IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received March 21, 2018, accepted April 20, 2018, date of publication May 3, 2018, date of current version June 5, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2832854

A Network Coding and DES Based Dynamic Encryption Scheme for Moving Target Defense

HANQI TANG, QIFU TYLER SUN, (Member, IEEE),
XIAOLONG YANG, (Member, IEEE),
AND KEPING LONG, (Senior Member, IEEE)
School of Computer and Communication Engineering, University of Science and Technology Beijing, Beijing 10083, China

Corresponding author: Qifu Tyler Sun (qfsun@ustb.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61471034 and Grant 61771045, in part by the Ministry of Education of China under Grant 6141A02033307, in part by the Fundamental Research Funds for the Central Universities under Grant FRF-GF-17-B26, and in part by the Open Research Fund of Key Laboratory of Space Utilization, CAS, under Grant LSU-DZXX-2017-03.

ABSTRACT Unlike prior efforts in cybersecurity research, a dynamic defense theory, called moving target defense, increases the complexity and costs for attacks by effectively restricting the vulnerability exposure and the attack opportunities through various continually-changing evaluation, development mechanisms and strategy. Data encryption standard (DES) was the classical scheme of the traditional symmetric-key encryption schemes. Now it has been gradually replaced by the triple DES or advanced encryption standard (AES) so that the encoder has a larger key space. However, both the triple DES and AES cannot meet the dynamic security requirements of dynamic defense due to their static extension to the key space. In this paper, we propose a dynamic three-layer encryption scheme based on DES and network coding, with a low-complexity partial key update mechanism. Based on the theoretical analysis, the new scheme is shown to have the benefit to achieve a dynamic transition between efficiency and security, which increases its adaptability to various cyber conditions. The simulation results also show that the running ratio of the new scheme is relatively lower than or comparable to the triple DES.

INDEX TERMS Moving target defense, dynamic defense theory, cyber security, linear network coding, DES.

I. INTRODUCTION

Moving target defense (MTD) is one of the cyberspace game-changing revolutionary technologies proposed by Federal Networking and Information technology Research and Development (NITRD) in recent years [1]. Nowadays, network security configurations are typically deterministic, static and homogeneous. These features reduce the difficulties for cyber attackers scanning the network to identify specific targets and gather essential information. Thus, the attackers take the asymmetric advantages of building up, launching and spreading attacks, and the defenders are at a passive position. The existing defense mechanisms and approaches cannot reverse this situation. Therefore, MTD is proposed as a new revolutionary technology to alter the asymmetric situation of attacks and defenses [2], [3]. It keeps moving the attack surface of the protected target through dynamic shifting, which can be controlled and managed by the administrator. In this way, the attack surface exposed to attackers appears chaotic and changes all the time. Thus, the work effort, i.e., the cost and complexity for the attackers to launch a successful attack, will be greatly increased. As a result, the probability of successful attacks will be decreased, and the resiliency and security of the protected target will be enhanced effectively. The revolutions of MTD can be summarized from the following three aspects [3]: (i)Dynamic defense: the transformation from static to dynamic in system architecture. (ii)Active defense: the transformation from passive perception into actively setting blocks to the weakness and virus in security mechanism. (iii)Flexible defense: the transformation from regular into a flexible operation mode. The basic goal of MTD is to achieve the active defense to the external attacks based on unknown vulnerabilities and backdoors. To date, MTD has been studied in various contexts, including cloud computing [4], [5] and web applications [6], [7].

The similar dynamic idea can also be adopted in cryptography design. It is well known that Data Encryption Standard (DES) has been widely used as a mainstream symmetrical encryption. Meanwhile, DES has laid a foundation for the development and application of modern block cipher theory [8]. At present, with the rapid development of computing power, the classic iterated block cipher DES has become very fragile, which causes the effective realization of DES crack by the exhaustive attack. So, it has gradually been replaced by the triple-DES algorithm or Advanced Encryption Standard (AES) so that the encoder has a large enough key space. However, due to the existence of S-box, DES still has benign incalculability to analysis attack [9]. Two of the most effective methods of iterated block-cipher attack are differential cryptanalysis (DC) and linear cryptanalysis (LC). DC is the first published method that can crack DES successfully in the average computational complexity of less than 2^{55}. It indicates that if there are 2^{47} chosen plaintexts, the upper bound of the computational complexity of crack is 2^{47}. Although 2^{47} are much smaller than 2^{55}, the condition of 2^{47} chosen plaintexts is only theoretically meaningful [9]. The latest LC shows that getting 2^{43} arbitrary plaintexts leads to cipher crack. Obviously, ‘‘this is better but a small forward step’’ [9], so the analysis attack is still difficult to crack DES in the real world. Besides, reference [10] concluded that DES can resist the Timing Attack successfully as well. In addition, in order to optimize DES, many improved algorithms have been proposed such as: multiple DES, mutable S-box DES, sub key DES, G-DES, DES-X, s^n DES and so on (See, e.g., [11]–[13]).

Although the above-mentioned algorithms such as the triple DES and AES have gradually replaced the classical DES, they still cannot meet the dynamic security requirements of the intelligent information network due to their static extension to the key space. In this paper, we present an encryption scheme to improve DES under the concept of MTD, by means of (linear) network coding (NC), which advocates linearly combining coding along with data propagation [14]. The following two reasons motivate us to choose NC. First, NC, which has been used in [15] and [16] for encryption scheme design, changes the static nature of network information transmission, so it is a good match to achieve the dynamic, active and random features of MTD as defined in [3]. Second, the use of NC as an encryption scheme has the potential to resist the exhaustive attack, as an L-bit plaintext may correspond to $\Omega(2^{L\times L})$ possible ciphertexts.

We make the following main contributions in this paper:

- We propose a novel encryption scheme consisting of 3 layers. The inner and outer layers essentially perform NC and the middle layer implements DES. In consequence, the new scheme has good behavior to resist both exhaustive and analysis attacks. We also validate that the running ratio of the proposed scheme is relatively lower than or comparable to the triple DES.
- The proposed scheme can achieve the MTD features by the following procedures. First, a re-encryption process can be implemented on the outer NC layer of the scheme, so that the key and ciphertexts can be *dynamically* changed. Second, the key length can be *actively* extended, so that the scheme is adaptable to the rapid development of the computing power. Third, the parameters in the scheme can be *flexibly* chosen, so that there is a transition between efficiency and security.

The reminder of the paper is organized as follows. In Section II, we review some useful fundamentals in DES and NC. In Section III, we present in detail the novel triple encryption scheme combining NC with DES. In Section IV, we theoretically justify and numerically validate the feasibility of the proposed scheme. Section V discusses the benefits of the proposed scheme and Section VI concludes the paper.

II. PRELIMINARIES

A. DES FEATURES AGAINST ANALYSIS AND EXHAUSTIVE ATTACK

In the study of Cryptanalysis, the efficiency of the analysis attack to crack a cipher depends on the times of encryption iteration. Reference [20] showed that the principle of choosing proper times of iteration follows that making the efficiency of analysis attack is lower than that of exhaustive attack: if the times of iteration in DES is less than 16, analysis attack (like DC or LC) will have the higher efficiency compared to the exhaustive attack. The reason why this principle is attractive is that it makes judging the strength and the advantages of an algorithm quite simple: if there is no breakthrough in Cryptanalysis, the strength of any encryption algorithm satisfying the principle only depends on the key space [8].

At present, DES cannot keep the computational security with the rapid development of computing power. The exhaustive attack causes the effective crack to DES with a much lower cost. However, DES still has benign incalculability to analysis attack and timing attack. The Achilles’ heel of S-box design in DES has not been found so far. Moreover, the design of iteration times in DES makes the lower efficiency of analysis attack than that of exhaustive attack. It turns out that the major weakness of DES is the *short key space*.

In existing DES variations, double encryption is very fragile in front of the man-in-the-middle attack [18] and it cannot achieve the goal of using multiple encryptions to increase the key length. The triple encryptions increase the key length to 112, which is computationally secure and widely used for now. However, the encryption/decryption complexity of the triple DES is 3 times the single DES, and every time the dynamic re-encryption process of the triple DES requires decrypting the original message first and then encrypting it again based on a new key. Thus, the triple DES is not suitable for efficient dynamic cyber security protection as required in intelligent networks. Meanwhile, the former analysis is also meaningful to AES. So it is important to find a new way to achieve dynamic cyber security with efficient operations and low complexity.

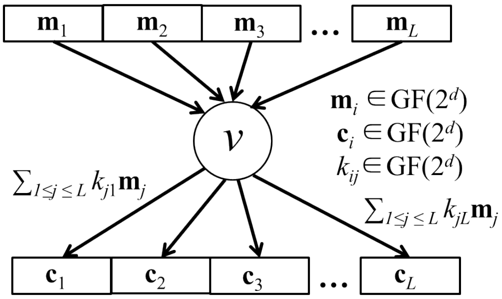

FIGURE 1. NC Operations.

B. NETWORK CODING (NC) AND MATRIX REPRESENTATION OF FINITE FIELDS

In the theory of NC [19], [20], data symbols transmitted along the edges in a network belong to a finite field GF(q), where q can be either a prime or a prime power. Every outgoing edge of a node v transmits a data symbol that is a GF(q)-linear combination of the incoming data symbols to v. Such a coding mechanism is referred to as NC. Specifically, assume the information on the incoming edges to be a binary sequence and then divide the sequence into L blocks (vectors) $\mathbf{m}_1$, $\mathbf{m}_2 \ldots \mathbf{m}_L$ of a fixed length d. The fixed length equals the dimension of the extension field GF(2^d) and every $\mathbf{m}_i$ can be regarded as either a binary vector over GF(2) or an element over GF(2^d). Then, every outgoing edge also transmits a GF(2^d)-linear combination of information on all incoming edges. This NC mechanism is shown in Fig.1.

The above mentioned mechanism of NC has been discussed for the use in large scale distributed storage systems [21], [22]. In the case that the dimension of the extension field GF(2^d) is very large and dynamically changed, instead of storing several lookup tables, it would be necessary to find a convenient way to realize both multiplication and addition arithmetic over GF(2^d).

The theory of finite fields shows that every generator α of the extension field GF(2^d) satisfies $p(\alpha) = 0$ for some primitive polynomial $p(x)$ of this extension field. In other words, the generator α is the root of the primitive polynomial and α is called the primitive root. Then GF(2^d) can be represented by α as an additive or a multiplicative structure in which the corresponding addition or multiplication operation is an efficient operation. But most elements in two structures do not have a specific mapping relation, which leads to the low efficiency of the other operation in a field.

Based on the above, in order to realize the efficient transformation, reference [23] has shown a *standard matrix representation* that naturally and simply displays both the multiplicative and the additive structures of the field GF(2^d). Actually, the standard matrix representation is the core of the dynamic encryption algorithm achieved by NC in this paper to simplify the operations over the extension field.

Let **K**(the symbols in bold-face are used to distinguish the vector or matrix from the ordinary variates) be the D × D companion matrix of an arbitrary primitive polynomial $p(x)$ of degree d over GF(2). If $p(x) = a_0 + a_1x + a_2x^2 + a_3x^3 \ldots + a_{d-1}x^{d-1} + x^d$, the structure of **C** will be designed as:

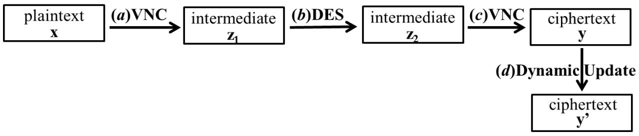

FIGURE 2. The basic procedures of the proposed encryption scheme.

$$\boldsymbol{K} = \begin{bmatrix} 0 & -a_0 \\ & -a_1 \\ & -a_2 \\ \mathbf{I} & \vdots \\ & -a_{d-1} \end{bmatrix}$$

Based on GF(2), a_i equals to 0 or 1, and hence $-a_i = a_{i.}$ mod 2. For instance,$p(x) = x^3 + x + 1$ is a primitive polynomial over GF(2). Then,

$$\boldsymbol{K} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

According to the Cayley-Hamilton theorem [23], $p(\mathbf{K}) = 0$ (Recall that Cayley-Hamilton theorem states that the characteristic polynomial of the matrix **K** is the annihilation polynomial of **K**). Thus, elements in GF(2^d) can be represented by $\{\mathbf{0}, \mathbf{K}, \mathbf{K}^2 \ldots \mathbf{K}^{2^d-1}(=\mathbf{I})\}$, and both addition and multiplication operations in GF(2^d) can be represented by matrix addition and multiplication. This matrix representation of extension fields makes NC a good candidate to achieve the dynamic security mechanism in a networking system due to its flexibility of choosing coding matrices.

III. DESIGN OF THE DYNAMIC ENCRYPTION SCHEME

In this section, we present a novel encryption scheme with an efficient dynamic re-encryption process, which has a good behavior to resist both exhaustive and analysis attacks, and the potential to be compatible with existing secure NC schemes.

The proposed encryption scheme, as shown in Fig.2, consists of 4 steps. The first 3 steps comprise the procedure to generate a ciphertext while the last step implements dynamic re-encryption, as outlined below:

(a) Inner Layer Encryption Embedding NC. In this step, the plaintext **x**, which is a binary row vector, is converted to a binary intermediate sequence $\mathbf{z_1}$ based on a high-dimensional binary invertible matrix $\mathbf{K}_a$ generated by the concept of NC. The main purpose of this step is to extend the key space of the algorithm, so as to resist the exhaustive attack.

(b) Middle Layer DES Encryption. The middle layer encryption step adopts DES to encode intermediate sequence $\mathbf{z_1}$, and get another intermediate sequence $\mathbf{z_2}$. The main purpose of this step is to exploit the design of S-box in DES to bring non-linearity into the encryption scheme, and hence to effectively defense the analysis attack.

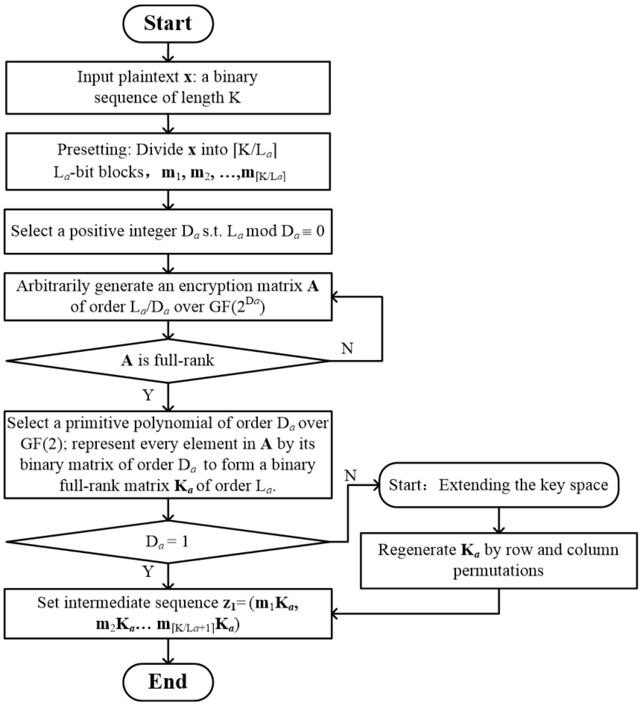
Start
Input plaintext x: a binary sequence of length K
Presetting: Divide x into ⌈K/L_a⌉ L_a-bit blocks, m_1, m_2, …, m_⌈K/La⌉
Select a positive integer D_a s.t. L_a mod D_a ≡ 0
Arbitrarily generate an encryption matrix A of order L_a/D_a over GF(2^Da)
A is full-rank
N
Y
Select a primitive polynomial of order D_a over GF(2); represent every element in A by its binary matrix of order D_a to form a binary full-rank matrix K_a of order L_a.
D_a = 1
N
Y
Start: Extending the key space
Regenerate K_a by row and column permutations
Set intermediate sequence z_1=(m_1K_a, m_2K_a… m_⌈K/La+1⌉K_a)
End

FIGURE 3. The procedure of Step (*a*).

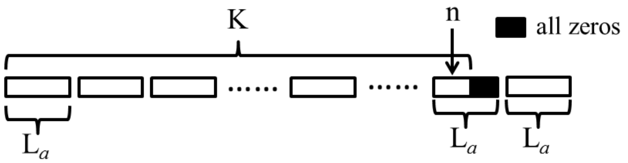

FIGURE 4. The presetting of the plaintext x.

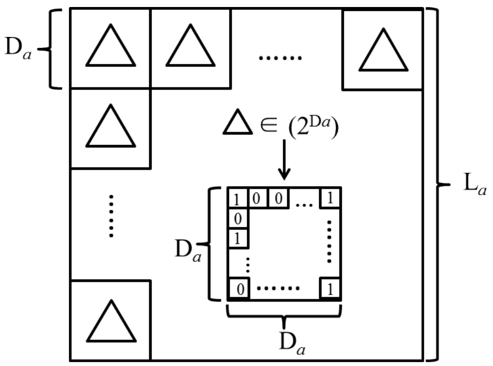

FIGURE 5. The formation of binary matrix $\mathbf{K}_a$ from matrix A over GF(2^{D_a}).

(c) Outer Layer Encryption Embedding NC. In this outer layer encryption step, NC is adopted again to generate a low-dimensional binary invertible matrix $\mathbf{K}_c$ to encode intermediate sequence $\mathbf{z}_2$, and the ciphertext **y** is subsequently obtained. The purpose of this step is to take advantage of NC to provide an interface for dynamic and efficient update, and to construct the triple encryption model to resist the man-in-the-middle attack, which is a common and efficient crack in double encryption schemes as mentioned before.

(d) Dynamic Update of the Ciphertext. The dynamic update procedure to the ciphertext can be regarded as a rerun of step (*c*) based on a new binary encoding matrix. It is particularly designed to realize dynamic security protection. The flexibility to choose the new binary encoding matrix endows a tradeoff between efficiency and security, which enhances the adaptability to different application scenarios.

It is worthwhile to note that using invertible matrices in step (*a*) and (*c*) for encryption is essentially a type of K-block cipher. The novel idea in this paper is that we can find an efficient way to get the feasible and dynamically updatable encryption matrix based on NC.

We next present the scheme in detail step by step.

A. INNER LAYER ENCRYPTION

The procedure of step (*a*) is illustrated in Fig.3.

Initially, plaintext **x** is the input of the encryption scheme. It is assumed to be a binary sequence of length K. Step (*a*) first divides the sequence into $\lceil K/L_a \rceil$ L_a-bit blocks: $\mathbf{m}_1\mathbf{m}_2\ldots\mathbf{m}_{\lceil K/L_a \rceil}$. If K is not divisible by L_a, then the last block $\mathbf{m}_{\lceil K/L_a \rceil}$ consists of n-bit plaintext data and L_a–n padded zero bits. Meanwhile, an additional L_a-bit block is added at the end of the entire sequence to indicate the value of n. The presetting of the plaintext is depicted in Fig.4.

Next, the routine will generate the encryption matrix $\mathbf{K}_a$. Select a positive integer D_a which divides L_a. Repeatedly and randomly generate a square matrix **A** of order L_a/D_a over the field GF(2^{D_a}), till **A** has full rank L_a/D_a. Select an arbitrary primitive polynomial $p(x)$ of degree D_a over GF(2). According to the standard matrix representation of an extension field as reviewed in Section II, represent every element in **A** by its binary matrix of order D_a, so that a binary matrix $\mathbf{K}_a$ of order L_a is formed, which will be shown invertible in the next section. The relation between matrix **A** over GF(2^{D_a}) and the binary invertible matrix $\mathbf{K}_a$ is illustrated depicted in Fig.5.

If the selected D_a is larger than 1, it is optional to perform an additional step which is referred to as *extending the key space* to enrich the key space, that is, the possible choices of $\mathbf{K}_a$:

- Randomly choose *permutation matrices* $\mathbf{P}_{1a}$, $\mathbf{P}_{2a}$ of order L_a and reset $\mathbf{K}_a$ to be $\mathbf{P}_{1a}\mathbf{K}_a\mathbf{P}_{2a}$.

The above method regenerates $\mathbf{K}_a$ by row and column permutations. In order to reduce the regeneration complexity, we can adopt the following two methods, which restrict the permutations within a smaller set.

- Randomly choose two permutation matrices $\mathbf{P}_{1a}$, $\mathbf{P}_{2a}$ of order L_a/D_a. Expand $\mathbf{P}_{1a}$, $\mathbf{P}_{2a}$ to permutation matrices $\mathbf{P}'_{1a}$, $\mathbf{P}'_{2a}$ of order L_a via replacing every entry with value 0 by a $D_a \times D_a$ zero matrix and every entry with value 1 by a $D_a \times D_a$ identity matrix. Reset $\mathbf{K}_a$ to be $\mathbf{P}'_{1a}\mathbf{K}_a\mathbf{P}'_{2a}$. This is equivalent to reset **A** to be $\mathbf{P}_{1a}\mathbf{A}\mathbf{P}_{2a}$ and then regenerate $\mathbf{K}_a$ from **A**.

- When $\mathbf{K}_a$ is generated by $\mathbf{A}$, different choices of primitive polynomial $p(x)$ of degree D_a over GF(2) yield different $\mathbf{K}_a$, each of which can be obtained by row and column operations by another. So, $p(x)$ can be randomly selected for generating $\mathbf{K}_a$.

Finally, based on the generated $\mathbf{K}_a$, the routine proceeds to generate intermediate binary sequence $\mathbf{z_1}$ as $(\mathbf{m}_1\mathbf{K}_a, \mathbf{m}_2\mathbf{K}_a \ldots \mathbf{m}_{\lceil K/L_a+1 \rceil}\mathbf{K}_a)$, which consists of $(\lceil K/L_a \rceil + 1)$ L_a-bit blocks. The step (a) is accomplished.

B. MIDDLE LAYER & OUTER LAYER ENCRYPTION

Initially, intermediate sequence $\mathbf{z_1}$ from the former step and a predetermined DES 64-bit key sequence are the inputs of this encryption scheme. The middle layer encryption adopts DES to generate intermediate binary sequence $\mathbf{z_2}$ from $\mathbf{z_1}$. Specifically, $\mathbf{z_1}$ is divided into L_b-bit blocks, where L_b is conventionally set to 64, same as the length of DES key. Every L_b-bit block is encrypted by the laws of DES and then the L_b-bit DES-encrypted blocks sequentially form $\mathbf{z_2}$.

Next, $\mathbf{z_2}$ will be encoded to ciphertext $\mathbf{y}$ by a similar routine to step (a). Divide the binary sequence $\mathbf{z_2}$ into L_c-bit blocks, where L_c is a parameter smaller than L_a of step (a) for the purpose of more efficient rerun of step (c) to achieve dynamic security protection. Select a positive integer D_c which divides L_c. Repeatedly generate a square matrix $\mathbf{C}$ of order L_c/D_c over the field GF(2^{D_c}), till $\mathbf{C}$ has full rank L_c/D_c. Select an arbitrary primitive polynomial $p(x)$ of degree D_c over GF(2). According to the standard matrix representation of an extension field, represent every element in $\mathbf{C}$ by its binary matrix of order D_c, so that a binary invertible matrix $\mathbf{K}_c$ of order L_c is formed. In order to further enrich the possible choices of $\mathbf{K}_c$, by the same method described previously, $\mathbf{K}_c$ can also be modified via random row/column permutation. Sequentially multiply the L_c-bit blocks in $\mathbf{z_2}$ by $\mathbf{K}_c$ and then juxtapose the resultant L_c-bit blocks to form the ciphertext $\mathbf{y}$.

C. DYNAMIC RE-ENCRYPTION

The procedure of step (d) is outlined in Fig.6. The dynamic update procedure to the ciphertext can be regarded as a rerun of step (c) based on a new binary encryption matrix $\mathbf{K}_c$, so that a new ciphertext $\mathbf{y}'$ can be generated. It is particularly designed to realize dynamic security protection and will be referred to as *partial key update*.

In general, the scheme can recover the binary intermediate sequence $\mathbf{z_2}$ via the multiplication between the original ciphertext $\mathbf{y}$ and inverse matrix of $\mathbf{K}_c$. Then reselect the L_c, D_c and primitive polynomial $p(x)$. Repeat the related steps and finally get the new ciphertext $\mathbf{y}'$again.

When high encryption efficiency is required, the following two optional steps with relatively low computational complexity and security can be used to update the encryption matrix $\mathbf{K}_c$ in step (d):

- There is no need to get the intermediate sequence $\mathbf{z_2}$ again. Keep L_c, D_c and the primitive polynomial $p(x)$ unchanged. Repeatedly generate a square matrix $\mathbf{D}$ of order L_c/D_c over the field GF(2^{D_a}), till $\mathbf{D}$ has full rank L_c/D_c. Matrix $\mathbf{D}$ is generated essentially in a same manner as matrix $\mathbf{C}$. Use the same primitive polynomial $p(x)$ of degree D_c over GF(2), and then represent every element in $\mathbf{D}$ by its binary matrix of order D_c to get a new full-rank matrix $\mathbf{K}_d$ of order L_c.

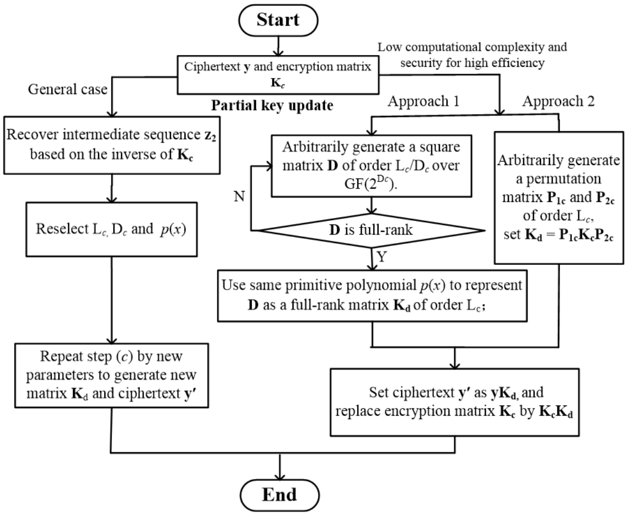
Start
Ciphertext y and encryption matrix K_c
General case
Partial key update
Low computational complexity and security for high efficiency
Approach 1
Approach 2
Recover intermediate sequence z_2 based on the inverse of K_c
Reselect L_c, D_c and p(x)
Repeat step (c) by new parameters to generate new matrix K_d and ciphertext y′
Arbitrarily generate a square matrix D of order L_c/D_c over GF(2^{D_c}).
N
D is full-rank
Y
Use same primitive polynomial p(x) to represent D as a full-rank matrix K_d of order L_c;
Arbitrarily generate a permutation matrix P_1c and P_2c of order L_c, set K_d = P_1cK_cP_2c
Set ciphertext y′ as yK_d, and replace encryption matrix K_c by K_cK_d
End

FIGURE 6. **The procedure of Step (*d*).**

- Directly choose two random permutation matrices $\mathbf{P_{1c}}$ $\mathbf{P_{2c}}$ of order L_c to achieve the permutation of $\mathbf{K}_c$ as $\mathbf{P_{1c}K_cP_{2c}}$. The result of the multiplication is the new full-rank matrix $\mathbf{K}_d$.

We can find that the second option in partial key update is a similar routine to the first option in extending the key space in step (a). Both options use permutation matrices to randomize order of the rows and columns in the matrices $\mathbf{K}_a$ and $\mathbf{K}_c$. There are still two more efficient alternative options in step (a), but they are not correspondingly adopted in step (d). The reason is that the values of L_c and D_c used in step (d) are much smaller than those in step (a), and thus the effect of the aforementioned two alternative options to reduce the regeneration complexity is relatively low. In other words, the aforementioned two alternative options in step (a) for efficiency enhancement are negligible in step (d). The specific selection of the parameters will be discussed in Section IV.

The result of the multiplication between $\mathbf{y}$ and $\mathbf{K}_d$ is the new ciphertext $\mathbf{y}'$. Then update the encryption matrix in step (c) stored in the system. Via the multiplication between $\mathbf{K}_c$ and $\mathbf{K}_d$, a new encryption matrix $\mathbf{K}_c$ is generated, which is regarded as a new encryption matrix for step (c) and will be stored in the system.

D. DECRYPTION OF THE SCHEME

The keys to be protected in the scheme are matrices $\mathbf{K}_a$, $\mathbf{K}_c$ used in NC, and the 64-bit key sequence used in DES. The designed encryption scheme is symmetrical and the safety of the keys must be ensured. It is assumed that the receivers can obtain a copy of the secret keys in some safe channels.

The receivers rebuild the plaintexts from the ciphertext in the following simple way, which is essentially a reverse run from Step (c) to (a) based on inverse keys. First, the ciphertext $\mathbf{y}$ is multiplied by the inverse matrix of $\mathbf{K_c}$ to obtain the intermediate sequence $\mathbf{z_2}$. Second, the intermediate sequence $\mathbf{z_2}$ is decrypted to $\mathbf{z_1}$ by the 64-bit DES key in a reversed order. Finally, $\mathbf{z_1}$ will be decoded to the plaintext $\mathbf{x}$by multiplying the inverse matrix of $\mathbf{K_a}$ and thus the original message is successfully rebuilt.

It is worthwhile to note that the whole cipher system proposed herein abides by the *Feistel* structure [18], which endows very similar, or even exactly the same encryption and decryption processes, and thus enhances the efficiency in implementation.

IV. JUSTIFICATION AND VALIDATION OF THE SCHEME

A. THEORETICAL JUSTIFICATION

In this section, we shall theoretically justify the encryption scheme by proving that the generated binary matrix $\mathbf{K_a}$ is indeed full rank. In particular, we discuss the effect of parameters L_a and D_a on the probability of a randomly generated matrix to be full-rank in step (a).

Recall that in step (a), the encryption matrix $\mathbf{K_a}$ is obtained from a full-rank $(\mathrm{L}_a/\mathrm{D}_a)\times(\mathrm{L}_a/\mathrm{D}_a)$ matrix $\mathbf{A}$ over GF(2^{D_a}). Let ϕ denote the mapping that maps every element in GF(2^{D_a}) to its binary $\mathrm{D}_a\times\mathrm{D}_a$ matrix representation under some primitive polynomial over GF(2) as discussed in Section II. Thus, ϕ is an homomorphism from GF(2^{D_a}) to the ring of binary $\mathrm{D}_a\times\mathrm{D}_a$ matrices. Applying componentwise, the homomorphism ϕ extends to a mapping from the ring of $(\mathrm{L}_a/\mathrm{D}_a)\times(\mathrm{L}_a/\mathrm{D}_a)$ matrices over GF(2^{D_a}) to the ring of binary $\mathrm{L}_a\times\mathrm{L}_a$ matrices. In this way, $\mathbf{K_a} = \phi(\mathbf{A})$. As $\mathbf{A}$ is a full rank matrix over GF(2^{D_a}), there exists another matrix $\mathbf{A}'$ over GF(2^{D_a}) such that $\mathbf{A}'\,\mathbf{A} = \mathbf{I}$. Consequently,

$$\phi(\mathbf{A}')\mathbf{K_a}{=}\phi(\mathbf{A}')\phi(\mathbf{A}) = \phi(\mathbf{A}'\mathbf{A}) = \phi(\mathbf{I}) = \mathbf{I},$$

where the second equality holds because ϕ is a homomorphism. This implies the full-rank of the encryption matrix $\mathbf{K_a}$.

In step (a), in the course of generating the encryption matrix $\mathbf{K_a}$, not only the parameter L_a, but also the parameter D_a endows a transition between the efficiency and security in the scheme. To see this, we first consider two extreme values of $\mathrm{D}_a$in step (a).

For the case $\mathrm{D}_a = 1$, that is, the minimum choice of D_a, the number of $\mathrm{L}_a\times\mathrm{L}_a$ optional matrices over GF(2) is $2^{\mathrm{L}_a\times\mathrm{L}_a}$ and the number of full-rank matrices over GF(2) is $(2^{\mathrm{L}_a}-1)\times(2^{\mathrm{L}_a}-2)\times\cdots\times(2^{\mathrm{L}_a}-2^{\mathrm{L}_a-1})$. Thus, the probability, to be denoted by P_{L_a}, of a randomly generated $\mathrm{L}_a\times\mathrm{L}_a$ matrix over GF(2) to be full-rank is

$$P_{\mathrm{L}_a} = \prod\nolimits_{i=0}^{\mathrm{L}_a-1}\frac{2^{\mathrm{L}_a}-2^i}{2^{\mathrm{L}_a}} \qquad (1)$$

Under a fixed L_a, the number to randomly generate an $\mathrm{L}_a\times\mathrm{L}_a$ matrix over GF(2) till a full rank matrix is obtained is geometrically distributed with parameter P_{L_a}, and hence an average of $1/P_{\mathrm{L}_a}$ generations is required to yield an $\mathrm{L}_a\times\mathrm{L}_a$ full-rank matrix over GF(2). According to (1), the value of L_a also affects the probability P_{L_a}.

TABLE 1. When $\mathbf{D_a = 1}$, the probability P_{L_a} of a randomly generated $\mathrm{L}_a\times\mathrm{L}_a$ matrix over GF(2) to be full-rank.

L_a	P_{L_a}
2	0.375
4	0.3076172
8	0.28991912
16	0.2887925
32	0.2887881
64	0.2887881
128	0.2887881

Proposition 1. The probability of a randomly generated $(\mathrm{L}_a+1)\times$ (L_a+1) matrix, denoted by P_{L_a+1}, over GF(2) to be full rank equals to $(1-1/2^{\mathrm{L}_a+1})P_{\mathrm{L}_a}$.

Proof: According to (1),

$$\begin{aligned}&P_{\mathrm{L}_a+1}\\&\quad=\prod\nolimits_{i=0}^{\mathrm{L}_a}\frac{2^{\mathrm{L}_a+1}-2^i}{2^{\mathrm{L}_a+1}}=\frac{2^{\mathrm{L}_a+1}-1}{2^{\mathrm{L}_a+1}}\cdot\prod\nolimits_{i=1}^{\mathrm{L}_a}\frac{2^{\mathrm{L}_a+1}-2^i}{2^{\mathrm{L}_a+1}}\\&\quad=(1-1/2^{\mathrm{L}_a+1})\cdot\prod\nolimits_{i=0}^{\mathrm{L}_a-1}\frac{2^{\mathrm{L}_a}-2^i}{2^{\mathrm{L}_a}}=(1-1/2^{\mathrm{L}_a+1})P_{\mathrm{L}_a}\end{aligned}$$

■

At the first glance, it seems that the larger the parameter L_a is, the smaller the probability P_{L_a} is as the factor $1–1/2^{\mathrm{L}_a+1}$ is smaller than 1. However, as listed in Table 1, the probability P_{L_a+1} does not decrease much under moderate L_a. Actually, there is a theoretical positive lower bound on P_{L_a}, as proved in the theorem below. This implies that the average number of random generations required to yield a full rank encryption matrix $\mathbf{K_a}$ over GF(2) is smaller than $1/0.2887 < 3.464$ and hence justifies that the basic design for $\mathbf{K_a}$ is feasible for the condition $\mathrm{D}_a = 1$.

Theorem 2. For any nonnegative integer L_a, P_{L_a} is lower bounded by 0.2887.

Proof: As a consequence of Proposition 1, $P_{\mathrm{L}_a+1} \leq P_{\mathrm{L}_a}$. When L_a tends to infinity,

$$\begin{aligned}P_{\mathrm{L}_a} &= \prod_{i=1}^{\infty}(1-1/2^i)\\&> (1-1/2)(1-1/4)\cdots(1-1/32)(1-\sum_{i=6}^{\infty}1/2^i)\\&= (1-1/2)(1-1/4)\cdots(1-1/32)(1-1/32)\\&= 0.2887\end{aligned}$$

■

For the case $\mathrm{D}_a = \mathrm{L}_a$, that is, the maximum choice of D_a, $(\mathrm{L}_a/\mathrm{D}_a)\times(\mathrm{L}_a/\mathrm{D}_a)$ matrices over GF(2^{L_a}) degenerate to elements in GF(2^{L_a}), and thus a full-rank $(\mathrm{L}_a/\mathrm{D}_a)\times(\mathrm{L}_a/\mathrm{D}_a)$ matrix over GF(2^{L_a}) degenerates a nonzero element in

TABLE 2. The connection between the P_{D_a} and D_a using (2).

L_a \ D_a	1	2	4	8	16
16	0.288792	0.688541	0.933596	0.996079	0.999985
32	0.288788	0.688538	0.933595	0.996079	0.999985
64	0.288788	0.688538	0.933595	0.996079	0.999985
128	0.288788	0.688538	0.933595	0.996079	0.999985

$\mathrm{GF}(2^{\mathrm{L}_a})$. Every nonzero element in $\mathrm{GF}(2^{\mathrm{L}_a})$ can be represented as a full-rank $\mathrm{L}_a \times \mathrm{L}_a$ matrix over GF(2) according to the matrix representation discussed in Sec. II-A. Consequently, in order to generate an encryption matrix $\mathbf{K}_a$ when $\mathrm{L}_a = \mathrm{D}_a$, one can just randomly generate a nonzero element in $\mathrm{GF}(2^{\mathrm{L}_a})$ and then represent it in the binary matrix form, which is very simple and efficient. As D_a is assumed not equal to 1 here, extending key space is an optional step, where row/column permutations can enrich the candidate set of $\mathrm{L}_a \times \mathrm{L}_a$ full-rank matrices to be generated. Specifically, for a fixed L_a, let $\mathrm{C}_{\mathrm{D}_a}$ denote the number of candidate binary matrices for $\mathbf{K}_a$ with parameter D_a, and $\mathrm{C}'_{\mathrm{D}_a}$ be the number of candidate binary matrices for $\mathbf{K}_a$ with further row operations allowed. Theoretically, $\mathrm{C}'_{\mathrm{D}_a}$ is approximately $\mathrm{L}_a!/(\mathrm{L}_a/\mathrm{D}_a)!$ times larger than $\mathrm{C}_{\mathrm{D}_a}$. In this optional step, the increasing number of $\mathrm{L}_a \times \mathrm{L}_a$ full-rank matrices to be possibly generated has a positive effect on the dynamics and security of the scheme.

Based on the two special choices of D_a, it can be observed that D_a will affect the key space and the computational complexity of encryption. By a similar argument to the case $\mathrm{D}_a = 1$, it can be shown that for a general choice of D_a, when an $(\mathrm{L}_a/\mathrm{D}_a) \times (\mathrm{L}_a/\mathrm{D}_a)$ matrix over $\mathrm{GF}(2^{\mathrm{D}_a})$ is generated, the probability for it to be full rank is

$$P_{D_a} = \prod_{i=0}^{\frac{L_a}{D_a}-1} \frac{2^{L_a} - 2^{i \times D_a}}{2^{L_a}} \tag{2}$$

Table 2 summarizes the calculated P_{D_a} for different L_a and D_a, from which the followings can be observed.

- From every column of the table, it can be seen that the probabilities P_{D_a} for different $\mathrm{L}_a(\geq 16)$ have negligible differences, so that no matter which value L_a (≥ 16) is assigned to, the expected number of randomly generated $\mathrm{L}_a \times \mathrm{L}_a$ matrices over GF(2) for yielding a full-rank one is almost the same. Thus, L_a can be flexibly chosen for the balance of the efficiency and the security.
- The rows of the table justify the statement at the beginning of this section that different choices of D_a with a fixed L_a can endow a transition between the efficiency and security. For instance, when $\mathrm{L}_a = 16$, as D_a increases from 1 to 8, the expected number of matrix generations till a full-rank one is obtained decreases from 3.4627 to 1.0039, so it becomes more efficient to randomly generate the encryption matrix $\mathbf{K}_a$. On the other hand, the cost for the lower generation complexity of encryption matrix $\mathbf{K}_a$ due to larger D_a is the smaller number $\mathrm{C}_{\mathrm{D}_a}$ of candidate matrices for $\mathbf{K}_a$. In particular, an overlarge $\mathrm{D}_a(\geq 16)$ does not help much to enhance the efficiency, while $\mathrm{C}_{\mathrm{D}a}$ will decrease dramatically so that the security level is also decreased. For instance, with $\mathrm{L}_a = 16$ and D_a equal to 8 or 16, the respective expected number of matrix generations till a full-rank one is obtained is 1.0039 and 1.00002, but the respective number of candidate matrices for $\mathbf{K}_a$ becomes $\mathrm{C}_8 = (2^{16}–1)(2^{16}–2^8)$ and $\mathrm{C}_{16} = 2^{16}–1$, where C_{16} is approximately the square root of C_8. In summary, a moderate D_a can provide a tradeoff between the generation complexity of $\mathbf{K}_a$ and the security provided by $\mathbf{K}_a$.

We have justified the process to generate encryption matrix $\mathbf{K}_a$ in step (*a*) and theoretically analyzed the effect of parameters L_a and D_a on the complexity of generating $\mathbf{K}_a$. The justification and analysis are also applicable to encryption matrix $\mathbf{K}_c$ and the concomitant parameters L_c and D_c in step (*c*).

B. NUMERICAL VALIDATION

1) DISCUSSION OF PARAMETER SETTINGS

We first discuss the specific choices of these parameters in practical encryption matrix generating and update scenarios. In the *generating* scenario, the length L_a can be an arbitrary value, but for convenient software realization, choosing 2^{n} as the block length is suggested. Because the purpose of step (*a*) is to effectively defense the exhaustive attack, the block length should be relatively large. The optional values can be 32, 64, 128 etc. If L_a is smaller than 32, it is difficult to defense the exhaustive attack. On the other hand, too large L_a (256 or even larger) will burden the system with tedious operations of high-dimensional matrices.

After we have selected L_a, the chosen D_a must divide L_a. With a proper D_a, the system has low computational complexity and the probability of getting a full-rank matrix is high. It is obvious that D_a is a *compromise* between efficiency and security, and choosing 4 or 8 as the value of D_a is suggested.

As the choices of L_c and D_c in step (*c*) can also be regarded as an interface for dynamic and efficient update in step (*d*), they will be discussed in the following *update* scenario.

In any security scenario, after the encryption process, it is indispensable to update the used key to prevent the key from leaking or cracking. So the key update design of every step is necessary in the scheme. The related key $\mathbf{K}_a$ used in step (*a*), with the related parameters as L_a, D_a and primitive polynomial $p(x)$, and the 64-bit key sequence used in step (*b*) will be optionally changed in the regular key update, while the key used in step (*c*) and updated in step (*d*) will be quite different.

The purpose of step (*c*) is to take advantage of NC to achieve the dynamic mechanism and efficient update, and to resist the man-in-the-middle attack, which is a common and efficient crack in double encryption. To ensure safety, compared with the regular key update in step (*a*) and (*b*), the partial key update in step (*d*) will be much more frequent in order to increase the dynamism, uncertainty and no persistency in the system. Because the partial key in step (*d*) is

TABLE 3. The average time to implement different procedures in the scheme for setting *(a)* $L_a = 64$, $D_a = 1$, $L_c = 16$, $D_c = 1$. *(b)* $L_a = 64$, $D_a = 1$, $L_c = 8$, $D_c = 1$. *(c)* $L_a = 64$, $D_a = 1$, $L_c = 4$, $D_c = 1$

(a)

	Average Time(s)	Ratio
Encryption for the whole scheme	115.24	1
DES in Step (*b*)	78.14	0.68
NC encoding in Step (*a*) and (*c*)	37.10	0.32
Triple DES encryption scheme	234.42	
Decryption for the whole scheme	91.24	1
DES decryption for step (*b*)	78.14	0.86
NC decoding	13.10	0.14
Partial Update	0.004	

(b)

	Average Time(s)	Ratio
Encryption for the whole scheme	116.91	1
DES in Step (*b*)	79.12	0.68
NC encoding in Step (*a*) and (*c*)	37.79	0.32
Triple DES encryption scheme	237.36	
Decryption for the whole scheme	92.80	1
DES decryption for step (*b*)	79.12	0.86
NC decoding	13.68	0.14
Partial Update	0.002	

(c)

	Average Time(s)	Ratio
Encryption for the whole scheme	112.55	1
DES in Step (*b*)	75.71	0.67
NC encoding in Step (*a*) and (*c*)	36.84	0.33
Triple DES encryption scheme	227.13	
Decryption for the whole scheme	88.67	1
DES decryption for step (*b*)	75.71	0.85
NC decoding	12.96	0.15
Partial Update	0.001	

TABLE 4. The average time to implement different procedures in the scheme for setting. *(a)* $L_a = 256$, $D_a = 1$, $L_c = 16$, $D_c = 1$. *(b)* $L_a = 256$, $D_a = 1$, $L_c = 8$, $D_c = 1$. *(c)* $L_a = 256$, $D_a = 1$, $L_c = 4$, $D_c = 1$.

(a)

	Average Time(s)	Ratio
Encryption for the whole scheme	453.48	1
DES in Step (*b*)	80.35	0.18
NC encoding in Step (*a*) and (*c*)	373.13	0.82
Triple DES encryption scheme	241.05	
Decryption for the whole scheme	93.67	1
DES decryption for step (*b*)	80.35	0.86
NC decoding	13.32	0.14
Partial Update	0.004	

(b)

	Average Time(s)	Ratio
Encryption for the whole scheme	434.78	1
DES in Step (*b*)	76.73	0.18
NC encoding in Step (*a*) and (*c*)	358.05	0.82
Triple DES encryption scheme	230.19	
Decryption for the whole scheme	89.76	1
DES decryption for step (*b*)	76.73	0.85
NC decoding	13.03	0.15
Partial Update	0.002	

(c)

	Average Time(s)	Ratio
Encryption for the whole scheme	441.04	1
DES in Step (*b*)	77.37	0.18
NC encoding in Step (*a*) and (*c*)	363.67	0.82
Triple DES encryption scheme	232.11	
Decryption for the whole scheme	90.49	1
DES decryption for step (*b*)	77.37	0.85
NC decoding	13.12	0.14
Partial Update	0.001	

updated frequently, the length L_c should be relatively small to reduce the computational complexity. The optional values can be 32 or 16. After L_c has been chosen, we can choose D_c, which is a divisor of L_c and primitive polynomial $p(x)$ to achieve the flexibility of the encryption. Obviously, D_c is rather small and there are much less primitive polynomials for the small extension field compared with the ones in step (a).

2) NUMERICAL RESULTS

The simulations have been performed in Matlab (ver. R2014a) under different parameter settings in order to numerically validate the encryption and decryption complexity of the proposed scheme. For simplicity, both parameters D_a and D_c are set to 1, in which cases complexity to generate encryption matrices $\mathbf{K}_a$ and $\mathbf{K}_c$ are highest. The length of plaintext equals to 1Mb.

Table 3 and 4 list the average time to implement different procedures in the scheme for $L_a = 64$ and 256, respectively, under 100 independent experiments. In each table, there are three separate subtables (a)-(c) for $L_c = 16$, 8 and 4. The unit of "average time" is second, and the "average time" is obtained based on the function $etime(t_\alpha, t_\beta)$ which calculates the time difference between two time date t_α and t_β.

First note that with the same L_c and L_a increasing from 64 to 256 in Table 3 and 4, the time cost involving the matrix operations in an encryption part like *Encryption for the whole scheme* and *NC encoding in Step*(a) *and* (c) increase very fast with the key space extending, but the one in a decryption part like *Decryption for the whole scheme* and *NC decoding* keeps basically unchanged. The reason is that the *NC encoding in Step*(a) *and* (c) function includes both the process of repeatedly generating random matrices till they are full-rank and the computation of invertible matrices of encryption matrices. This function also returns both the encryption and decryption matrices to the main function.

Next, from both Table 3 and Table 4, it can be observed that the changes of L_c have little impact on the time cost to almost all procedures in the scheme and they only influence on the update complexity.

When $L_a = 64$, the ratio of NC encoding time over the encryption time of the whole scheme is 0.33, and the NC decryption time over the decryption time of the whole scheme is 0.14. The ratio of the total NC encoding and decryption time over the total encryption and decryption time of the whole scheme is 0.24. Similarly, when $L_a = 256$, the above mentioned ratios are 0.82, 0.14 and 0.71 respectively. Besides, from the tables we can also calculate the ratios between the triple DES encryption time over the encryption time of the whole scheme are 2.03 and 0.53 for $L_a = 64$ and 256 respectively. The results indicate that even though the simulations in Matlab adopt the "low-efficiency" modular operations and do not implement the matrix representation which can reduce the encryption complexity of the system, the running ratio which NC takes is relatively lower than or comparable to DES and the triple DES in the encryption process and the whole encryption scheme is feasible.

We finally remark that the simulation time in Matlab is rather long even for a 1Mb plaintext due to the low code execution efficiency of software, tedious program initialization and omitted core matrix representation with $D_a = 1$ and $D_c = 1$. Still, the ratios in the Tables to embody the comparison in NC and DES are fair and indicative, because we simulate NC and DES in the same software environment.

V. BENEFIT DISCUSSION OF THE SCHEME

In Section II, we mentioned that there are many proposed algorithms in order to improve DES, such as: multiple DES, mutable S-box DES, sub key DES, G-DES, DES-X, s^nDES and so on. In this section, we shall give a brief description to these algorithms and present comparisons between our encryption scheme from the viewpoint of performance and complexity.

1) Multiple DES: The most classical multiple DES algorithm is the triple DES which we have discussed in Section V. The complexity of multiple DES is several times larger than single DES and therefore much larger than our encryption scheme. Besides, as we said at the beginning, multiple DES cannot meet the dynamic security requirements of the intelligent information network.

2) Mutable S-box DES: This algorithm can change the content order of S-box based on the change of encryption key or directly change the content of S-box. Obviously, it can be used to resist differential cryptanalysis (DC) but has no contribution to enlarge the key space.

3) Sub key DES: This algorithm uses different sub key on every iteration during encryption in DES. Due to 48-bit key required in every iteration process, after 16 iterations, the key space of this improved DES is 768. This algorithm greatly increases the complexity of key space and has a good behavior to resist the exhaustive attack. However, its adaptability and extension with the rapid development of the computing power is even less than the multiple DES algorithm for the reason that its key space is strictly static.

Other DES improved algorithms like G-DES, DES-X and s^nDES are more complicated than the above three algorithms and their encoding complexity is much larger than the single DES. Compared with these algorithms, the encoding scheme we proposed uses NC and the running ratio which NC takes is relatively lower than or comparable to DES. Moreover, from the viewpoint of performance, the NC nature of the proposed scheme makes it endow the dynamic, active and random characteristics in the concept of Moving Target Defense (MTD).

Based on the above-mentioned theoretical analysis, simulation validation and complexity comparison, we summarize the following specific benefits of the proposed dynamic encryption scheme as follows:

- With inosculating the respective advantages of DES and NC, the scheme forms a triple encryption—different sizes of linear NC encryption matrices protecting the non-linear DES on both sides. No matter which direction (former or latter) the attackers choose to perform an exhaustive attack, there is not an effective way under the current restricted computing power. Meanwhile, the triple encryption structure can defense the Man-in-the-middle attack effectively.
- The regular key update with frequency f_r will totally change the keys in every update cycle (including the structure and elements inside the encryption matrices and the key sequences of DES); moreover, in order to increase the dynamic and security level of the system, the partial key update with frequency f_p, which is much higher than f_r, will partially change the key without changing the structure of the matrices. The dynamic security is achieved by combining the low-frequency regular key update and the high-frequency partial key update. As a result, different from the triple DES and AES, most update processes in the new scheme just perform matrix multiplications, whose complexity is low enough for the dynamic update process, so that dynamic characteristic of MTD is embodied in the new scheme.
- The scheme has a better adaptability and extension with the rapid development of the computing power. In contrast, the encryption systems such as the triple DES and AES cannot extend the designed key space. (Recall that there are three optional key spaces in AES: 128, 192, 256, but neither the data block length nor the maximum length of the key space can be flexibly changed.) However, the present encryption scheme can actively extend the size of the encryption matrices, such as L_a, L_c etc., so as to enhance the resistance to the exhaustive attack with increasing computing power. This reflects the active characteristic of MTD.
- By changing the related parameter pairs in the related encryption steps, such as (L_a, D_a), (L_c, D_c) and (f_r, f_p), the scheme can achieve a transition between efficiency and security, which increases its adaptability to various network conditions. This reflects the flexible characteristic of MTD.
- This scheme can assist to recognize the polluted data packets because in the present encryption scheme, the probability of getting a meaningful plaintext from a tampered data packet is very low. First, the randomness of the encryption scheme guarantees that the attackers cannot give a targeted attack. Second, multiple encryptions ensure that when the attackers tamper the ciphertext, the effect on the plaintext decoded by this ciphertext will be random and unknown. If the plaintext decoded by the receivers is not logical and meaningful, the system will regard the packet as a tampered data packet and then abandon it.
- At last, it is well known that the pattern of the triple DES is DED or EDE (E means the encryption operation and D means the decryption operation) but not DDD or EEE, so that it is compatible with single DES having the same

key as in the triple processes. Similar to the triple DES, the present encryption scheme has the same feature with the identity matrices as keys in step (a) and step (c). In this way, the scheme is feasible and can be practically applied to protect the systems which still use single DES for now due to the unachievable equipment replacement.

VI. CONCLUDING REMARKS

In this paper, we proposed a novel encryption scheme which combines both the DES and the network coding characteristic, which has good behavior to resist both exhaustive and analysis attacks. The simulation results show that the running ratio of the proposed scheme is relatively lower than or comparable to the triple DES. The NC nature of the proposed scheme makes it endow the dynamic, active and random characteristics in the concept of Moving Target Defense (MTD). The security level of the proposed scheme will be tested in our future work.

REFERENCES

[1] S. Jajodia, A. K. Ghosh, V. Swarup, C. Wang, and X. S. Wang, *Moving Target Defense Creating Asymmetric Uncertainty for Cyber Threats*. Berlin, Germany: Springer, 2011.

[2] S. Jajodia, A. K. Ghosh, V. Subrahmanian, V. Swarup, C. Wang, and X. S. Wang, "Application of game theory and adversarial modeling," in *Moving Target Defense II* (Advances in Information Security). New York, NY, USA: Springer, 2013.

[3] M. Carvalho and R. Ford, "Moving-target defenses for computer networks," *IEEE Security Privacy*, vol. 2, no. 12, pp. 73–76, Mar. 2014.

[4] W. Peng, F. Li, C.-T. Huang, and X. Zou, "A moving-target defense strategy for Cloud-based services with heterogeneous and dynamic attack surfaces," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Sydney, NSW, Australia, Jun. 2014, pp. 804–809.

[5] A. D. Keromytis, R. Geambasu, and S. Sethumadhavan, "Themeerkats cloud security architecture," in *Proc. IEEE Int. Conf. Distrib. Comput. Syst. Workshops*, Macau, China, Jun. 2012, pp. 446–450.

[6] S. G. Vadlamudi, S. Sengupta, and S. Kambhampati, "Moving target defense for Web applications using Bayesian stackelberg games," in *Proc. Int. Conf. Auto. Agents Multiagent Syst.*, Singapore, May 2016, pp. 1377–1378.

[7] M. Taguinod, A. Doupé, Z. Zhao, and G.-J. Ahn, "Toward a movingtarget defense for Web applications," in *Proc. IEEE Int. Conf. Inf. Reuse Integr. (IRI)*, San Francisco, CA, USA, Aug. 2015, pp. 510–517.

[8] L. Z. Gu, Z. H. Zheng, and Y. X. Yang, *Modern Cryptography*. Beijing, China: Beijing University of Posts and Telecommunications, 2015.

[9] W. Stallings, *Cryptography and Network Security Principles and Practice*, 5th ed. Upper Saddle River, NJ, USA: Prentice-Hall, 2010.

[10] A. Hevia and M. Kiwi, "Strength of two data encryption standard implementations under timing attacks," *ACM Trans. Inf. Syst. Secur.*, vol. 2, pp. 416–437, Nov. 1999.

[11] X. Wang and R. Zeng, "The analysis and improvement of DES algorithm," *J. Shiyan Tech. Inst.*, vol. 19, no. 5, pp. 84–86, Oct. 2006.

[12] B. Jiang, "Analysis of DES algorithm implementation and improvement process," *J. Langfang Teachers College*, vol. 10, no. 5, pp. 46–47, Oct. 2010.

[13] J. X. Gao, "Implementation and improvement of DES algorithm," *Netw. Secur. Technol., Appl.*, vol. 14, no. 1, pp. 61–62, Jan. 2014.

[14] S. R. Li, Q. T. Sun, and Z. Shao, "Linear network coding: Theory and algorithms," *Proc. IEEE*, vol. 99, no. 3, pp. 372–387, Mar. 2011.

[15] P. F. Oliveira and J. Barros, "A network coding approach to secret key distribution," *IEEE Trans. Inf. Forensics Security*, vol. 3, no. 3, pp. 414–423, Sep. 2008.

[16] P. Zhang, C. Lin, Y. Jiang, Y. Fan, and X. Shen, "A lightweight encryption scheme for network-coded mobile ad hoc networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 9, pp. 2211–2221, Sep. 2014.

[17] B. Schneier, *Applied Cryptography*. New York, NY, USA: Wiley, 1996, pp. 873–874.

[18] C. Paar and J. Pelzl, *Understanding Cryptography: A Textbook for Students and Practitioners*. Berlin, Germany: Springer, 2009.

[19] S.-Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Trans. Inf. Theory*, vol. 49, no. 2, pp. 371–381, Feb. 2003.

[20] R. Koetter and M. Médard, "An algebraic approach to network coding," *IEEE/ACM Trans. Netw.*, vol. 11, no. 5, pp. 782–795, Oct. 2003.

[21] A. G. Dimakis, K. Ramchandran, Y. Wu, and C. Suh, "A survey on network codes for distributed storage," *Proc. IEEE*, vol. 99, no. 3, pp. 476–489, Mar. 2011.

[22] C. Gkantsidis and R. P. Rodriguez, "Network coding for large scale content distribution," in *Proc. IEEE INFOCOM*, Miami, FL, USA, Mar. 2005, pp. 2235–2245.

[23] W. P. Wardlaw, "Matrix representation of finite fields," *Math. Mag.*, vol. 67, pp. 289–293, Oct. 1994.

[24] S. D. Dummit and M. R. Foote, *Abstract Algebra*. 2nd ed. New York, NY, USA: Wiley, 1999.

HANQI TANG received the B.S. degree in computer science from the University of Science and Technology Beijing in 2015, where he is currently pursuing the D.S. degree with the School of Computer and Communication Engineering. His research interests include network coding and communication security.

QIFU TYLER SUN received the B.Eng. (Hons.) and Ph.D. degrees from the Department of Information Engineering, The Chinese University of Hong Kong, Hong Kong, in 2005 and 2009, respectively. He was a Post-Doctoral Fellow with the Institute of Network Coding, The Chinese University of Hong Kong, and also a Visiting Research Fellow with the University of New South Wales. He is currently an Associate Professor with the School of Computer and Communication Engineering, University of Science and Technology Beijing, China. His research interests include network coding and coding theory. He has been holding, as the principal investigator, three research grants of National Science Foundation of China.

XIAOLONG YANG received the B.Eng., M.S., and Ph.D. degrees in communication and information system from the University of Electronic Science and Technology of China, Chengdu, China, in 1993, 1996, and 2004, respectively. He is currently a Professor with the School of Computer and Communication Engineering, Institute of Advanced Networking Technologies and Services, University of Science and Technology Beijing, Beijing, China. His research focuses on the next-generation Internet, data center networking, and network security. He has fulfilled over 30 research projects, including the National Natural Science Foundation of China, National Hi-Tech Research and Development Program (863 Program), and National Key Basic Research Program (973 Program). In his research field, he authored over 80 papers and holds 16 patents.

KEPING LONG received the Ph.D. degree from the University of Electronic Science and Technology of China, Chengdu, China, in 1997. From 1998 to 2000, he was a Post-Doctoral Research Fellow with the National Laboratory of Switching Technology and Telecommunication Networks, Beijing University of Posts and Telecommunications, Beijing. From 2001 to 2002, he was a Research Fellow with the ARC Special Research Center for Ultra Broadband Information Networks, The University of Melbourne, Australia. He was elected to Chang Jiang Scholars Program of the Ministry of Education of China in 2009. He is currently a Professor with the University of Science and Technology Beijing, Beijing, China. His research interests include the theory and technology of communication networks.