security, one-time key

1. Introduction

In cyptography, block cipher algorithms are used to encrypt a message or block of plaintext with a fixed-length [1]. If the message or plaintext length is greater than the algorithm's block size, the message should be partitioned into a number of blocks with equal size. If the message size does not fit into an integer multiple of the block size, extra characters are padded to the last block of the message using one of the available padding methods such as PKSC \#5 [2]. Then, one of the block cipher modes of operation is used to provide the required confidentiality and/or authenticity.

Several standard block cipher modes of operation have been introduced. These standard modes include electronic code book (ECB), cipher block chaining (CBC), cipher feedback (CFB), output feedback (OFB), and counter (CTR). These modes can be utilized with any block cipher to cover a variety of applications for encryption purposes [3-5]. These modes provide confidentiality but they are prone to malicious modification which can be discovered with separate authentication code. This requires that both confidentiality and authenticity modes be combined together to provide secrecy and authenticity. However, the integration of two modes could be hard and error prone. To overcome these shortcomings, several block cipher modes were recently developed to integrate both confidentiality and authenticity into a single mode. Modes providing both confidentality and authenticity are reffered to as authenticated encryption modes. These modes include TinySec, Counter with CBC-MAC mode (CCM), CYPHER-C3, and Counter Chain (CC).

The TinySec [6] supports two services namely, authenticated encryption referred to as TinySec-AE and authentication only denoted by TinySec-Auth. In this mode of operation, the encryption is achieved by the Skipjack algorithm together with the CBC mode of operation while the authentication is supported by the CBC mode of operation. The CCM mode [7] has two phases to achieve both authentication and confidentiality. Specifically, in the authentication phase, it employs the CBC mode to generate a message authentication code (MAC) for data integrity. In the confidentiality phase, it uses the CTR mode for data encryption. The CYPHER-C3 [8] is a block cipher mode of operation that provides authenticated encryption for packet-based communications networks. The authors argued that their proposed CYPHER-C3 achieves improvements in processing energy requirement, processing latency and packet throughput when compared to CCM mode and TinySec-AE mode. Finally, the CC mode [9] is authenticated encryption that integrates the CTR mode with the CBC mode. It achieves the encryption speed of the CTR mode while providing better security than the CBC mode in terms of protection against attacks associated with initial vector. Surveys on cipher block modes of operation can be found in [10-12].

The aforementioned block cipher modes of operation provided confidentiality and/or authenticity. They also attempted to improve the encryption performance and protect against security attacks. However, they are prone to several attacks such as known plaintext and chosen plaintext/cipher-text attacks. In particular, several attack models [13-15] have been developed to break the security of these modes. Wu [13] presented a related-cipher attack model that is applicable only to cipher-text encrypted with similar algorithms differing in the key size and number of rounds. Phan *et al*. [14] generalized the related-cipher attack model to be applied on a larger class of related ciphers. They also showed that when a cryptanalyst has access to an oracle for any mode of operation, then almost all other related cipher modes can be easily attacked. Wang *et al*. [15] used the CTR mode to attack other modes and employed other modes to attack the CTR mode under the related-mode attack model. These attack

models indicated that most of the current block cipher modes of operation are vulnerable to several attacks including known plaintext and chosen plaintext/cipher-text attacks.

Accordingly, in this paper, we are devoted to present a secure block cipher mode of operation that protects against these types of attacks. In particular, we propose a Secure Parallel Cipher Block Chaining (SPCBC) mode of operation to enhance the security of the current modes. The proposed SPCBC mode is an authenticated encryption mode that combines one-time chain keys with plaintext blocks before their encryption to preclude attacks. The key issue in the proposed SPCBC mode is how to generate these chain keys. The SPCBC mode makes use of chaotic maps [16,17] and a *nonce* to generate dynamic and unique chain keys, one for each plaintext block. Specifically, it employs the logistic map implemented in several cryptographic algorithms [18-25] to utilize their desirable features of ergodicity, sensitivity to initial conditions, and control parameters. The logistic chaotic map is designed to initialize its parameters with secret values and then dynamically obtain the chaotic values based on the encrypted plaintext. On the other hand, the uniqueness of the key chains is guaranteed through using the *nonce* which is a unique random value generated for each message. Therefore, the generated chain key has three features: uniqueness, secrecy, and dynamic behavior. These features result in a secret one-time key for each plaintext block even if the same block is encrypted again. This remarkably enhances the security of the proposed SPCBC mode over the current modes of operation. In particular, it effectively resists several attacks including known plaintext and chosen plaintext/cipher-text attacks. Hence, the proposed mode of operation is suitable for applications that require stringent security.

The organization of this paper is as follows. Section 2 presents the background related to logistic map and highlights the merits and demerits of the current related modes of operation. Section 3 introduces the proposed SPCBC mode of operation. Section 4 provides a detailed security analysis of the proposed mode while Section 5 demonstrates the experimental results. Finally, Section 6 concludes the paper.

2. Background

This section presnets a general background about the logistic map and the current related block cipher modes of operation. The logistic map discussed in Section 2.1 is used to generate a chaotic sequence while the merits and demerits of related block cipher modes of operation are introduced in Section 2.2.

2.1 Logistic Map Principles

Chaos theory is a prominent theory related to the study and analysis of nonlinear dynamic systems. Chaotic systems have several outstanding features including sensitivity to initial parameters, topologically mixing, deterministic, ergodicity, and noise-like behaviour [16-18]. Based on these chaotic systems, a pseudo-random iterative bounded sequence referred to as *chaotic sequence* is genereted. Therefore, with the produced chotic sequence, a slightly different initial parameter/condition of the system results in a completely different random sequence with non-periodic and non-convergent traits. These brilliant features make the chaotic systems potential candidates for encryption applications. One of the simplest chaotic functions that have been recently investigated for security applications is the logistic map [19-26]. The logistic map can be defined as in Equation 1.

$$x(\eta) = \mu\, x(\eta - 1)[1 - x(\eta - 1)] \tag{1}$$

Where the intial value x (0) belongs to (0, 1) and η referes to the number of iterations while μ is a control or bifurcation parameter of the logistic map. The behavior of the map totally depends on the value of the bifurcation parameter [21, 25] which can be viewed in three intervals: $\mu \in$ (0, 3], $\mu \in$ (3, 3.57), and $\mu \in$ [3.57, 4]. The dynamic behavior of the logistic map in these ranges is illustrated in **Fig. 1** through developing a Matlab program and executing it with three different values for the parameter μ: μ = 2.70, μ = 3.05 and μ = 3.90, a value for each interval. The program is executed for each μ value using the initial value x (0) = 0.2 and number of iterations η = 100. In the first interval (e.g., μ = 2.70), the logistic map always converges to one fixed point as shown in **Fig. 1a**. In the second case (e.g., μ = 3.05), the logistic map emerges periodic behavior where the map oscillates between some fixed points as displayed in **Fig. 1b**. In the last interval (e.g., μ = 3.90), the map exhibits a good chaotic behavior in which the periodic phenomena vanish as depicted in **Fig. 1c**. Also, the bifurcation diagram of logistic map which describes the output distribution of the map along its control parameter μ is visulized in **Fig. 1d**. The bifurcation diagram proves that the logistic function operates in chaotic behavior when $\mu \in$ [3.57,4]. In other words, the logistic map in this range of the bifurcation parameter generates a sequence that is extremely sensitive to the initial parameters with a good random distribution and ideal pseudorandom characteristics.

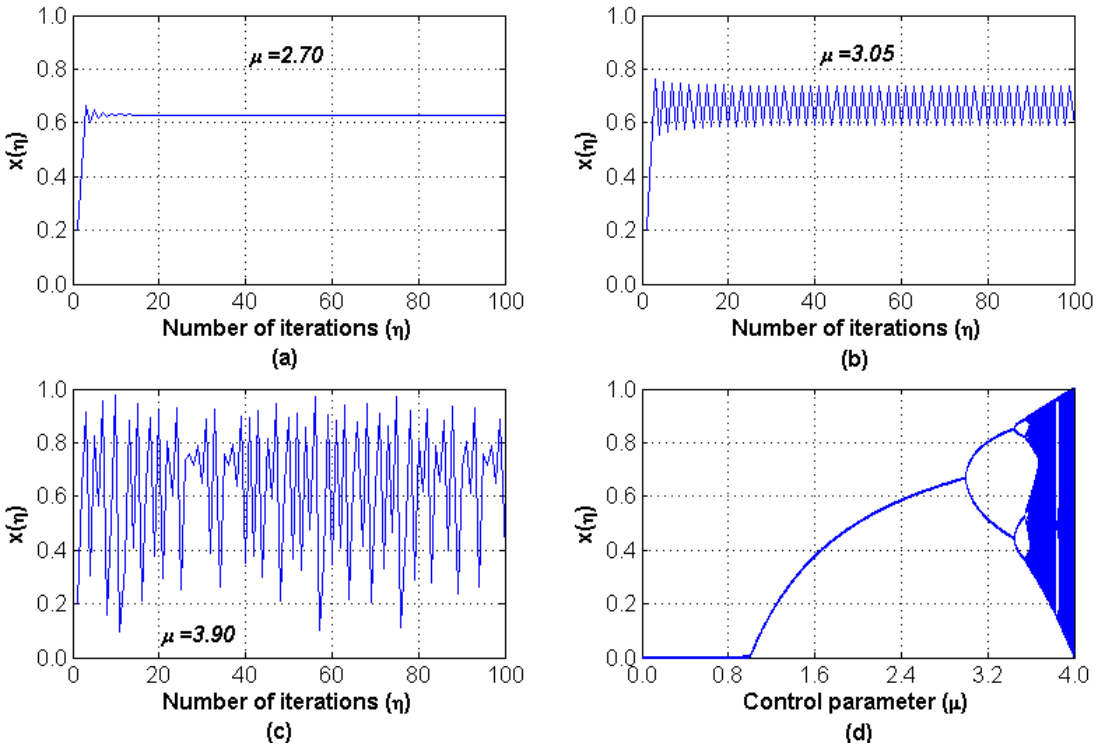


Fig. 1. Logistic map behavior and its bifurcation in three regions of control parameter μ.

2.2 Related Block Cipher Modes of Operation

This section discusses briefly the advantages and disadvantages of the current block cipher modes of operation related to the proposed SPCBC mode. Specifically, these modes include the CBC, CTR, CCM, and CC modes of operation. A detailed evaluation of several block cipher modes of operation is found in [27].

The CBC mode is a single-pass mode that provides both confidentiality and authenticity. It uses an initial vector (*IV*) to afford chain dependency between cipher-text blocks (i.e., the encryption of any plaintext block depends on the cipher-text resulted from the previous plaintext block except for the encryption of the first plaintext block which depends on the *IV*). Even though the CBC mode provides chain dependency and message integrity along with message confidentiality, it has several drawbacks. The CBC mode produces the same cipher-text for repeated messages because it uses the same *IV* for all messages. In addition, the CBC mode cannot encrypt a plaintext in parallel due to the chain dependency. Moreover, the CBC mode propagates errors in one cipher-text block into its next cipher-text block. In other words, 1-bit error in a cipher-text block will affect the corresponding plaintext block along with the subsequent block. Finally, the CBC mode is vulnerable to different types of attacks such as chosen plaintext/cipher-text attacks [13-15].

The CTR mode of operation is a single-pass mode that uses a preconfigured counter to generate a key stream. The key stream is exclusive-ORed with the plaintext to produce the corresponding cipher-text. The CTR mode has several advantages including overcoming the error propagation problem of the CBC mode. In addition, the CTR mode produces a cipher-text with exactly the same size as the corresponding plaintext. This is because it does not require any padding of the plaintext to match the block size of the encryption algorithm. Finally, the CTR mode provides highly parallelizable architecture that makes it attractive to high-speed applications. In contrast, the CTR mode does not provide message authentication [3]. Moreover, the CTR mode is vulnerable to chosen plaintext/cipher-text attacks [15].

The CCM mode of operation is a two-pass mode that combines the CTR mode and raw CBC-MAC to provide authenticated encryption. Specifically, the CBC-MAC is responsible for message authentication while the CTR mode is concerned with data encryption [7]. Even though the CCM mode efficiently achieves authenticated encryption, it has several shortcommings including working only with 128-bit block cipher algorithms such as AES. In addition, the CCM is not appropriate for high-speed applications since the CBC-MAC can neither be pipelined nor parallelized. Moreover, it could suffer reduced performance if it disrupts word-alignment. Furthermore, the CCM requires identifying the length of the plaintext before starting the encryption process. Finally, the CCM introduces computational cost twice as much as the cost resulted from a single-pass authenticated encryption mode such as CBC [27].

Finally, the CC mode increases the encryption speed of the CBC by acheiveing a parallelizable architecture to allow parallel encryption of a plaintext. In addition, the CC produces different cipher-texts for the same message through implementing a unique nonce. Furthermore, the CC mode can provide message confidentiality and authentication. On the other hand, the CC mode does not employ dynamic key that may facilitate a chosen plaintext/cipher-text attack.

3. SPCBC Mode of Operation

The proposed secure parallel cipher block chaining (SPCBC) mode of operation supports both confidentiality and authentication. The SPCBC mode is developed to enhance the security over the current modes. It strengthens the security through combining a one-time chain key with each plaintext block before its encryption. The SPCBC mode also improves the performance through supporting parallelism achieved by dividing the plaintext into two or more parts called processes. Each part can be encrypted independently. The most challenging issue in the SPCBC mode is the generation of the chain keys. During the generation of the

chain keys, the SPCBC mode utilizes the features of logistic map together with a *nonce* to assure the dynamic behavior and the uniqueness of the generated chain keys. Actually, the logistic map is initialized with secret values and then iterated based on encrypted plaintext to guarantee both secrecy and dynamic behavior of chain keys. On the other hand, the *nonce* is a unique random number generated for each message to ensure a unique set of key chains for each message. In this way, it is guaranteed to generate a dynamic and unique secret chain key for each block even if the same block is encrypted again. Specifically, the SPCBC mode has two phases applied on each block to enhance the security. The first phase generates a unique chain key while the second phase is responsible for encrypting the message resulted from combining the plaintext block and its related chain key. Before proceeding with the details of the proposed mode, it is importatnt to declare that we adobt the same assumption used in [28] in which the authors assume that the CCM mode utilizes one of the secure automated key management techniques.

The rest of this section discusses in more details how the SPCBC encrypts a plaintext, generates and verifies message authentication code (MAC), and decrypts a cipher-text in Sections 3.1, 3.2, and 3.3, respectively. Finally, the generation of chain keys in SPCBC mode is presented in Section 3.4 while generating secret values based on chaotic maps is introduced in Section 3.5.

3.1 Encryption in SPCBC Mode

In the SPCBC mode, a plaintext M is divided into a sequence of l plaintext blocks: $M_1M_2 \ldots M_l$. The length of each block denoted by S depends on the deployed encryption algorithm and M_l is padded if necessary. For example, if the DES or AES encryption algorithm is used, the block size S will be 64 or 128 bits, respectively. To provide parallelism to the SPCBC mode, the plaintext blocks: $M_1M_2 \ldots M_l$ are divided into a sequence of processes: $p_1p_2 \ldots p_t$, where t refers to the number of processes as shown in **Fig. 2**. Each process p_j has a sequence of n plaintext blocks such that $1 \leq j < t$ while the last process p_t has the last n_t plaintext blocks, where n and n_t are obtained by Equations 2 and 3, respectively.

$$n = \left\lceil \frac{l}{t} \right\rceil \tag{2}$$

$$n_t = l - n\,(t-1) \tag{3}$$

where n is calculated as the ceiling of the number of plaintext blocks l divided by the number of processes t.

Accordingly, the first n plaintext blocks are assigned to the first process p_1, the second n plaintext blocks are allocated to the second process p_2, and so on until the last set of plaintext blocks of length n_t are assigned to the last process p_t. This is depicted in **Fig. 2** in which the plaintext blocks M_1 to M_n are allocated to p_1, M_{n+1} to M_{2n} are assigned to p_2, and so on until reaching $M_{(t-1)n}$. Finally, $M_{(t-1)n+1}$ to M_l are assigned to p_t. Then each process can start its encryption independently of the other processes according to Equation 4.

$$C_{i,j} = E_k(k_{i,j} \oplus M_{i,j}) \tag{4}$$

Where E_k is a symmetric block cipher with the key k. $M_{i,j}$ is the i^{th} plaintext block in the j^{th} process. $k_{i,j}$ is the chain key generated for the block $M_{i,j}$ and it can be calculated by Equation 8. Finally, $C_{i,j}$ is the i^{th} cipher-text block in the j^{th} process and it corresponds to the plaintext block $M_{i,j}$. The $C_{i,j}$ is calculated as the symmetric encryption of the XORed of $k_{i,j}$ and $M_{i,j}$. In other words, the $M_{i,j}$ and $C_{i,j}$ refer to the plaintext block $M_{(j-1)n+i}$ in the whole plaintext and the corresponding cipher-text block $C_{(j-1)n+i}$ in the whole cipher-text where n is the number of blocks in each process. To illustrate Equation 4, suppose n is equal to 100 and the 3rd plaintext block in the 2nd process needs to be encrypted. In this case, $C_{3,2} = E_k (k_{3,2} \oplus M_{3,2})$ or $C_{103} = E_k (k_{103} \oplus M_{103})$.

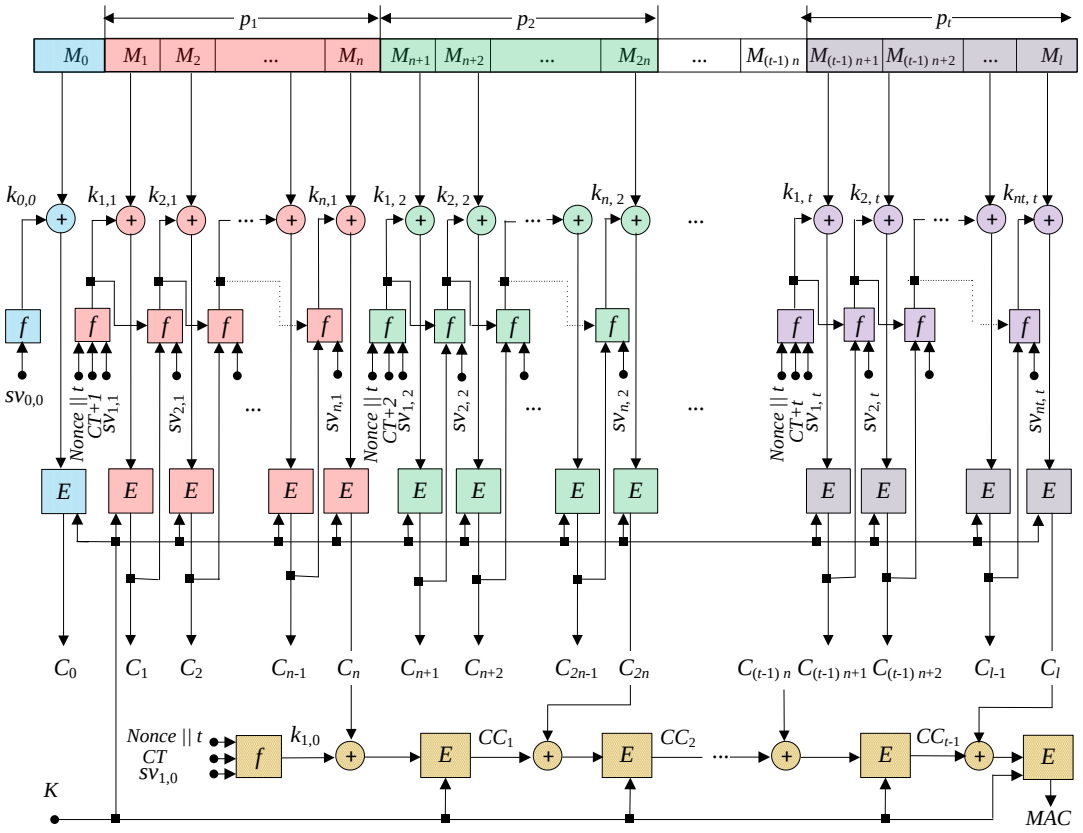


Fig. 2. SPCBC encryption with message authentication code.

After finishing the encryption phase of the whole plaintext, the SPCBC mode has two other tasks. The first task is to obtain the message authentication code (MAC) of the plaintext that will be discussed in more details in Section 3.2. The second task is to create and then encrypt the plaintext block number zero M_0. The block M_0 is created according to the format shown in **Fig. 3**. The size of M_0 denoted by $S = |M_0|$ is equal to the block size of the encryption algorithm used (e.g., S is 64 or 128 bits for DES or AES; respectively) as depicted in the upper part of **Fig. 3**. The format divides M_0 into three parts to accommodate three secret variables initiated by the sender: *nonce*, *t*, and *CT*. The *nonce* is a unique random number for each message and it has the size of S-32 bits of the block. The parameter t refers to the number of processes used in the encryption and its length is five bits long to accommodate up to 32 processes. Finally, the parameter *CT* is a counter that has 27 bits that are enough to hold a counter value up to 2^{27}-1.

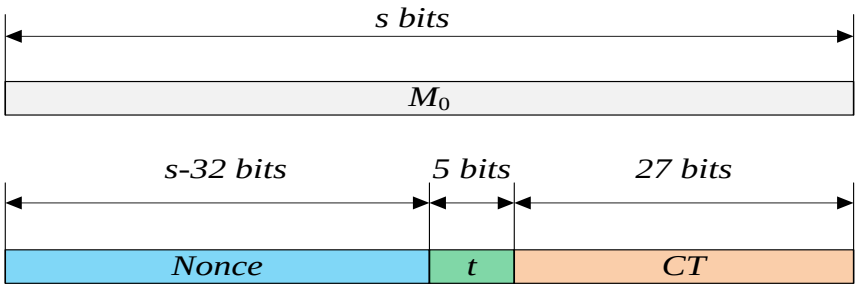


Fig. 3. The formats of plaintext block M_0.

Once M_0 is created, it should be encrypted to get its corresponding cipher-text block C_0 which is equal to E_k ($k_{0,0} \oplus M_0$), where $k_{0,0}$ that can be obtained by Equation 8 is the chain key generated for the plaintext block zero (i.e., M_0). After obtaining both C_0 and *MAC*, they are combined with the cipher-text such that C_0 is inserted at the beginning of the whole cipher-text while the *MAC* is injected at its end (i.e., the whole cipher-text will be $C_0C_1C_{2...}C_l MAC$). The size of the whole cipher-text produced by the SPCBC is equal to the size of the plaintext, $M_1M_{2...}M_l$, plus the size of two extra blocks, C_0 and *MAC*. The C_0 contains information about the *nonce*, the number of processes *t*, and the counter *CT*. On the other hand, the *MAC* is a message authentication code to provide message integrity. In other words, the size of the cipher-text produced by encrypting a plaintext in the SPCBC mode is equal to the number of plaintext blocks plus two extra blocks referred to as C_0 and *MAC* blocks. Note that the size of *MAC* is equal to the block size of the employed block cipher (e.g., 64 or 128 bits for DES or AES, respectively).

3.2. MAC in SPCBC Mode

The SPCBC mode not only provides message confidentiality but also supports authentication through a message authentication code (*MAC*). In SPCBC, the *MAC* is obtained through two phases: local and global authentications. The local authentication phase is achieved by each process while the global authentication is done by a master process p_0 that distributes the work among individual processes. The local authentication phase is performed through generating a local message authentication code $LMAC_j$ for each process p_j, where $1 \leq j \leq t$. Since the encryption of each plaintext block in each process p_j depends on the previous plaintext blocks in the same process, the last cipher-text block in p_j is considered as the $LMAC_j$ of the process p_j. In other words, if the number of plaintext blocks in each process is *n* as shown in **Fig. 2**, then the cipher-text blocks *n*, 2*n*, …, and *l* will correspond to the local message authentication codes $LMAC_1$, $LMAC_2$, …, and $LMAC_t$; respectively. Note that, $LMAC_1$ is the local MAC generated for p_1, $LMAC_2$ is the local *MAC* for p_2 and so on. On the other hand, the global authentication completes the authentication by generating a *MAC* as a function of the associated $LMAC_1$, $LMAC_2$, …, and $LMAC_t$. The *MAC* as depicted in **Fig. 4** is obtained by encrypting the message resulted from XORing C_l with the CC_{t-1} according to Equation 5.

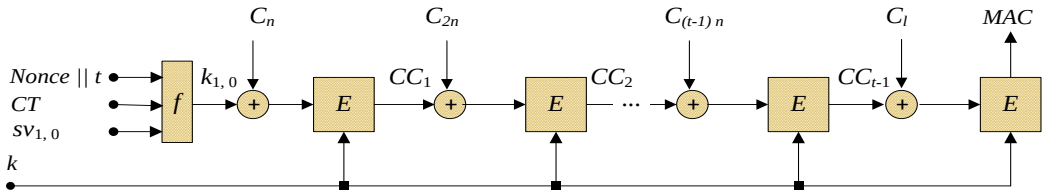
$$MAC = E_k(C_l \oplus CC_{t-1}) \tag{5}$$

Where CC_{t-1} is resulted from the encryption of the cipher-text block $C_{(t-1)n}$ XORed with CC_{t-2} as shown in **Fig. 4a**. In order to find the *MAC* according to Equation 5, it is essential to obtain the value of CC_{t-1} which in turn needs to get the value of CC_{t-2} and so on until reaching the

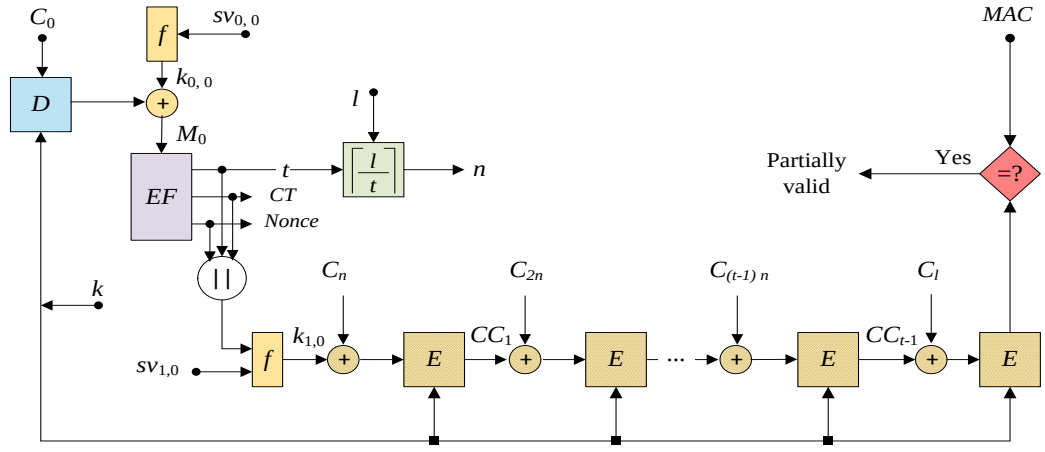
value of CC_1. Therefore, the iterative Equation 6 is constructed to obtain the value of CC_i, where i = 1 to t-1.

$$CC_i = \begin{cases} E_k(C_{i\times n} \oplus k_{1,0}) & if\ i=1 \\ E_k(C_{i\times n} \oplus CC_{i-1}) & if\ 2 \le i \le t-1 \end{cases} \quad (6)$$

Where $k_{1,0}$ is a chain key generated for the cipher-text block C_n and it can be obtained by Equation 8. Finally, the *MAC* verification step starts by extracting C_0 and *MAC* from the cipher-text. Then, M_0 should be recovered from C_0 by XORing $k_{0,0}$ and the decryption of C_0 (i.e., $M_0 = k_{0,0} \oplus D_k(C_0)$). After obtaining the value of M_0, the *nonce*, *t*, and *CT* are extracted by an extraction function called *EF* as shown in **Fig. 4b**. From the value of *t* and the number of received cipher-text blocks *l*, the value of *n* can be calculated according to Equation 2. Next, the *MAC* of the received cipher-text is evaluated according to Equations 5. Then, both the evaluated *MAC* and the received one are compared. If both *MAC*s are matched, then the global authentication is valid. Otherwise, the cipher-text is ignored. This verification process is described in **Fig. 4b** in more details.



a) MAC generation.



b) MAC verification.

Fig. 4. Generating and verifying message authentication code globally in SPCBC mode.

3.3 Decryption in SPCBC Mode

After verifying the global *MAC*, the cipher-text is divided into a sequence of *t* processes, each of *n* cipher-text blocks except the last process that has n_t cipher-text blocks as depicted in **Fig. 5**. The value of *n* and n_t can be calculated as indicated earlier from Equations 2 and 3,

respectively. After associating the corresponding cipher-text blocks to its process p_j, the process p_j can decrypt its cipher-text blocks using Equation 7.

$$M_{i,j} = D_k(C_{i,j}) \oplus k_{i,j} \tag{7}$$

Where $C_{i,j}$ and $M_{i,j}$ are the i^{th} cipher-text block and its corresponding plaintext block in the j^{th} process, respectively. In addition, the $k_{i,j}$ obtained from Equation 8 is the chain key generated for the i^{th} cipher-text block in the j^{th} process. To demonstrate the decryption in SPCBC mode, suppose the value of n is equal to 100 as noted before and the cipher-text block $C_{5,2}$ (the fifth cipher-text block in the second process) is the one to be decrypted. This corresponds to the cipher-text block $C_{(2-1)100+5}$ (i.e., C_{105}) in the cipher-text. Therefore, the corresponding plaintext block is M_{105} which is equal to the XOR of the D_k (C_{105}) and the chain key $k_{5,2}$ (k_{105}).

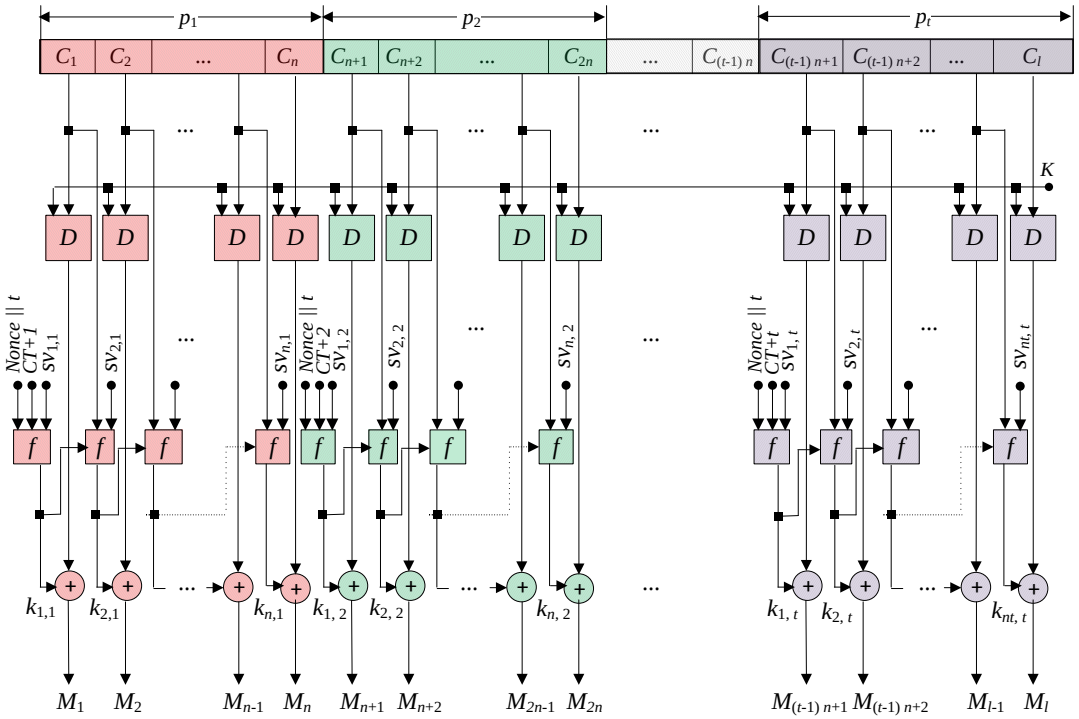


Fig. 5. Decryption of formatted cipher-text in SPCBC mode.

3.4 Chain Key Generation in SPCBC Mode

The proposed SPCBC mode generates a number of dynamic and unique secret chain keys to secure the plaintext and deviate the related attacks such as known plaintext and chosen plaintext/cipher-text attacks. Specifically, it generates two chain keys, $k_{0,0}$ and $k_{1,0}$. In addition, it generates a set of chain keys $k_{r,q}$, one for each block r in the process p_q such that $1 \leq r \leq n$ and $q = 1, 2, \ldots, t$; where t and n are the number of processes and plaintext blocks in each process; respectively. The $k_{0,0}$ is combined with the plaintext block M_0 while $k_{1,0}$ is combined with cipher-text block C_n (i.e., the last cipher-text block resulted from the first process p_1) to produce the *MAC*. As a result, Equation 8 is constructed to generate the required chain keys.

$$k_{i,j} = \begin{cases} f(sv_{i,j}, 0, 0, bs) & if\ i = 0 \\ f(nonce \,\|\, t, sv_{i,j}, CT + j, bs) & if\ i = 1 \\ f(k_{(i-1),j}, sv_{i,j}, C_{(i-1),j}, bs) & if\ i > 1 \end{cases} \quad (8)$$

where $0 \le i \le n$ and $j = 0, 1, 2, \ldots, t$ and $sv_{i,j}$ is a secret value generated for the i^{th} plaintext block in the j^{th} process (i.e., the plaintext block $M_{(j-1)n+i}$ in the whole plaintext). The $sv_{i,j}$ can be obtained from Equation 11 while the function $f(\,)$ is defined by Equation 9.

$$f(x, y, z, bs) = T_{bs}(S_{256}(x \,\|\, y \,\|\, z)) \quad (9)$$

Where "x || y || z" is the concatenation of x, y, and z while $S_{256}(\,)$ is the secure hash algorithm that digests its input message to produce a corresponding hash value of 256 bits. Finally, the function T_{bs} is a truncation function that produces the most significant *bs* bits of its input. The value of *bs* depends on the block size of the symmetric algorithm used in the encryption. For example, if DES or AES is used in the encryption, then the value of *bs* will be 64 or 128; respectively. Therefore, the function $f(\,)$ produces the most significant *bs* bits of the hash value resulted from the $S_{256}(\,)$. Note that the secure hash algorithm with 256 bits is used rather than using a 128-bit algorithm to accommodate any symmetric encryption algorithms with block size up to 256 bits.

3.5 Secret Value Generation using Logistic Map

As previously mentioned, the chaotic maps are a good potential for information encryption due to their features such as their sensitivity to initial conditions and control parameters. There are several types of chaotic systems including Logistic, Tent, Lorenz, and Chen [17]. The proposed SPCBC mode takes the advantages of the chaotic maps to generate the needed secret values. It actually uses the logistic map formulated in Equation 1 which is reconsidered here again as.

$$x(\eta) = \mu x(\eta - 1)[1 - x(\eta - 1)]$$

Where x (0) ϵ (0, 1), μ, and η are the initial value, control parameter of the logistic map, the number of iterations, respectively. The generated sequence $\{x(\eta), \eta = 1, 2, 3, \ldots\}$ is chaotic which is neither periodic nor convergent. It is also sensitive to the initial value x (0) and the control parameter μ as well as the number of iterations η. Therefore, the proposed SPCBC mode considered these values (i.e., x (0), μ, η) as secret materials. Accordingly, the proposed SPCBC mode adapts the chaotic system in this Equation to generate a chaotic value $x_{i,j}(\eta)$ for each plaintext block $M_{i,j}$ based on Equation 10.

$$x_{i,j}(\eta_{i,j}) = \mu_{i,j}\, x_{i,j}(\eta_{i,j} - 1)[1 - x_{i,j}(\eta_{i,j} - 1)] \quad (10)$$

Where $x_{i,j}(\eta_{i,j})$ is the chaotic value associated with the i^{th} block in the j^{th} process (i.e., the chaotic value generated for the plaintext block $M_{(j-1)n+i}$ in the whole plaintext), $\eta_{i,j}$ is the number of iterations applied on $x_{i,j}$, and $\mu_{i,j}$ is the control parameter associated with $x_{i,j}$. The chaotic value $x_{i,j}(\eta_{i,j})$ is used in calculating the secret value $sv_{i,j}$ according to Equation 11.

$$SV_{i,j} = f_1(x_{i,j}(\eta_{i,j})) \tag{11}$$

Where f_1 ( ) is a function that maps the chaotic value $x_{i,j}(\eta_{i,j})$ into an integer value through calculating the Math floor function as defined in Equation 12.

$$f_1(x_{i,j}(\eta_{i,j})) = \lfloor x_{i,j}(\eta_{i,j}) \times 10^{14} \rfloor \tag{12}$$

In order to find the value of $f_1(x_{i,j}(\eta_{i,j}))$, $x_{i,j}(\eta_{i,j})$ should be calculated according to Equation 10. Also to evaluate $x_{i,j}(\eta_{i,j})$, the initial value $x_{i,j}(0)$ and the control parameter $\mu_{i,j}$ of chaotic map should be determined. Therefore, Equations 13 and 14 are constructed to find these values.

$$x_{i,j}(0) = \begin{cases} x_0 & if\ i = 0 \\ (x_0 + \dfrac{(t + j + CT)}{nonce}) \bmod 1 & if\ i = 1 \\ (x_0 + \dfrac{1}{f_2(C_{(i-1),j}) + j}) \bmod 1 & if\ 2 \le i \le t \end{cases} \tag{13}$$

$$\mu_{i,j} = \begin{cases} \mu_0 & if\ i = 0 \\ w + 0.43(\mu_0 + \dfrac{(j + CT)}{nonce}) \bmod 1 & if\ i = 1 \\ w + 0.43(\mu_0 + \dfrac{1}{f_2(C_{(i-1),j})}) \bmod 1 & if\ 2 \le i \le t \end{cases} \tag{14}$$

Where x_0 and μ_0 are preselected secret materials while CT is the selected counter. $C_{i\text{-}1,j}$ is the cipher-text block corresponding to the plaintext block $M_{i\text{-}1,j}$. The value of w is 3.57 because Equation 14 is designed in such a way that it satisfies the chaotic condition $3.57 \le \mu_{i,j} \le 4.0$ of the map. The *nonce* is a unique random value for each plaintext. Finally, the function f_2 ( ) is used to generate a dynamic value as a function of a cipher-text block $C_{i\text{-}1,j}$ and can be calculated according to Equation 15. In this way, employing both *nonce* and the function f_2 ( ) generates unique and dynamic values for $x_{i,j}(0)$ and $\mu_{i,j}$. Accordingly, the proposed SPCBC mode generates a different chain key for each plaintext block even if it is encrypted again. Thus, the SPCBC mode significantly enhances the security over the current block cipher modes of operation as discussed in Section 4.

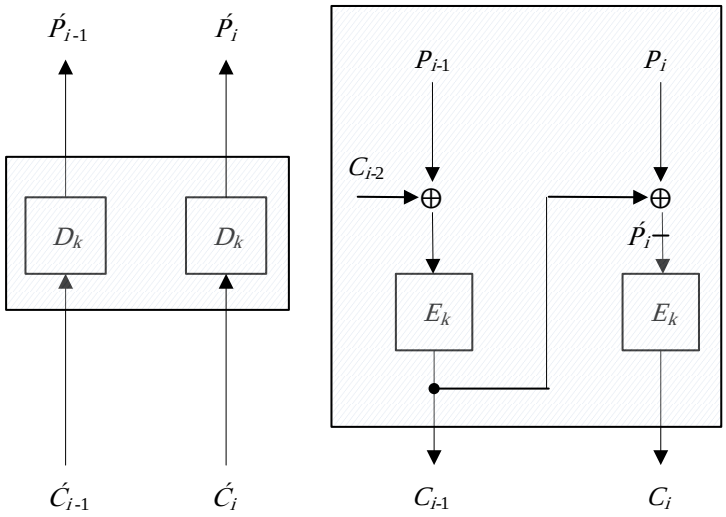
$$f_2(C) = 1 + (\sum_{i=0}^{m-1} (c_i \times 2^i) \bmod CT) \bmod CT \tag{15}$$

Where f_2 ( ) is a function that maps a cipher-text block into a value between 1 and CT inclusive, C is a cipher-text block, m is the size of the block C, and c_i is the i^{th} bit in the block C.

4. SPCBC Security Analysis

This section discusses the security aspects of the proposed SPCBC mode against the current modes of operation. Indeed, a secure cipher must resist all types of known attacks such as cipher-text only attacks, known-plaintext attacks and chosen plaintext/cipher-text attacks. Unfortunately, the current standard block cipher modes of operation such as ECB, CBC, and CTR are vulnerable to these types of attacks. In addition the nonstandard modes such as CC mode could be breached under these types of attacks. Such attacks can take place based on the related-mode attack model discussed in [13-15]. Accordingly, the rest of this section briefly applies the attack model on the current modes. In addition, it deeply employs the attack model into the proposed SPCBC mode to prove its robustness.

The related attack model illustrated that any of the current standard block cipher modes of operation can be exploited to attack other block cipher modes. The model assumes that an attacker has an access to either encryption or decryption oracle under the exploited mode. Phan and Siddiqi [14] exploited the ECB mode to attack the CBC, OFB, and CFB modes. For example, they attacked the CBC mode by exploiting the ECB mode as shown in **Fig. 6**. They assumed that a cryptanalyst or attacker has an access to ECB decryption oracle as depicted in **Fig. 6a**. So if the attacker has a cipher-text block $\acute{C}_i$ (also equals to C_i), he can obtain its corresponding plaintext block $\acute{P}_i$ in the CBC mode as shown in **Fig. 6b**, $\acute{P}_i$ is equal to $P_i \oplus C_{i-1}$ (i.e., $\acute{P}_i = P_i \oplus C_{i-1}$). Accordingly, the unknown plaintext block P_i can be obtained by XORing the two sides with C_{i-1}. Note that the hashed rectangular boxes in **Fig. 6**, **Fig. 7**, and **Fig. 8** delimit the parts inaccessible to a cryptanalyst or an attacker.



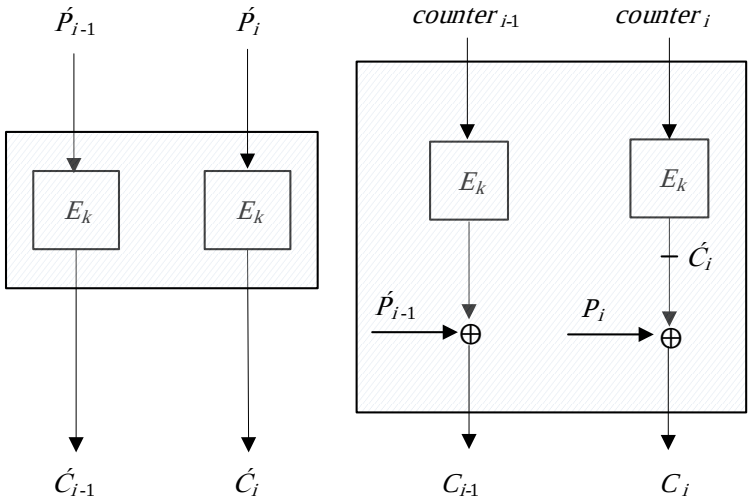
a) ECB decryption oracle. b) Attacking CBC mode.

Fig. 6. Exploiting ECB mode to attack CBC mode.

We can also follow the same way to attack other current modes such as the CC and CTR modes by exploiting the ECB mode. The CC mode divides a plaintext into independent sub-plaintexts, each sub-plaintext is independently encrypted in CBC mode (i.e., CC mode can end up as CBC mode). Therefore, CC mode can be broken with the same methodology used for attacking CBC mode if the attacker successfully recovers the first plaintext block from its

corresponding cipher-text block. This can be achieved if the attacker has unlimited access to ECB decryption oracle.

To attack the CTR mode by exploiting the ECB mode, assume that an attacker has an access to ECB encryption oracle as depicted in **Fig. 7a** and the initial counter is known [15]. In this case, the attacker can obtain the cipher-text block $\acute{C}_i$ of any plaintext block $\acute{P}_i$ (also equals to $counter_i$). In the CTR mode as visualized in **Fig. 7b**, the unknown plaintext P_i is equal to $\acute{C}_i \oplus C_i$ (i.e., $P_i = \acute{C}_i \oplus C_i$). Since the attacker has both the $\acute{C}_i$ and C_i, he can recover the unknown plaintext block P_i.



a) ECB encryption oracle. b) Attacking CTR mode.

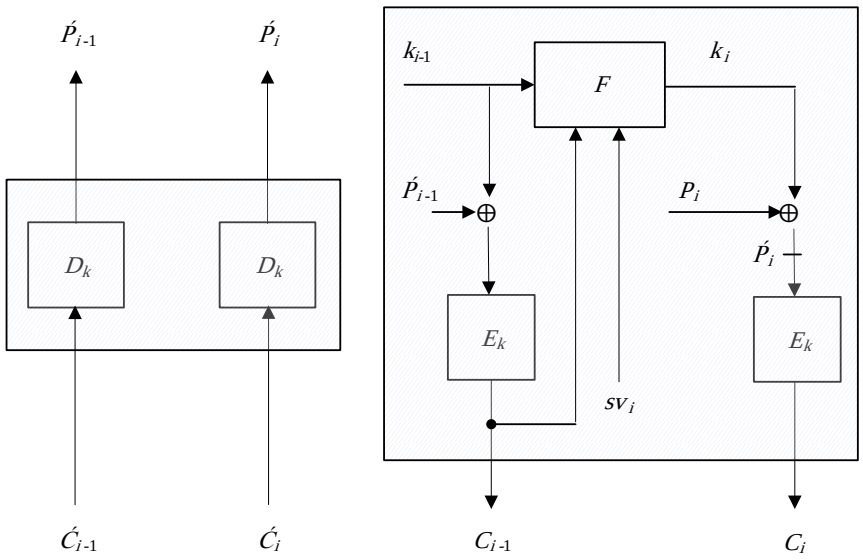
Fig. 7. Exploiting ECB mode to attack CTR mode.

In order to demonstrate the superior security of the proposed SPCBC mode, we are going to apply the underlying attack model to the SPCBC mode. For simplicity without loss of generality, we assumed that the proposed SPCBC mode is executed under a single processor. Therefore, the second subscript referred to the process *id* is removed from all variables in the discussion hereafter. For example, $C_{i,1}$, $k_{i,1}$, and $P_{i,1}$ become C_i, k_i, and P_i, respectively. To attack the SPCBC under the ECB, it is assumed that an attacker has an access to an ECB decryption oracle as depicted in **Fig. 8a**. In particular, if the attacker desires to recover the unknown plaintext block P_i corresponding to an intercepted cipher-text block C_i encrypted with the proposed mode, he should choose $\acute{C}_i = C_i$ to be fed to the ECB decryption oracle and hence obtains the corresponding $\acute{P}_i$. In this case, the attacker can obtain the plaintext block $\acute{P}_i$ for any intercepted cipher-text block $\acute{C}_i$. According to the structure of SPCBC, the unknown plaintext block P_i can be equal to $\acute{P}_i \oplus k_i$ (i.e., $P_i = \acute{P}_i \oplus k_i$) as depicted in **Fig. 8b**.

To recover the P_i, we need the value of $\acute{P}_i$ and k_i. The value of $\acute{P}_i$ is already obtained from the decryption oracle. Fortunately, k_i is unknown and hard to be obtained since the k_i, in SPCBC mode, is a dynamic and unique chain key generated for the i^{th} plaintext block. Note that, the creation of k_i is based on both a unique random value (*nonce*) generated for each message and a dynamic secret value sv_i. The sv_i inherits the features of randomness, noise-like behavior, and sensitivity to initial conditions and control parameters of logistic map. Hence, the dynamic behavior and the uniqueness of the secret value is guaranteed. Moreover, this results in producing a different chain key for each plaintext block even if the same plaintext

block is encrypted again. Therefore, the attacker has to search for the value k_i through brute force attack.

To illustrate this point, suppose you want to encrypt a message using a block cipher under the SPCBC mode. Also the message has l plaintext blocks M_1, M_2,…, M_l, each of size S. This means that a unique key stream k_1, k_2,…, k_l has to be generated for this specific message. In this case, if an attacker wants to recover the encrypted message, he has to search the key space associated with the message. According to the structure of the proposed SPCBC mode, the key space of chain keys is equal to (l x 2^s) under the underlying attack model. This indicates that the robustness of the proposed SPCBC mode increases as the size of the block increases. For example, suppose you want to encrypt a message using the AES under the SPCBC and the message has 1000 plaintext blocks. This means that the number of keys that should be searched by the attacker to recover the message is equal to (1000 x 2^{128}). Therefore, we can conclude that the proposed SPCBC mode is highly resistant to the current types of attacks such as cipher-text-only and chosen plaintext/cipher-text attacks. In addition, it has a superior security over the current block cipher modes of operation.



a) ECB decryption oracle. b) Attacking SPCBC mode.

Fig. 8. Exploiting ECB mode to attack the SPCBC mode.

5. Results

This section is devoted to demonstrate the encryption performance of different symmetric encryption algorithms deploying SPCBC mode. It first provides the settings of the experimental environment in Section 5.1. Then, the encryption performance of the algorithms deploying the proposed SPCBC mode against other current modes is discussed in Section 5.2.

5.1 Experimental Environment

To show the overhead cost in terms of encryption time due to the security strength implemented by the proposed SPCBC mode, we compare its performance in terms of encryption time with that of the CBC, CTR, and CC modes. The choice of using the encryption

time as the main criterion is to reflect encryption speed of each of these modes. To achieve the required performance verification, a multi-processor machine is needed. However, due to the unavailability of such machine, we alternatively use one of the methods of emulating multi-core machines [29]. In our experiment, we emulate an Octa-core (8 processors) machine with a network of eight machines as servers and a single client machine. A client-server application is developed to coordinate among these machines. The application on the client machine sends plaintext messages along with the control commands (such as selecting the encryption algorithm, used key, initial vector and mode) to one or more of the eight machines and the received encryption speed of each mode of operations under the test is registered. Each experiment is repeated 50 times and their normalized average results are reported hereafter.

5.2 SPCBC Mode Performance Evaluation

Based on the aforementioned experimental settings, we conducted different experiments and their results are shown hereafter. Accordingly, **Fig. 9** compares the normalized encryption time of the AES algorithm in CBC mode versus the proposed SPCBC mode. **Fig. 10** visualizes the performance of the three parallelizable modes of operation namely, the proposed SPCBC, CTR and CC modes. Finally, **Fig. 11** demonstrates the performance of the DES and AES algorithms in the CBC and SPCBC modes.

The results in **Fig. 9** plot the normalized encryption time in its normalized form against the message size for the CBC and SPCBC modes using AES algorithm. Note that, the CBC mode does not support multi-processors while the SPCBC supports parallel processors. Accordingly, the results of the CBC together with SPCBC mode using 4 and 8 processors are provided. The result indicates that the CBC mode has less encryption time than SPCBC mode when using up to 4 processors due to the overhead resulted from the security enhancement of the SPCBC mode. However, when using 8 processors or more, the SPCBC mode outperforms the CBC mode by providing lower encryption time. This means that the parallelization feature of the proposed SPCBC mode can compensate the security overhead.

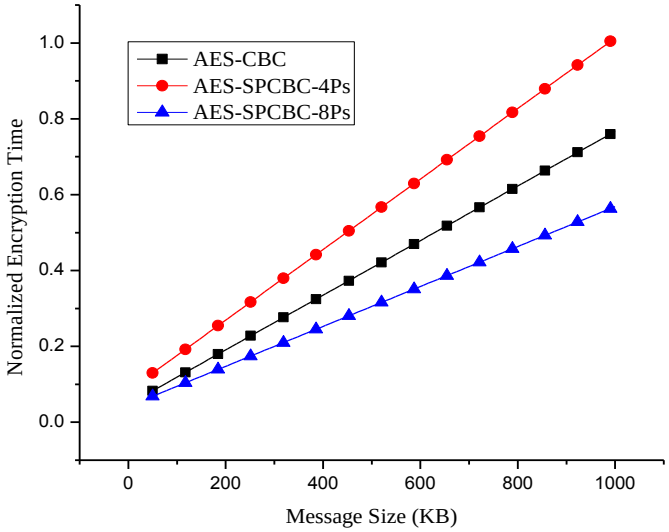


Fig. 9. AES encryption performance in CBC vs. SPCBC modes.

On the other hand, the results shown in **Fig. 10** depict the normalized encryption time of the proposed SPCBC against other modes that support parallelism; namely, the CTR and CC

modes. The results in the figure indicate that the CTR mode has the lowest encryption time and the CC mode has a little higher encryption time as a penalty for avoiding attacks associated with initial vector. Similarly, SPCBC has higher encryption time than both CTR and CC due to the stringent security it offers as discussed in Section 4. It is also notable that, the percentage increase in encryption time of the proposed SPCBC mode compared to the CC or CTR is fixed for all values of the message size due to the linear relationship between the message size and encryption time.

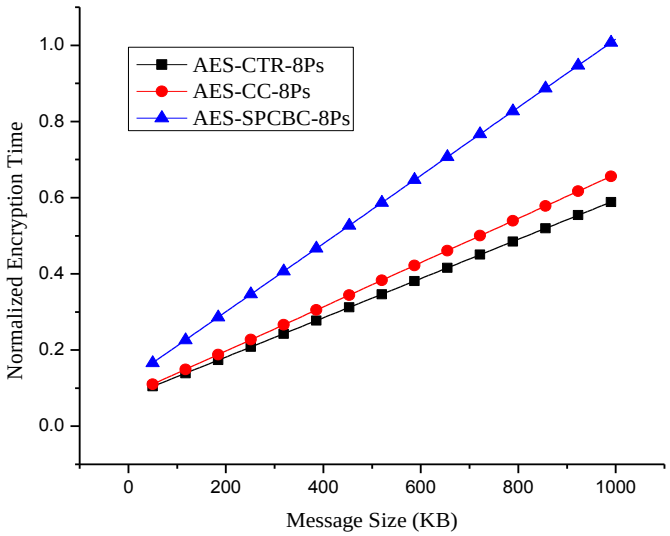


Fig. 10. AES encryption performance in CTR, CC, and SPCBC modes.

Furthermore, the results presented in **Fig. 11** demonstrate the performance of DES and AES algorithms in CBC and SPCBC modes. The results related to the CBC mode show that the DES algorithm has less encryption time than the AES algorithm. Similar results have been pointed out in [30]. On the other hand, the results related to the SPCBC mode indicate that the AES algorithm provides less encryption time than the DES algorithm. This is due the fact that the block size of AES algorithm is double that of the DES algorithm. Therefore, when a given message is encrypted under the SPCBC mode, it requires a number of chain keys for DES algorithm twice the number of chain keys required for the AES algorithm. This means that the time of obtaining chain keys for the DES algorithm is approximately double the time of creating chain keys for the AES algorithm. This leads to the finding that, in the SPCBC mode, the greater the block size, the better the encryption performance of an algorithm.

Finally, to better clarify the characteristics of the proposed SPCBC mode, **Table 1** presents a detailed comparison between SPCBC and other current related modes of operation. The comparison illustrates that the main features which distinct SPCBC mode from other modes generally and CBC and CTR modes particularly are its robustness against chosen plaintext/cipher-text attacks. This is due to that the SPCBC mode provides dynamic and unique key by employing the logistic map together with a nonce. More specifically, SPCBC surpasses CBC mode in providing a parallelizable implementation. When compared to CTR mode, SPCBC provides message integrity and chain dependency that may lead to an error propagation of one block. Moreover, SPCBC mode has the advantage over the CCM mode in its ability to use any message size and any block cipher. Finally, each of the CCM, CC, and SPCBC mode introduces a cipher-text that has the same size of the plaintext along with two

extra blocks, one block for providing authentication and the other block for conveying secret materials. In contrast, each of CBC and CTR produces cipher-text that has the same size as the plaintext.

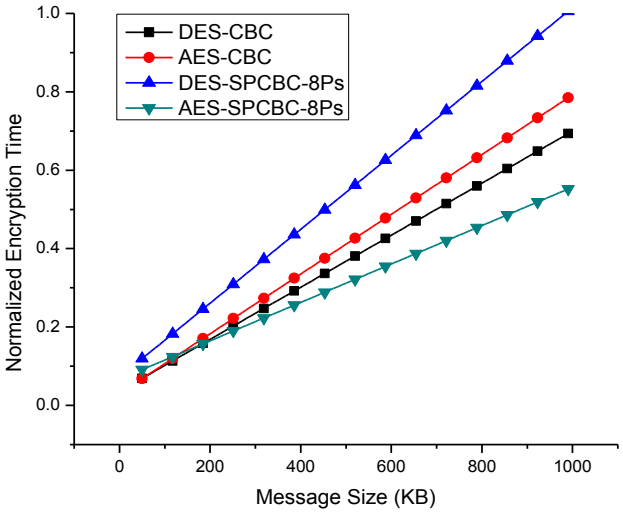


Fig. 11. Encryption performance of DES and AES in CBC and SPCBC modes.

Table 1. Comparison between SPCBC mode and different modes of operation.

Evaluation Criteria	CBC Mode	CTR Mode	CCM Mode	CC Mode	SPCBC Mode
Chain Dependency	Yes	No	No	Yes	Yes
Error Propagation	One Block	No	No	One Block	One Block
Providing Message Integrity	Yes	No	Yes	Yes	Yes
Providing Message Confidentiality	Yes	Yes	Yes	Yes	Yes
Number of Passes	One	One	Two	Two	Two
Providing Parallelism	No	Yes	No	Yes	Yes
Using Nonce	No	Possible	Yes	Possible	Yes
Employed Block Cipher	Any	Any	128-bit block size algorithms	Any	Any
Using Dynamic Key	No	No	No	No	Yes
Utilizing Unique Key for each Message	No	No	No	No	Yes
Chaos-based Mode	No	No	No	No	Yes
Message Size	Any	Any	Fixed	Any	Any
Vulnerable to attacks	Yes	Yes	No	Yes	No
Ciphertext size	The same size as the plaintext	The same size as the plaintext	The same size as the plaintext + 2 blocks	The same size as the plaintext + 2 blocks	The same size as the plaintext + 2 blocks

6. Conclusion

This paper proposed a new Secure Parallel Cipher Block Chaining (SPCBC) mode of operation that provides enhanced security over the current modes. The SPCBC mode achieved this stringent security through combining one-time chain keys with plaintext blocks before their encryption. The main challenge during the course of this paper was how to generate such chain keys. In particular, the SPCBC utilized the features of randomness, noise-like behavior, and sensitivity to initial conditions and control parameters of logistic map to obtain dynamic chain keys. In addition, it employed a nonce to guarantee the uniqueness of the generated chain keys. In this way, the produced chain keys have two main features: dynamic behavior and uniqueness. The SPCBC mode also provides a suitable performance via supporting encryption parallelism. Finally, the security analysis and experimental results indicated that the proposed SPCBC mode not only effectively resists attacks including known plaintext and chosen plaintext/cipher-text attacks but also has acceptable performance compared to the current modes.

Acknowledgement

This research was supported by Grant Number 3040/1434 from deanship of scientific research at Taibah University.

References

[1] Nigel Smart, "Cryptography: An Introduction," *McGraw-Hill*, 3rd Edition, 2002. ISBN: 0-077-09987-7. Article (CrossRef Link)

[2] Burt Kaliski, "PKCS# 5: Password-based cryptography specification version 2.0," *Internet RFC 2898*, Septemper, 2000. Article (CrossRef Link)

[3] William Stallings, *Cryptography and Network Security: Principles and Practices*, 4th Edition, Prentice Hall, Upper Saddle River, NJ, USA, 2005. ISBN: 0131873164. Article (CrossRef Link)

[4] FIPS PUB 81, DES Modes of Operation, National Bureau of Standards, U.S. National Institute of Standards and Technology (NIST), U.S. Department of Commerce, 1980. Article (CrossRef Link)

[5] Morris J. Dworkin, "Recommendation for Block Cipher Modes of Operation: Methods and Techniques," *Technical Report*, SP 800-38A, National Institute of Standards and Technology (NIST), 2001. Article (CrossRef Link)

[6] Chris Karlof, Naveen Sastry and David Wagner, "TinySec: a link layer security architecture for wireless sensor networks," in *Proc. of the 2nd ACM international conference on Embedded networked sensor systems*, pp. 162-175, November 3 – 5, 2004. Article (CrossRef Link)

[7] Morris J. Dworkin, "Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality," *Technical Report*, SP 800-38c, National Institute of Standards and Technology (NIST), 2004. Article (CrossRef Link)

[8] A. A. Adekunle and S. R. Woodhead, "A Resourceful Combined Block Cipher Mode of Operation for Packetised Network Communication," in *Proc. of the 4th International Conference on Next Generation Mobile Applications, Services and Technologies*, pp. 180-185, July 27-29, 2010. Article (CrossRef Link)

[9] Aly M. El-Semary, Mohamed M.A. Azim, "Counter Chain: A New Block Cipher Mode of Operation," *International Journal of Information Processing Systems*, vol. 11, no. 2, pp. 266-279, 2015. Article (CrossRef Link)

[10] Xu. Dewu, Chen Wei, "A survey on cryptanalysis of block ciphers," in *Proc. of the IEEE National Conference on Computer Application and System Modeling*, pp. 218-220, Oct. 22-24, 2010. Article (CrossRef Link)

[11] L.R. Knudsen, "Block Ciphers—a survey," *State of the Art in Applied Cryptography*, LNCS 1528, pp. 18-48, 1998. Article (CrossRef Link)

[12] Lars R. Knudsen and Matthew J. Robshaw, "the Block Cipher Companion," *Information Security and Cryptography*, vol. 2, pp. 1-12, Springer, 2011. Article (CrossRef Link)
[13] Hongjun Wu, "Related-cipher attacks," *Information and Communications Security*, vol. 2513 of the series Lecture Notes in Computer Science, Springer, pp. 447-455, 2002. Article (CrossRef Link)
[14] Raphael C. Phan, Mohammad U. Siddiqi, "Related-Mode Attacks on Block Cipher Modes of Operation," *Lecture Notes in Computer Science*, Springer, vol. 3482, pp. 661–671, 2005. Article (CrossRef Link)
[15] Dayin Wang, Dongdai Lin, and Wenling Wu, "Related-Mode Attacks on CTR Encryption Mode," *International Journal of Network Security*, vol. 4, no. 3, pp. 282-287, 2007. Article (CrossRef Link)
[16] Kathleen T. Alligood, Tim D. Sauer, James A. Yorke, *Chaos: an Introduction to Dynamical Systems*, Springer, New York, USA, 1996. ISBN: 978-0-387-94677-1. Article (CrossRef Link)
[17] Steven H. Strogatz, "Nonlinear Dynamics and Chaos: With Applications to Physics, Biology Chemistry, and Engineering," 2nd Edition, *Westview Press*, 2015. ISBN 978-0-813-34910-7 Article (CrossRef Link)
[18] Gonzalo Alvarez and Shujun Li, "Some Basic Cryptographic Requirements for Chaos-Based Cryptosystems," *International Journal of Bifurcation and Chaos*, vol. 16, no. 8, pp. 2129-2151, 2006. Article (CrossRef Link)
[19] Erdem Yavuz, Rifat Yazıcı, Mustafa Cem Kasapbaşı, and Ezgi Yamaç, "A chaos-based image encryption algorithm with simple logical functions," *Computers and Electrical Engineering*, vol 54, pp. 471-483, 2016. Article (CrossRef Link)
[20] José María Amigó, "Chaos-based cryptography," *Intelligent computing based on chaos*, vol. 184 of the series Studies in Computational Intelligence, pp. 291-313, 2009. Article (CrossRef Link)
[21] Nooshin Bigdeli, Yousef Farid, and Karim Afshar, "A robust hybrid method for image encryption based on Hopfield neural network," *Computers and Electrical Engineering*, vol. 38, no. 2, pp. 356-369, 2012. Article (CrossRef Link)
[22] Nanrun Zhou, Yixian Wang, Lihua Gong, Hong He, and Jianhua Wu, "Novel single-channel color image encryption algorithm based on chaos and fractional Fourier transform," *Optics Communications*, vol. 284, no. 12, pp. 2789-2796, 2011. Article (CrossRef Link).
[23] Yangzhong Zhou, Zhe Hua, Chi-Man Pun, CL Philip Chen, "Cascade Chaotic System with Applications," *IEEE Transactions on Cybernetics*, vol. 45, no. 9, pp. 2001-2012, 2015. Article (CrossRef Link)
[24] Bin Wang, Yingjie Xie, Changjun Zhou, Shihua Zhou and Xuedong Zheng, "Evaluating the permutation and diffusion operations used in image encryption based on chaotic maps," *Optik-International Journal for Light and Electron Optics*, vol. 127, no. 7, pp. 3541-3545, 2016. Article (CrossRef Link)
[25] A. Díaz-Méndez, J.V. Marquina-Pérez, M. Cruz-Irisson, R. Vázquez-Medina, and J. L. Del-Río-Correa, "Chaotic noise MOS generator based on logistic map," *Microelectronics Journal*, vol. 40, no. 3, pp. 638-640, 2009. Article (CrossRef Link)
[26] Xiaowei Li, Chengqing Li and In-Kwon Lee, "Chaotic image encryption using pseudo-random masks and pixel mapping," *Signal Processing*, vol. 125, pp. 48-63, 2016. Article (CrossRef Link)
[27] Phillip Rogaway, "Evaluation of some blockcipher modes of operation," *Technical Report*, Institute of Cryptography Research and Evaluation Committees for the Government of Japan, 2011. Article (CrossRef Link)
[28] R. Housley, "Using AES-CCM and AES-GCM Authenticated Encryption in the Cryptographic Message Syntax (CMS)," *RFC 5084*, November, 2007. Article (CrossRef Link)
[29] Tomasz Buchert, Lucas Nussbaum, and Jens Gustedt, "Methods for Emulation of Multi-Core CPU Performance," in *Proc. of IEEE 13th International Conference on High Performance Computing and Communications (HPCC-2011)*, pp. 288-295, September 2-4, 2011. Article (CrossRef Link)
[30] A. Nadeem and M. Y. Javed, "A performance comparison of data encryption algorithms," in *Proc. of IEEE 1st International Conference on Information and communication technologies*, pp. 84–89, August 27-28, 2005. Article (CrossRef Link)

Aly M. El-Semary received his B.S. degree in Systems and Computer Engineering, Faculty of Engineering, Al-Azhar University, Cairo, Egypt in 1992, and M.S. and Ph.D. degrees in Computer Science from Tulsa University, USA in 2001 and 2004, respectively. He works for the Department of Systems and Computer Engineering, Faculty of Engineering, Al-Azhar University, where he is currently an associate Professor. However, he is currently working as a visitor for Computer Science and Engineering College, Taibah University, Saudi Arabia. His current interests include network and computer security, sensor networks, fuzzy logic, data-mining, and neural networks. He is a member of IEEE and the author of several research papers published in the most reputable journals and conferences.

Mohamed Mostafa A. Azim received the B.Sc. degree in electrical engineering from Cairo University, Egypt in 1994, the M.Sc. degree jointly from the Fontys University and Technical University Eindhoven, Netherlands, in 1997, and the Ph.D. degree in computer sciences from Tohoku University, Japan, in 2006. He is the associate professor of electronics technology, Beni-suef University, Egypt. In 2008, he joined the college of computer science and engineering at Taibah University, KSA. On 2012, he became the chair of networks and communications systems department. He is the author of several research papers published in the most reputable journals and conferences. He is the author of the book titled “Optical Networks: A Restoration Perspective with Active.” He is also the editor of the book titled "Wireless Sensor Multimedia Networks: Architectures, Protocols and Applications" published by CRC Press USA. He is member of the editorial boards of the International Journal of Sensor and Related Networks and International Journal of Communication Networks and Information Security.

Hossam Diab graduated from computer science in 1999 and received the M.Sc. degree and Ph.D. degrees in Computer Science from faculty of science, Menoufia University, Egypt in 2004 and 2010, respectively. He is assistant professor at the department of mathematics and computer science, faculty of science, Menoufia University, Egypt. However, he is currently working as a visitor for Computer Science and Engineering College, Taibah University, Saudi Arabia. His research interests are in the areas of cryptography, application of chaotic systems in multimedia content encryption, digital image processing, image compression, image watermarking.