SCIENCE CHINA
Information Sciences

• **RESEARCH PAPER** •

October 2016, Vol. 59 102309:1–102309:15
doi: 10.1007/s11432-015-5452-1

CodeHop: physical layer error correction and encryption with LDPC-based code hopping

Zhao CHEN[1,3], Liuguo YIN[2,3]*, Yukui PEI[1] & Jianhua LU[1]

[1]*Department of Electronic Engineering, Tsinghua University, Beijing* 100084*, China;*
[2]*School of Aerospace, Tsinghua University, Beijing* 100084*, China;*
[3]*Electronic Design Automation (EDA) Laboratory, Research Institute of Tsinghua University in Shenzhen, Shenzhen* 518057*, China*

Received July 3 2015; accepted July 30, 2015; published online March 30, 2016

Abstract This paper proposes a novel scheme named CodeHop, which provides both information reliability and security using code hopping based on low-density parity-check (LDPC) codes. In contrast to traditional systems that perform error correction and encryption at different layers, CodeHop combines these two operations into a single step at physical layer, such that each plaintext message is jointly encoded and encrypted by a hopping parity-check matrix. According to a pseudo-random number generator (PRNG), the hopping matrix may rapidly switch among a sequence of LDPC parity-check matrices, which is randomly generated by a structured-random protograph expanding technique. Simulations show that reliable communication can be achieved by CodeHop with good error-correcting performance. In the meantime, CodeHop may improve the security of traditional systems such as GSM. Taking the A5/1 stream cipher used in GSM as the PRNG, it is shown that CodeHop is resistant to existing chosen-plaintext attacks that break A5/1 cipher already. Moreover, the security of CodeHop will be enhanced in the presence of channel errors as well.

Keywords physical layer security, error correction, data encryption, low-density parity-check codes, code hopping

Citation Chen Z, Yin L G, Pei Y K, et al. CodeHop: physical layer error correction and encryption with LDPC-based code hopping. Sci China Inf Sci, 2016, 59(10): 102309, doi: 10.1007/s11432-015-5452-1

1 Introduction

Information reliability and security are both critical issues in wireless communications [1]. Due to the broadcast nature of wireless medium, there are channel noise and potential wiretappers over wireless channels. As shown in Figure 1, traditional communication systems cope with channel noise by error-correcting codes at physical layer and defend against wiretappers by cryptographic algorithms at upper layers [2]. Thus, it is assumed that channel errors are removed at physical layer and the received ciphertext at upper layers is error-free for both Bob and Eve. Therefore, the security of information only relies on data encryption in upper layers, which, however, cannot be perfectly guaranteed since it has been proved that some widely used ciphers are not secure any more, such as A5/1 stream cipher used in GSM [3].

*Corresponding author (email: yinlg@tsinghua.edu.cn)

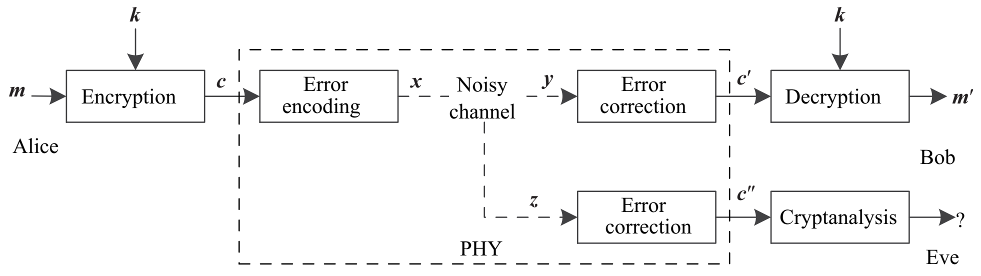

Figure 1 Block diagram of a traditional communication system.$^{1)}$

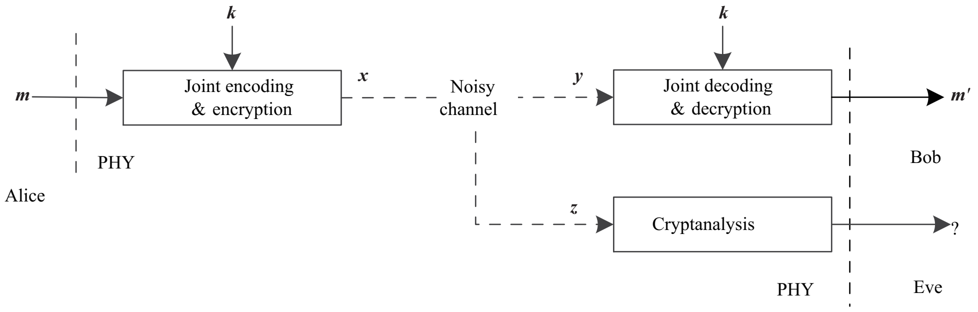

Figure 2 Block diagram of a joint channel coding and encryption scheme.

Instead of handling reliability and security separately at different layers, joint design of channel coding and encryption emerges recently, aiming to tackle these two critical issues together at physical layer. For instance, information-theoretic physical layer security [1] has received much attention after Wyner's work of wiretap channel [4], where it has been proved that perfect secrecy is feasible if the channel of the legitimate receiver (Bob) is less noisy than that of the eavesdropper (Eve). Besides, the supremum of all the achievable secure transmission rates is characterized by secrecy capacity. There are several coding techniques developed to approach the secrecy capacity of wiretap channels, especially for the application of low-density parity-check (LDPC) codes to binary erasure channel (BEC) [5], binary symmetric channel (BSC) [6], and Gaussian channel [7]. However, restricted by the superior channel condition for Bob, perfect secrecy is so rigorous that secrecy capacity varies frequently over wireless channels. In some scenarios, it will be very low or even zero. Hence, in this paper, security will be addressed at physical layer from a cryptographic perspective, since it has been shown in the literature that error-correcting codes and encryption algorithms can be combined together as shown in Figure 2, which can effectively improve the security level or reduce the power assumption or hardware usage of the whole system.

Random channel errors in received ciphertexts of Eve are exploited by joint channel coding and encryption schemes to improve system security by making the cryptanalysis of Eve more difficult. In [8], Zuquete et al. swapped the order of encryption and channel coding by first encoding plaintexts to codewords and then encrypting them with a stream cipher, which will be resistant to known-plaintext attacks because of channel errors. But if the channel of Eve becomes noiseless, a ciphertext-only attack is feasible by using the redundancy in codewords [9]. In [10], Harrison discussed how to exploit the effect of error amplification when stopping sets occur in LDPC decoding, which requires Bob's channel to be better than Eve's. Due to the diffusion property of block ciphers [11], the resulting error propagation effect will randomly change half number of bits in the decrypted plaintext, which weakens the reliability of transmission in block-ciphered systems. With sacrificing the bit error rate (BER) after decryption, Wei et al. [12] considered a trade-off between security and reliability, which improves the security in terms of increasing the required plaintext-ciphertext pairs in known-plaintext attacks for the cipher feedback (CFB) mode of Data Encryption Standard (DES).

Some other joint schemes can obtain shorter processing time or more efficient implementation, with

1) Note that Alice, Bob and Eve in the figure represent sender, legitimate receiver and eavesdropper, respectively.

combining channel coding and encryption in a single step. Mathur et al. [13] designed the high diffusion codes as a substitute for the diffusion layer of advanced encryption standard (AES) [14], which can correct channel errors with built-in security features. In [15], Adamo et al. presented the error correction based cipher (ECBC), which takes a fixed and private generator matrix as the secret key and a pseudo-random sequence as the artificial noise. The artificial noise will be expanded by a block cipher to confuse Eve, which, however, is broken by Chai [16].

Considering the joint schemes mentioned above, there are two categories, depending on whether the parity-check matrix is private or not. If it is public, channel errors will be beneficial when Bob's channel condition is better than Eve's, otherwise a trade-off between security and reliability is required since channel errors can influence both Bob and Eve. If the matrix becomes private, security can be improved without affecting the error correcting performance. However, the parity-check matrix is possible to be recovered under chosen-plaintext attacks, due to the fact that channel coding is a linear transformation and the parity-check matrix is fixed for all codewords, which will be proved in Subsection 5.1.

In this paper, motivated by the widely applied techniques of frequency hopping, time hopping and beam hopping [17], we extend the concept of hopping to channel coding and propose a novel scheme named CodeHop, in which each plaintext message is jointly encoded and encrypted by a hopping parity-check matrix of LDPC codes. Synchronized by a pair of pseudo-random number generators (PRNG) at Alice and Bob, the parity-check matrix will fast switch among a sequence of LDPC parity-check matrices, which is secretly generated using a structured-random protograph expanding technique. Compared to traditional systems, CodeHop can be regarded as a cross-layer security scheme [18], which may improve system security without sacrificing error correcting performance. Besides, CodeHop outperforms joint schemes in the literature, since it is resistant to known or chosen plaintext attacks owing to the hopping property of parity-check matrices. Therefore, information reliability and security can be both successfully guaranteed at physical layer by CodeHop.

The rest of the paper is organized as follows. In Section 2, the framework of CodeHop is described in detail. Then, a structured-random expanding technique for the hopping sequence of LDPC parity-check matrices is discussed in Section 3. After that, the encoder and decoder design Schemes are proposed in Section 4. In Section 5, cryptanalysis of CodeHop under chosen-plaintext attacks is presented. Finally, Section 6 concludes the paper.

2 Framework of CodeHop

As is known, frequency hopping and time hopping are introduced against jamming or eavesdropping by rapidly switching the carrier frequency or transmission time slot according to a pseudo-random number sequence. Inspired by these ideas, CodeHop is proposed to employ hopping in channel coding such that LDPC coding with a fast hopping sequence of secret parity-check matrices $\boldsymbol{H}_1, \boldsymbol{H}_2, \ldots, \boldsymbol{H}_i$ is performed, which combines the operations of encryption and error correction in a single step. Specifically, each plaintext message $\boldsymbol{m}_i$ will be encoded and encrypted by a unique parity-check matrix $\boldsymbol{H}_i$ to obtain a codeword $\boldsymbol{x}_i$ as the ciphertext. Then, each ciphertext $\boldsymbol{x}_i$ will be transmitted over a noisy channel and received by Bob as $\boldsymbol{y}_i$, which can be jointly decoded and decrypted by $\boldsymbol{H}_i$ to obtain the plaintext $\widehat{\boldsymbol{m}}_i$.

As for the eavesdropper Eve, each ciphertext is received as erroneous $\boldsymbol{z}_i$ which will be analyzed by him to recover the plaintext without knowing $\boldsymbol{H}_i$. To avoid leakage of plaintext messages in the information bits of systematic codewords, nonsystematic codes are necessary in CodeHop. Then, it is impossible for Eve to directly recover the plaintext $\boldsymbol{m}_i$ from codewords. Furthermore, CodeHop is immune to chosen-plaintext attacks, when the hopping of parity-check matrices $\boldsymbol{H}_i$ is controlled by a proper pseudo-random number generator (PRGN). Therefore, the reliability and security of information can be guaranteed at the same time by the proposed CodeHop.

For a session of secure transmission, the block diagram of CodeHop is illustrated in Figure 3. To be clear, all the notations and procedures in the figure are explained as follows.

- The secret message source to be transmitted by Alice is divided into k-bit plaintext messages $\boldsymbol{m}_i$

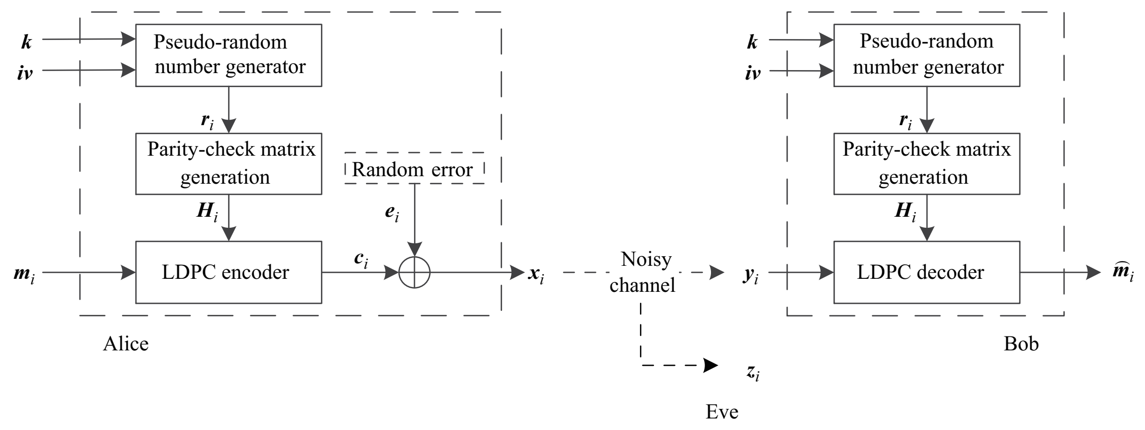

Figure 3 Block diagram of the proposed CodeHop. Note that Eve will try his best to recover the plaintext message $\boldsymbol{m}_i$ without knowing the parity-check matrix $\boldsymbol{H}_i$.

for $i = 1, 2, 3, \ldots$, which are then expected to be correctly and securely recovered by Bob.

• With a session key $\boldsymbol{k}$ and an initialization vector $\boldsymbol{iv}$, the PRNG produces a binary keystream, which is divided into h-bit vectors $\boldsymbol{r}_i$ and then used to synchronize the hopping sequence of parity-check matrices $\boldsymbol{H}_i$ between Alice and Bob.

• In the parity-check matrix generator (PCMG), every secret parity-check matrix $\boldsymbol{H}_i = \boldsymbol{H}(\boldsymbol{r}_i)$ is generated according to the keystream vector $\boldsymbol{r}_i$ using a structured-random protograph expanding technique, which will be shown completely in Section 3.

• At Alice, each plaintext $\boldsymbol{m}_i$ is encoded and encrypted by a generator matrix $\boldsymbol{G}_i$ to obtain an n-bit codeword $\boldsymbol{c}_i$ via LDPC encoding. Then, the ciphertext $\boldsymbol{x}_i$ is given by

$$\boldsymbol{x}_i = \boldsymbol{c}_i \oplus \boldsymbol{e}_i = \boldsymbol{m}_i \cdot \boldsymbol{G}_i \oplus \boldsymbol{e}_i, \tag{1}$$

which is produced by disturbing the codeword $\boldsymbol{c}_i$ with an intentional random error vector $\boldsymbol{e}_i$. Note that the generator matrix $\boldsymbol{G}_i$ is converted from $\boldsymbol{H}_i$ during LDPC encoding.

• The ciphertext $\boldsymbol{x}_i$ is broadcasted over a noisy channel, which then will be received by Bob and Eve as erroneous ciphertexts $\boldsymbol{y}_i$ and $\boldsymbol{z}_i$, respectively.

• At Bob, the erroneous ciphertext $\boldsymbol{y}_i$ is decoded and decrypted by the synchronized hopping parity-check matrix $\boldsymbol{H}_i$ to obtain the plaintext message $\widehat{\boldsymbol{m}}_i$ via LDPC decoding. Meanwhile, without knowing $\boldsymbol{H}_i$, Eve will fail to decode and decrypt $\boldsymbol{z}_i$.

A5/1 stream cipher is considered for the PRNG, which is standardized in global system for mobile communications (GSM) and widely used all over the world. As known for its low hardware implementation complexity, A5/1 is a linear feedback shift register (LFSR) based stream cipher, which has been analyzed by cryptographers for years. For example, correlation attack proposed by Barkan et al. [3] is one of the best chosen-plaintext attack. However, it will be shown that CodeHop can improve the security level of the original A5/1 by joint channel encoding and encryption with LDPC codes. Note that CodeHop can be also generalized for block ciphers such as AES in operational modes like CFB or output feedback (OFB), which is much more complicated to implement.

The session key $\boldsymbol{k}$ in CodeHop and A5/1 cipher is the same, which is assumed to be privately agreed between Alice and Bob. Besides, the initialization vector $\boldsymbol{iv}$ contains information about initial state of the PRNG and general structure of the hopping sequence of parity-check matrices, which is required to be non-repeating for different sessions and can be transmitted to Bob publicly in the beginning of a session.

3 Structured-random LDPC codes

In this section, we will show how to construct a hopping sequence of LDPC parity-check matrices for CodeHop via structured-random protograph expanding. This problem will be tackled in the following four parts. Firstly, compared with systematic codes, nonsystematic LDPC codes are analyzed how to avoid leakage of plaintext messages. Then, the protograph expanding technique [19] is extended to construct a large set of LDPC parity-check matrices in a structured-random manner, which can be applied to generate the hopping sequence. Moreover, the implementation of LDPC encoders for hopping generator matrices is discussed. Lastly, the structured-random technique is demonstrated and evaluated with an example.

3.1 Systematic vs. nonsystematic

To start with, it is necessary to explain why nonsystematic LDPC codes are needed for CodeHop. If systematic LDPC codes are adopted, the plaintext message $\boldsymbol{m}_i$ will be included in the ciphertext $\boldsymbol{x}_i$ as information bits and then transmitted over the channel. Even though the received ciphertext $\boldsymbol{z}_i$ is erroneous, Eve is possible to recover $\boldsymbol{m}_i$ without LDPC decoding when channel condition is good enough and without knowing the secret parity-check matrix $\boldsymbol{H}_i$. Therefore, systematic LDPC codes are not resistant to ciphertext-only attacks.

Assume a rate-1/2 channel code over the additive white gaussian noise (AWGN) channel. In Figure 4, BER curves of plaintext messages leaked to Eve for systematic and nonsystematic codes are illustrated. For nonsystematic codes, the plaintext message $\boldsymbol{m}_i$ is not transmitted in the ciphertext $\boldsymbol{x}_i$, which implies the BER of leaked $\boldsymbol{m}_i$ will be always 0.5. However, for systematic codes, although the codeword $\boldsymbol{z}_i$ is hard decided from modulation symbols without LDPC decoding, $\boldsymbol{m}_i$ is transmitted in $\boldsymbol{x}_i$ and there is just about 3 dB degradation in E_b/N_0 for the BER of leaked $\boldsymbol{m}_i$ to Eve compared with uncoded BPSK. From the figure, it can be seen that the BER of leaked $\boldsymbol{m}_i$ will be lower than 10^{-6} when $E_b/N_0 > 14$ dB, which means that the plaintext messages $\boldsymbol{m}_i$ will be completely recovered by Eve in high E_b/N_0 regions. Therefore, nonsystematic codes should be considered to avoid the transmission of information bits $\boldsymbol{m}_i$.

3.2 Structured-random protograph expanding

In this part, we will construct a large set of nonsystematic LDPC codes via structured-random protograph expanding. The concept of protograph was introduced by Thorpe in [19], which can be regarded as the minimal base Tanner graph to describe an LDPC code. Using protograph, the accumulate-repeat-accumulate (ARA) [20] code is constructed and standardized by CCSDS [21], which is known as one of the best LDPC code. Different from the systematic ARA code, our target is to puncture all of the k information bits and transmit only the n parity bits in the original $(n+k)$-bit codeword. However, without the help of information bits, the iterative decoding of nonsystematic codes is difficult to converge. Fortunately, Brink investigated the design of nonsystematic RA codes in [22]. As shown in Figure 5, it is proved that iterative decoding can be triggered by the method of code doping, which requires that there exits at least one accumulate node satisfying $d_{c,i} = 1$ for $i = 1, \ldots, n$.

As for a rate-1/2 nonsystematic LDPC code, the result of an optimized protograph $P = (V, C, E)$ is give in Figure 6, which is equivalent to a 4×6 base parity-check matrix $\boldsymbol{H}_{\mathrm{B},0}$ as shown below,

$$\boldsymbol{H}_{\mathrm{B},0} = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 3 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 2 & 0 & 0 \end{bmatrix}. \tag{2}$$

Note that the elements in the base parity-check matrix $\boldsymbol{H}_{\mathrm{B},0}$ indicate the number of parallel edges between a variable node and a check node in the protograph P. There are a variable node set $V = \{v_1, v_2, ..., v_6\}$, a check node set $C = \{c_1, c_2, ..., c_4\}$ and an edge set $E = \{e_1, e_2, ..., e_{16}\}$ in the protograph P. Among all of these variable nodes, the information nodes denoted by v_1 and v_2 will be punctured. Therefore, to trigger the convergence of the iterative decoding, the check node c_4 is designated to be connected to only one punctured variable node v_2 according to the requirement of code doping.

Starting with the protograph P, a larger Tanner graph can be obtained by copy-and-permute operation. Let each edge $e \in E$ stand for a unique edge type. If expanded with a factor of T, the copy-and-permute

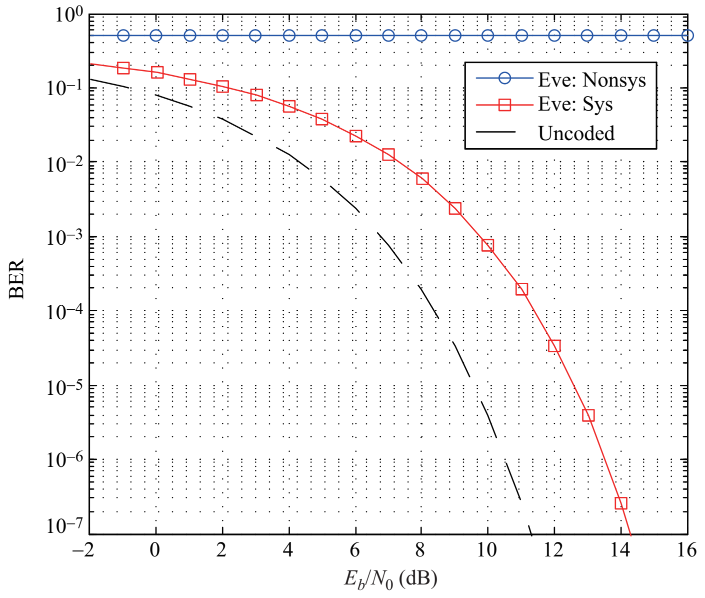

Figure 4 (Color online) BER curves of plaintext messages leaked to Eve for rate-1/2 systematic (Sys) and nonsystematic (Nonsys) codes, which are BERs of information bits hard decided from modulation symbols without channel decoding.

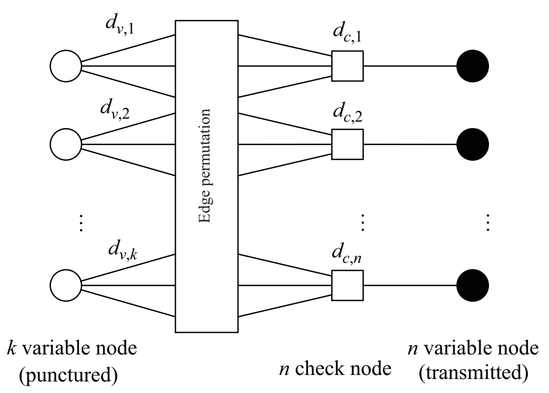

Figure 5 Conceptual protograph for the nonsystematic RA code [22].

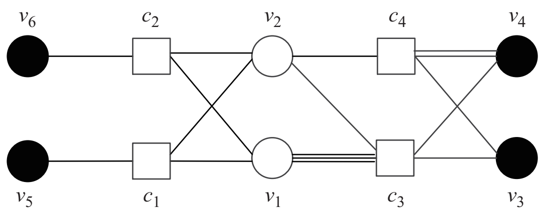

Figure 6 The optimized protograph P for the rate-1/2 nonsystematic LDPC code.

operation firstly replicates the protograph P for T times and forms a set of T edge copies for each edge type, whose endpoints are then permuted among the variable and check nodes in the set. Thus the T copies of the protograph are all interconnected in the derived graph, which defines the Tanner graph for the derived code. Note that the operation of edge expanding is equivalent to matrix expanding for each element in the base matrix $\boldsymbol{H}_{\mathrm{B},0}$. Each element of value w will be expanded to a $T \times T$ matrix with w ones in each row or column, and thus the equivalent parity-check matrix can be of size $4T \times 6T$.

Instead of using random permutations on the set of each edge type, permutations defined by algebraic structures such as cyclic shift are preferred for the ease of description and efficient implementation. That is to say, the protograph can be expanded by choosing appropriate $T \times T$ circulant permutation matrices (CPMs) $\boldsymbol{I}_T(t)$ for each edge type, and the derived parity-check matrix will become a T-circulant matrix, which can be regarded as Quasi-Cyclic LDPC (QC-LDPC) codes [23, 24]. Note that each row in $\boldsymbol{I}_T(t)$ can be obtained by one bit cyclic right shifting its previous row, and thus $\boldsymbol{I}_T(t)$ is defined by its first row $\boldsymbol{u}_T(t)$, where $\boldsymbol{u}_T(t) = (0, \ldots, 0, 1, 0, \ldots, 0)$ and the shift value $t \in [\![0, T-1]\!]$ [2) denotes the position of the unique one in the vector. Therefore, given the protograph P, the derived LDPC code can be described

2) Note that $[\![a, b]\!] := \{a, a+1, \ldots, b\}$, which denotes the set of integers between a and b.

by shift values of all the CPMs.

Usually searching for only one good code is satisfactory for channel coding. However, now it is far more challenging to construct a large number of highly efficient LDPC codes for CodeHop to guarantee both reliability and security. Expanding with just one single stage is not enough, so we solve this problem with a structured-random protograph expanding technique, which can be referred to as a multi-stage expanding scheme [25]. Specifically, the protograph P will be expanded via L stages such that for $l \in [\![1, L]\!]$, the total expanding factor T satisfies

$$T = T_1 T_2 \cdots T_l \cdots T_L, \tag{3}$$

where $L > 1$ and each T_l is the expanding factor in the l-th expanding stage, respectively. Equivalently, the base matrix $\boldsymbol{H}_{\mathrm{B},0}$ will also be expanded and the expanded matrix in the l-th expanding stage is denoted by $\boldsymbol{H}_{\mathrm{B},l}$. Note that all the L expanding stages can be divided into two categories as follows.

• Structured expanding: In the first $L-1$ stages, the protograph is carefully expanded to avoid short loops, low-weight codewords or parallel edges, which can restrict the general structure of the code. After $L-1$ stages, all the non-zero elements in $\boldsymbol{H}_{\mathrm{B},L-1}$ will equal 1.

• Random expanding: In the L-th stage, all of the edges in the Tanner graph will be randomly expanded, or equivalently, all the non-zero elements in $\boldsymbol{H}_{\mathrm{B},L-1}$ will be expanded to CPMs according to the keystream vector $\boldsymbol{r}_i$ produced by the PRNG.

After expanding the protograph P in Figure 6, each parity-check matrix $\boldsymbol{H}_i$ in the hopping sequence is a T_L-circulant matrix with a size of $n \times (n+k)$, which can be written as $\boldsymbol{H}_i = [\boldsymbol{A}(\boldsymbol{r}_i), \boldsymbol{B}(\boldsymbol{r}_i)]$ such that

$$\boldsymbol{A}(\boldsymbol{r}_i) = \left[\boldsymbol{A}^w_{\alpha\beta}\right]_{2\times 4} = \begin{bmatrix} \boldsymbol{A}^1_{11} & \boldsymbol{A}^1_{12} \\ \boldsymbol{A}^1_{21} & \boldsymbol{A}^1_{22} \\ \boldsymbol{A}^3_{31} & \boldsymbol{A}^1_{32} \\ \boldsymbol{0} & \boldsymbol{A}^1_{42} \end{bmatrix}, \tag{4}$$

$$\boldsymbol{B}(\boldsymbol{r}_i) = \left[\boldsymbol{B}^w_{\alpha\beta}\right]_{4\times 4} = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{B}^1_{13} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{B}^1_{24} \\ \boldsymbol{B}^1_{31} & \boldsymbol{B}^1_{32} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{B}^1_{41} & \boldsymbol{B}^2_{42} & \boldsymbol{0} & \boldsymbol{0} \end{bmatrix}. \tag{5}$$

Here, $n = 4T$, $k = 2T$, and the first $k = 2T$ nodes are punctured as information nodes among all the $n + k = 6T$ variable nodes. If we define a $T_L \times T_L$ circulant matrix as a micro-block, each block-entry $(\boldsymbol{h}_i)_{i_1,j_1}$ in $\boldsymbol{H}_i$ is a micro-block for $i_1 \in [\![1, \frac{n}{T_L}]\!]$ and $j_1 \in [\![1, \frac{n+k}{T_L}]\!]$, which is expanded from an element in $\boldsymbol{H}_{\mathrm{B},L-1}$. There are totally $J = |E|T/T_L$ non-zero elements in $\boldsymbol{H}_{\mathrm{B},L-1}$, each of which will be then replaced by a $T_L \times T_L$ CPM $\boldsymbol{I}_{T_L}(t)$. Meanwhile, all the zero elements in $\boldsymbol{H}_{\mathrm{B},L-1}$ will be replaced by a $T_L \times T_L$ zero matrix $\boldsymbol{0}_{T_L \times T_L}$. Then we have

$$(\boldsymbol{h}_i)_{i_1,j_1} = \begin{cases} \boldsymbol{I}_{T_L}(r_{i,j(i_1,j_1)}), & \text{if } (\boldsymbol{H}_{\mathrm{B},L-1})_{i_1,j_1} = 1; \\ \boldsymbol{0}_{T_L \times T_L}, & \text{if } (\boldsymbol{H}_{\mathrm{B},L-1})_{i_1,j_1} = 0, \end{cases} \tag{6}$$

where $j(i_1, j_1) \in [\![0, J-1]\!]$ indicates the index of each non-zero element $(\boldsymbol{H}_{\mathrm{B},L-1})_{i_1,j_1}$ in a certain order, and $r_{i,j(i_1,j_1)}$ is the corresponding shift value in $\boldsymbol{r}_i$. Now, we rewrite the binary keystream vector $\boldsymbol{r}_i$ as

$$\boldsymbol{r}_i = (r_{i,0}, r_{i,1}, \ldots, r_{i,j}, \ldots, r_{i,J-1}), \tag{7}$$

where each number $r_{i,j} \in [\![0, T_L - 1]\!]$ is an unsigned integer represented by $\log_2 T_L$ bits in the keystream $\boldsymbol{r}_i$. Therefore, all the random shift values in the L-th stage will be controlled by the keystream vector $\boldsymbol{r}_i$, whose length is required to be $h = J \log_2 T_L$ bits. Except for the random expanding stage, parameters selected in the first $L-1$ stages, i.e. all the expanding factors and shift values, are constant during the whole transmission session and will be publicly sent to Bob in the initialization vector $\boldsymbol{iv}$ before the session begins.

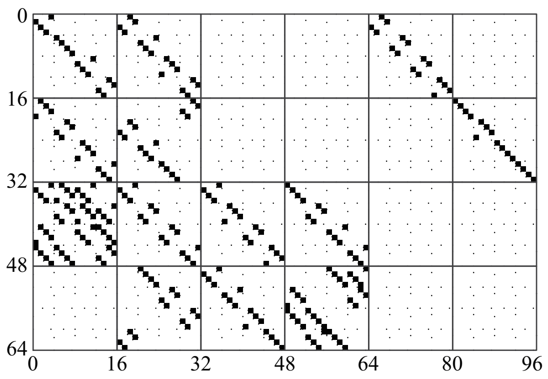

Figure 7 The equivalent parity-check matrix $\boldsymbol{H}_{\mathrm{B},2}$ after structured expanding, which will be then randomly expanded. Note that each non-zero element is black and equals 1.

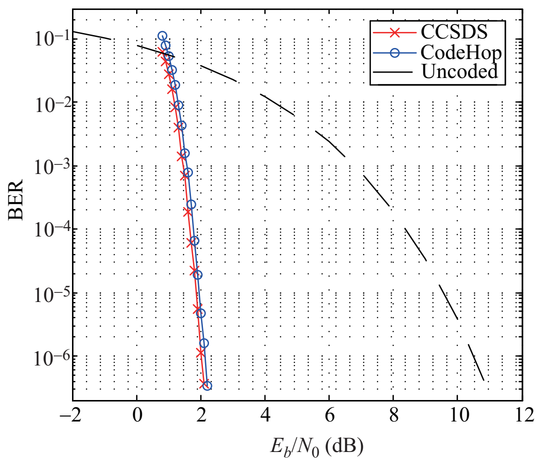

Figure 8 (Color online) For $(2048, 1024)$ LDPC codes, the average BER of the hopping nonsystematic LDPC codes in CodeHop is compared with the systematic code specified in CCSDS.

What is more, we also denote a $\frac{T}{T_L} \times \frac{T}{T_L}$ block matrix with micro-block entries as a macro-block. Each $T \times T$ submatrix $\boldsymbol{A}^w_{\alpha\beta}$ or $\boldsymbol{B}^w_{\alpha\beta}$ in (4) and (5) is a macro-block, which is expanded from a non-zero element in the base parity-check matrix $\boldsymbol{H}_{\mathrm{B},0}$, and the variable w at the top-right corner indicates that there are w permutation matrices $\boldsymbol{I}_{T_L}(t)$ in each of its micro-block row or column. Therefore, it can be known that except for $\boldsymbol{A}^3_{31}$, $\boldsymbol{B}^2_{42}$ and zero matrices $\boldsymbol{0}$, all the other macro-blocks in $\boldsymbol{A}(\boldsymbol{r}_i)$ and $\boldsymbol{B}(\boldsymbol{r}_i)$ are permutation matrices because their $w = 1$.

3.3 An example

With the structured-random expanding technique, we generate a hopping sequence of parity-check matrices for $(2048, 1024)$ nonsystematic LDPC codes as an example. From the protograph P in Figure 6, each parity-check matrix $\boldsymbol{H}_i$ in the hopping sequence will be constructed by a total expanding factor $T = T_1T_2T_3 = 4 \times 4 \times 32 = 512$ via $L = 3$ stages.

In the first two stages, the protograph P is expanded twice by factors of $T_1 = T_2 = 4$, which is intended to separate all the parallel edges and guarantee the girth is larger than 4 in the derived Tanner graph. The equivalent parity-check matrix $\boldsymbol{H}_{\mathrm{B},2}$ is shown in Figure 7. In the third stage, totally there are $|E|T/T_3 = 256$ edges to be randomly expanded by a factor of $T_3 = 32$, which is described by the keystream vector $\boldsymbol{r}_i$ with a length of $h = J \log_2 T_L = 1280$ bits. Thus there will be a large set of parity-check matrices $\mathcal{H} = \{\boldsymbol{H}(\boldsymbol{r}) : \boldsymbol{r} \in [\![0, 2^{1280}-1]\!]\}$ available to be randomly chosen in the hopping sequence of CodeHop. The period of the hopping sequence $\boldsymbol{H}_i = \boldsymbol{H}(\boldsymbol{r}_i) \in \mathcal{H}$ is decided by the period of the keystream $\boldsymbol{r}_i$ produced by the PRNG. E.g. for the A5/1 cipher with a 64-bit key, the period of the keystream will be nearly 2^{64}, which means the period of the hopping sequence $\boldsymbol{H}_i$ is at least $2^{64}/1280 \approx 2^{53.6}$.

Now a hopping sequence of LDPC codes has been generated, with each code randomly selected from the set $\mathcal{H}$ by the PRNG. But, to evaluate the error correcting performance of CodeHop, it is different from what is usually done with a fixed LDPC parity-check matrix. The BER of one hopping sequence will be evaluated as how CodeHop works, that is to say, by performing numerical simulations with switching parity-check matrices. Then, random initial states of the stream cipher will be tested to generate different hopping sequences, whose BERs are then collected to compute the average BER performance of CodeHop. As shown in Figure 8, the average BER performance of the structured-random nonsystematic $(2048, 1024)$ LDPC codes is plotted, which is compared with the LDPC code standardized by CCSDS. Note that the LDPC code in CCSDS is systematic and fixed, which is expected to be much better than a sequence of nonsystematic codes in CodeHop. The iterative decoding will be repeated until the parity checks are all satisfied or the maximum number of iterations 63 is reached in the simulation. It can be seen that it is slightly degraded by no more than 0.2 dB in E_b/N_0 for the average BER of nonsystematic LDPC codes, which means reliable transmission can be guaranteed by CodeHop.

4 Efficient encoder and decoder design

4.1 Encoder design

To implement LDPC encoders for CodeHop, each parity-check matrix $\boldsymbol{H}_i$ needs to be converted to a $k \times n$ nonsystematic generator matrix $\boldsymbol{G}_i$ for each plaintext message $\boldsymbol{m}_i$, which is derived by

$$\boldsymbol{G}_i = \left(\boldsymbol{B}(\boldsymbol{r}_i)^{-1} \cdot \boldsymbol{A}(\boldsymbol{r}_i)\right)^{\mathrm{T}} = ((\boldsymbol{g}_i)_{i_2,j_2})_{\frac{k}{T_L} \times \frac{n}{T_L}}, \tag{8}$$

where $\boldsymbol{G}_i$ is a T_L-circulant matrix and can be written as a $\frac{k}{T_L} \times \frac{n}{T_L}$ block-circulant matrix consisting of micro-blocks $(\boldsymbol{g}_i)_{i_2,j_2}$ for $i_2 \in [\![1, \frac{k}{T_L}]\!]$ and $j_2 \in [\![1, \frac{n}{T_L}]\!]$. According to (5), the submatrix $\boldsymbol{B}(\boldsymbol{r}_i)$ is full rank, which indicates that its inverse $\boldsymbol{B}(r_i)^{-1}$ is also T_L-circulant [26]. Since $\boldsymbol{A}(r_i)$ and $\boldsymbol{B}(r_i)^{-1}$ are both T_L-circulant, $\boldsymbol{G}_i$ is T_L-circulant.

On the other hand, $\boldsymbol{G}_i$ can be partitioned into eight macro-blocks $(\boldsymbol{G}_i)_{a,b}$ as follows,

$$\begin{aligned}\boldsymbol{G}_i &= \left(\boldsymbol{B}(\boldsymbol{r}_i)^{-1} \cdot \boldsymbol{A}(\boldsymbol{r}_i)\right)^{\mathrm{T}} = ((\boldsymbol{G}_i)_{a,b})_{2\times 4} \\ &= \begin{bmatrix} (\boldsymbol{A}_{31}^3)^{\mathrm{T}}\boldsymbol{D}_1 & (\boldsymbol{A}_{31}^3)^{\mathrm{T}}\boldsymbol{D}_3 & \boldsymbol{A}_{11}^{\mathrm{T}}\boldsymbol{B}_{13} & \boldsymbol{A}_{21}^{\mathrm{T}}\boldsymbol{B}_{24} \\ \boldsymbol{A}_{32}^{\mathrm{T}}\boldsymbol{D}_1 \oplus \boldsymbol{A}_{42}^{\mathrm{T}}\boldsymbol{D}_2 & \boldsymbol{A}_{32}^{\mathrm{T}}\boldsymbol{D}_3 \oplus \boldsymbol{A}_{42}^{\mathrm{T}}\boldsymbol{C}^{\mathrm{T}} & \boldsymbol{A}_{12}^{\mathrm{T}}\boldsymbol{B}_{13} & \boldsymbol{A}_{22}^{\mathrm{T}}\boldsymbol{B}_{24} \end{bmatrix},\end{aligned} \tag{9}$$

where we define the following macro-blocks by

$$\boldsymbol{D}_1 = \left(\boldsymbol{I} \oplus \boldsymbol{B}_{31}\boldsymbol{B}_{41}^{\mathrm{T}}\boldsymbol{C}^{\mathrm{T}}\boldsymbol{B}_{32}^{\mathrm{T}}\right)\boldsymbol{B}_{31}, \tag{10}$$

$$\boldsymbol{D}_2 = \boldsymbol{C}^{\mathrm{T}}\boldsymbol{B}_{32}^{\mathrm{T}}\boldsymbol{B}_{31}, \tag{11}$$

$$\boldsymbol{D}_3 = \boldsymbol{B}_{31}\boldsymbol{B}_{41}^{\mathrm{T}}\boldsymbol{C}^{\mathrm{T}}, \tag{12}$$

$$\boldsymbol{C} = \left((\boldsymbol{B}_{42}^2)^{\mathrm{T}} \oplus \boldsymbol{B}_{32}^{\mathrm{T}}\boldsymbol{B}_{31}\boldsymbol{B}_{41}^{\mathrm{T}}\right)^{-1}. \tag{13}$$

Note that the transpose of a matrix is denoted by $(\cdot)^{\mathrm{T}}$, and the value of w for macro-blocks $\boldsymbol{A}_{\alpha\beta}^w$ and $\boldsymbol{B}_{\alpha\beta}^w$ is neglected when $w = 1$.

Compared with efficient encoders for QC-LDPC codes with a fixed parity-check matrix such as CCSDS [26,27], it can be seen from (9) to (13) that the additional computational complexity for CodeHop will be the inversion operation to derive the macro-block $\boldsymbol{C}$ in (13). Nevertheless, Bogdanov et al. [28] showed that a parallel hardware architecture can be implemented to finish the inversion of a binary matrix via Gaussian elimination in $O(n)$ time. What is more, inherited from the quasi-cyclic structure of hopping matrices $\boldsymbol{H}_i$, Gaussian elimination for the macro-block $\boldsymbol{C}$ can be executed on micro-blocks, which further reduces the computational complexity by a factor of T_L.

Then, with the macro-block $\boldsymbol{C}$, there is no need to explicitly compute and store macro-blocks $\boldsymbol{D}_1$ to $\boldsymbol{D}_3$. Alternatively, we can multiply information bits with the macro-block matrices which derive $\boldsymbol{D}_1$ to $\boldsymbol{D}_3$ in several steps as in [29]. For example, to multiply by $\boldsymbol{D}_3$ can be divided into three steps by successively multiplying by $\boldsymbol{B}_{31}$, $\boldsymbol{B}_{41}^{\mathrm{T}}$ and $\boldsymbol{C}^{\mathrm{T}}$. Hence, similar to the encoders for QC-LDPC codes, since the macro-blocks are all circulant, all of the required multiplication operations can be done in $O(n)$ time.

According to (13), the size of the macro-block $\boldsymbol{C}$ is about 1/8 of the size of the generator matrix $\boldsymbol{G}_i$. Therefore, compared with LDPC encoders for a fixed QC-LDPC code, there will be a slight increase in storage of about 1/8 for CodeHop because of the matrix inversion module for $\boldsymbol{C}$. Nevertheless, owing to the block-circulant property of $\boldsymbol{H}_i$, the encoding processing still can be completed in $O(n)$ time.

4.2 Decoder design

As for an efficient decoder of CodeHop, it can be extended from the conventional decoder of QC-LDPC codes [30], since the structured-random LDPC codes constructed in Section 3 are also quasi-cyclic. The only difference is that, for conventional QC-LDPC decoders, the shift values of CPMs in the parity-check matrix are fixed, which, however, become flexible now and will be updated for each hopping matrix $\boldsymbol{H}_i$. When the shift values are successfully updated for $\boldsymbol{H}_i$, the iterative decoding process is exactly the same.

According to (6) and (7), the shift values $r_{i,j(i_1,j_1)}$ of CPMs, i.e. non-zero micro-blocks $(\boldsymbol{h}_i)_{i_1,j_1}$, will be determined by the binary keystream $\boldsymbol{r}_i$ of the PRNG. It is worth noting that each hopping matrix $\boldsymbol{H}_i$ is very sparse, e.g. there are only 256 CPMs and 1280 bits of a keystream for each $\boldsymbol{H}_i$ in the example mentioned in Subsection 3.3. Meanwhile, the LFSR-based stream cipher to generate binary keystreams is easy to be implemented on shift registers. Furthermore, the shift values of $\boldsymbol{H}_{i+1}$ can be computed in advance when the i-th ciphertext $\boldsymbol{y}_i$ is being decoded by $\boldsymbol{H}_i$. Therefore, the updating time for the shift values can be neglected and a negligible increase of storage for the next binary keystream $\boldsymbol{r}_{i+1}$ is needed.

5 Cryptanalysis

In this section, the security of CodeHop will be analyzed under chosen plaintext attacks. Firstly, it is shown that how a fixed secret parity-check matrix $\boldsymbol{H}_0$ can be recovered by Eve when hopping is not used. Then, it will be proved that chosen-plaintext attacks cannot break CodeHop when the secret parity-check matrix $\boldsymbol{H}_i$ rapidly hops according to a pseudo-random sequence produced by a PRNG, such as the A5/1 stream cipher. Furthermore, the security of the scheme will be enhanced in the case of noisy channels.

5.1 Security of a non-hopping parity-check matrix

To evaluate the security of a fixed and single parity-check matrix, we assume that the secret parity-check matrix keeps constant $\boldsymbol{H}_0 = [\boldsymbol{A}, \boldsymbol{B}]$ without hopping in the whole session. It can be seen that once Eve accumulates enough number of plaintext-ciphertext pairs, $\boldsymbol{H}_0$ will be recovered. Thus, the security of $\boldsymbol{H}_0$ is defined as how many plaintext-ciphertext pairs are required to recover it. Note that Eve needs to mount the attack by analyzing the structure of $\boldsymbol{G}_0$, since the $n \times (n+k)$ parity-check matrix $\boldsymbol{H}_0$ is converted to a $k \times n$ nonsystematic generator matrix $\boldsymbol{G}_0 = (\boldsymbol{B}^{-1}\boldsymbol{A})^{\mathrm{T}}$ during LDPC encoding.

From (8), it can be known that $\boldsymbol{G}_0$ can be referred to as a $\frac{k}{T_L} \times \frac{n}{T_L}$ block-circulant matrix, where each block-entry $(\boldsymbol{g}_0)_{i_2,j_2}$ is a $T_L \times T_L$ circulant micro-block. Note that each $(\boldsymbol{g}_0)_{i_2,j_2}$ can be represented by the first row of it, thus each micro-block row of $\boldsymbol{G}_0$ can be also recovered by its first row, which needs only one plaintext-ciphertext pair. So, $\frac{k}{T_L}$ plaintext-ciphertext pairs are required to learn all the micro-block rows of $\boldsymbol{G}_0$, which finishes the recovering of the generator matrix.

Let us take the $(2048, 1024)$ nonsystematic LDPC code constructed in Section 3.3 as an example. Here, $\boldsymbol{G}_0$ is a 32-circulant matrix with $\frac{k}{T_L} = 32$ micro-block rows and $\frac{n}{T_L} = 64$ micro-block columns, where each micro-block is a 32×32 circulant matrix. Then, 32 plaintext-ciphertext pairs are enough to reconstruct the whole $\boldsymbol{G}_0$. For $i_2 \in [\![1, 32]\!]$, the i_2-th micro-block row of $\boldsymbol{G}_0$ can be recovered by the ciphertext $\boldsymbol{z}_{i_2} = \boldsymbol{m}_{i_2} \cdot \boldsymbol{G}_0$ by choosing plaintexts to be $\boldsymbol{m}_{i_2} = \boldsymbol{u}_{1024}(32 \cdot (i_2 - 1))$.

When transmitting over noisy channels, the ciphertext $\widetilde{\boldsymbol{z}}_{i_2}$ received by Eve will be erroneous as

$$\widetilde{\boldsymbol{z}}_{i_2} = \boldsymbol{z}_{i_2} \oplus \boldsymbol{e}_{i_2} = \boldsymbol{m}_{i_2} \cdot \boldsymbol{G}_0 \oplus \boldsymbol{e}_{i_2}, \tag{14}$$

where $\boldsymbol{e}_{i_2}$ is the channel error vector. In this case, the erroneous ciphertext $\widetilde{\boldsymbol{z}}_{i_2}$ cannot be directly exploited to recover the generator matrix $\boldsymbol{G}_0$. Eve needs to detect the error vector $\boldsymbol{e}_{i_2}$ and remove it to obtain the correct ciphertext $\boldsymbol{z}_{i_2}$. An effective way for Eve can be a repetition coding strategy as follows. Repeating the same plaintext $\boldsymbol{m}_{i_2}$ for N times, the ciphertext $\boldsymbol{z}_{i_2}$ will be transmitted over the channel for N times and the required number of plaintext-ciphertext pairs will be $32N$. Then, for each bit in the original ciphertext $\boldsymbol{z}_{i_2}$, it forms a rate$-1/N$ repetition code, where maximum likelihood (ML) decoding can be used to find the most likely transmitted bit value. Hence, the BER of decoded ciphertext bits can be significantly decreased. From Figure 9, it can be seen that as the number of plaintext-ciphertext pairs grows, the BER of decoded ciphertext bits after repetition coding will be decreased to a very low level, which then will be used to successfully recover the generator matrix $\boldsymbol{G}_0$. Therefore, joint channel coding and encryption with a fixed parity-check $\boldsymbol{H}_0$ can be easily broken by chosen-plaintext attacks.

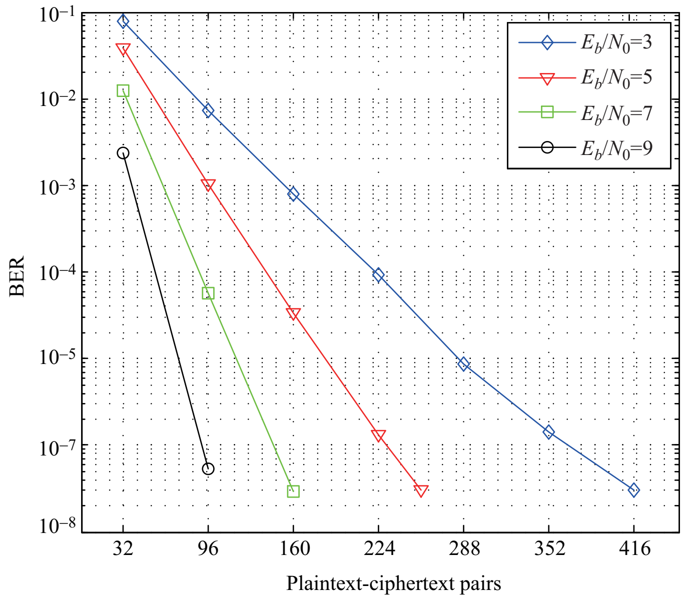

Figure 9 (Color online) For a non-hopping parity-check matrix, given E_b/N_0 of the received ciphertext $\widetilde{\boldsymbol{z}}_{i_2}$, the BER curves of decoded ciphertext bits after repetition coding are illustrated with regarding to the number of plaintext-ciphertext pairs.

5.2 Security of hopping parity-check matrices

In CodeHop, the secret parity-check matrix $\boldsymbol{H}_i$ fast switches in a hopping sequence, such that each $\boldsymbol{H}_i$ is used only for one plaintext message $\boldsymbol{m}_i$ to be encoded and encrypted. As mentioned in Section 2, we take A5/1 stream cipher as the PRNG. A5/1 is an LFSR-based stream cipher which has been used in GSM for air-interface encryption. LFSR-based stream ciphers are generally designed for high speed secure data transmission, which can be easily implemented in hardware with very low complexity and power assumption. However, it was reported by Barkan et al. [3] in 2006 that the session key $\boldsymbol{k}$ of A5/1 can be recovered by correlation attack in 133 s in average, with 2000 frames' known keystream, where each frame consists of 224 bits. Such amount of keystream can be captured using just 9.2 s' transmissions of known plaintext-ciphertext pairs, which indicates that A5/1 cipher is not immune against known-plaintext attacks. To improve the security of wireless communication without increasing complexity, CodeHop will perform joint channel coding and encryption, combining LDPC codes with stream ciphers, such as the already broken A5/1 as the PRNG.

To attack CodeHop, Eve will try to find out the hopping sequence of parity-check matrices $\boldsymbol{H}_i$, which depends on the keystream $\boldsymbol{r}_i$ produced by the PRNG via structured-random expanding. As discussed above, each $\boldsymbol{H}_i$ is converted to $\boldsymbol{G}_i$ during LDPC encoding, and $\boldsymbol{G}_i$ cannot be inferred from just one plaintext-ciphertext pair $(\boldsymbol{m}_i, \boldsymbol{z}_i)$. Thus Eve can only obtain some relationships between keystream bits in $\boldsymbol{r}_i$. We will prove that CodeHop can resist known or chosen plaintext attacks when transmitting over noiseless channels, and the security of CodeHop will be enhanced over noisy channels.

5.2.1 *Noiseless channel*

Firstly we assume that the ciphertext received by Eve is error-free, i.e. $\boldsymbol{z}_i = \boldsymbol{m}_i \cdot \boldsymbol{G}_i$. For a circulant matrix $\boldsymbol{h}$ of size $T_L \times T_L$, it can be associated with $h(x) \in \mathbb{F}_2[x]$, i.e. a univariate polynomial with coefficients in $\mathbb{F}_2$. The mapping of $\boldsymbol{h}$ to $h(x)$ [31] will take the values in the first row $(h_{1,1}, h_{1,2}, \ldots, h_{1,T_L})$ of $\boldsymbol{h}$ as $h(x) = h_{1,1} + h_{1,2}x + \cdots + h_{1,T_L}x^{T_L-1}$. Then, each micro-block $(\boldsymbol{h}_i)_{i_1,j_1}$ in $\boldsymbol{H}_i$ can be denoted by $(h_i)_{i_1,j_1}(x) \in \mathbb{F}_2[x]$, and $\boldsymbol{H}_i$ can be represented by a matrix of polynomials

$$H_i(x) = ((h_i)_{i_1,j_1}(x))_{\frac{n}{T_L} \times \frac{n+k}{T_L}}. \tag{15}$$

Since the micro-blocks in $\boldsymbol{H}_i$ are all circulant matrices expanded from elements in $\boldsymbol{H}_{\mathrm{B},L-1}$, according to (6), we have

$$(h_i)_{i_1,j_1}(x) = \begin{cases} x^{r_{i,j(i_1,j_1)}}, & \text{if } (\boldsymbol{H}_{\mathrm{B},L-1})_{i_1,j_1} = 1; \\ 0, & \text{if } (\boldsymbol{H}_{\mathrm{B},L-1})_{i_1,j_1} = 0. \end{cases} \tag{16}$$

Similarly, the generator matrix $\boldsymbol{G}_i$ can be represented as follows,

$$G_i(x) = ((g_i)_{i_2,j_2}(x))_{\frac{k}{T_L}\times\frac{n}{T_L}}, \tag{17}$$

where each $(g_i)_{i_2,j_2}(x)$ represents a micro-block $(\boldsymbol{g}_i)_{i_2,j_2}$. Note that from (9), micro-blocks $(\boldsymbol{g}_i)_{i_2,j_2}$ in the left four dense macro-blocks, i.e. $(\boldsymbol{G}_i)_{a,b}$ for $b = 1$ or 2, are too complicated to describe. However, it is much easier to analyze the right four macro-blocks, i.e. $(\boldsymbol{G}_i)_{a,b}$ for $b = 3$ or 4, since each of them is sparse with a unique micro-block of CPM $\boldsymbol{I}_{T_L}(t_{i,(a,j_2)})$ in each micro-block row or column. The variable a denotes the row index of the macro-block which $\boldsymbol{I}_{T_L}(t_{i,(a,j_2)})$ is located in and $j_2 \in [\![\frac{n}{2T_L}+1, \frac{n}{T_L}]\!]$ denotes the micro-block column index of $\boldsymbol{I}_{T_L}(t_{i,(a,j_2)})$ in $\boldsymbol{G}_i$. With a and j_2, the micro-block row index of $\boldsymbol{I}_{T_L}(t_{i,(a,j_2)})$ can be determined by $i_2(a,j_2)$. Each $\boldsymbol{I}_{T_L}(t_{i,(a,j_2)})$ is the product of two micro-blocks in $\boldsymbol{A}_{\alpha\beta}$ and $\boldsymbol{B}_{\alpha\beta}$. Letting $j_{\mathrm{A}}(a,j_2)$ and $j_{\mathrm{B}}(j_2)$ denote the indices of shift values for the two micro-blocks in the keystream $\boldsymbol{r}_i$, we have

$$(g_i)_{i_2(a,j_2),j_2}(x) = x^{t_{i,(a,j_2)}} = x^{r_{i,j_{\mathrm{B}}(j_2)} - r_{i,j_{\mathrm{A}}(a,j_2)}} \bmod (x^{T_L}-1), \tag{18}$$

from which we can derive

$$t_{i,(a,j_2)} = \left(r_{i,j_{\mathrm{B}}(j_2)} - r_{i,j_{\mathrm{A}}(a,j_2)}\right) \bmod T_L. \tag{19}$$

For more details, the proof of (18) is presented in Appendix A.

However, CodeHop allows Eve to try just one pair of $(\boldsymbol{m}_i, \boldsymbol{z}_i)$ for each $\boldsymbol{G}_i$, which means it is not possible to reconstruct the whole $\boldsymbol{G}_i$. Nevertheless, Eve can still learn some equations over bits of the keystream $\boldsymbol{r}_i$. If the plaintext is chosen to be

$$\boldsymbol{m}_i = (\underbrace{\boldsymbol{u}_{32}(0),\ldots,\boldsymbol{u}_{32}(0)}_{16},\underbrace{0,\ldots,0}_{16}). \tag{20}$$

Eve can receive the error-free ciphertext

$$\boldsymbol{z}_i = \left(\boldsymbol{v}_{i,1},\ldots,\boldsymbol{v}_{i,j_2},\ldots,\boldsymbol{v}_{i,32},\boldsymbol{u}_{32}(t_{i,(1,33)}),\ldots,\boldsymbol{u}_{32}(t_{i,(1,j_2)}),\ldots,\boldsymbol{u}_{32}(t_{i,(1,64)})\right), \tag{21}$$

where $\boldsymbol{v}_{i,j_2}$ is a 32-bit dense vector that accumulates the first rows of $(\boldsymbol{g}_i)_{i_2,j_2}$ for $1 \leqslant i_2 \leqslant 16$ by XOR operation and can be represented by the coefficients of the following polynomial

$$v_{i,j_2}(x) = \oplus\sum_{i_2=1}^{16}(g_i)_{i_2,j_2}(x), \tag{22}$$

and $\boldsymbol{u}_{32}(t_{i,(1,j_2)})$ is the first row of $(\boldsymbol{g}_i)_{i_2(1,j_2),j_2}$. Thus from (18), the equations over bits of $\boldsymbol{r}_i$ can be established as follows, for $33 \leqslant j_2 \leqslant 64$,

$$\left(r_{i,j_{\mathrm{B}}(j_2)} - r_{i,j_{\mathrm{A}}(1,j_2)}\right) \bmod 32 = t_{i,(1,j_2)}, \tag{23}$$

where each $t_{i,(1,j_2)}$ is obtained from $\boldsymbol{z}_i$. According to (7), each $r_{i,j} = (r_{i,j}[4]||\ldots||r_{i,j}[0])$ is described by 5 bits of the keystream $\boldsymbol{r}_i$, where $d[v]$ denotes the v-th bit position of an integer d from right to left with the least significant bit assigned with the position $v = 0$. Thus, Eq. (23) can be rewritten as

$$r_{i,j_{\mathrm{A}}(1,j_2)}[0] \oplus r_{i,j_{\mathrm{B}}(j_2)}[0] = t_{i,(1,j_2)}[0], \tag{24}$$

$$r_{i,j_{\mathrm{A}}(1,j_2)}[1] \oplus r_{i,j_{\mathrm{B}}(j_2)}[1] \oplus r_{i,j_{\mathrm{B}}(j_2)}[0] \oplus \left(r_{i,j_{\mathrm{A}}(1,j_2)}[0]\cdot r_{i,j_{\mathrm{B}}(j_2)}[0]\right) = t_{i,(1,j_2)}[1], \tag{25}$$

$\cdots\cdots\cdot$

Here, only equations over bit positions 0 and 1 are shown and the other equations over bits 2, 3 and 4 are omitted for simplicity. Recall that correlation attack [3] requires the XOR of two consecutive bits in the keystream $\boldsymbol{r}_i$, which is not available in CodeHop. Specifically, using chosen plaintext-ciphertext pairs, Eq. (24) gives the XOR of two keystream bits which are at least 5 bits apart, and Eq. (25) combines an additional nonlinear term of two keystream bits, not to mention the equations over bits $2, 3, 4$. It can be known that as the position v grows, the equation over the keystream bits is more and more complicated. Thus, such kind of correlation attack will not work and it is computationally infeasible to break the A5/1 stream cipher. As a result, CodeHop can withstand chosen-plaintext attacks in the case of noiseless channels.

5.2.2 *Noisy channel*

Practically, we consider transmitting over wireless channels, e.g. AWGN channels. With the synchronized parity-check matrix $\boldsymbol{H}_i$, Bob can decode the plaintext messages $\boldsymbol{m}_i$ correctly. But for Eve, LDPC decoding cannot be performed without the parity-check matrix $\boldsymbol{H}_i$. Thus, channel symbols will be hard decided, and received ciphertexts are corrupted by channel noise as

$$\widetilde{\boldsymbol{z}}_i = \boldsymbol{z}_i \oplus \boldsymbol{e}_i = \boldsymbol{m}_i \cdot \boldsymbol{G}_i \oplus \boldsymbol{e}_i, \tag{26}$$

where $\boldsymbol{e}_i$ is the channel error vector. Considering BPSK modulation, the BER for Eve after hard decision is given by $p_e = Q(\sqrt{2R \cdot E_b/N_0})$, where $Q(\cdot)$ is the Q-function and R is the LDPC coding rate. For a rate-1/2 LDPC code, the BER of codeword bits with hard decision will be degraded by 3 dB in E_b/N_0 even compared with the uncoded BPSK.

Due to the randomness of channel errors, it is difficult for Eve to tell whether a ciphertext bit is affected or not. However, some channel errors can still be detected. In $\widetilde{\boldsymbol{z}}_i$, if there is more than one non-zero bit in the sub-vector expected to be $\boldsymbol{u}_{32}(t_{i,(1,j_2)})$, it indicates such a sub-vector is erroneous. Thus, Eve can try to send another chosen-plaintext $\boldsymbol{m}_i$ to find an error-free ciphertext $\boldsymbol{z}_i$, which requires more plaintext-ciphertexts pairs. On the other hand, Eve can try to search all the error patterns $\boldsymbol{e}_i$ exhaustively, which will highly increase the computational complexity of the attack. Therefore, the security of CodeHop will be improved with the help of channel errors, in terms of increased number of plaintext-ciphertext pairs or computational complexity.

6 Conclusion

We have presented a CodeHop scheme for physical layer error correction and security. It employs non-systematic LDPC codes to combine channel coding and data encryption in a single step. In particular, a hopping sequence of parity-check matrices $\boldsymbol{H}_i$ is secretly generated via structured-random protograph expanding, which will be synchronized between Alice and Bob according to the pseudo random sequence produced by a pair of PRNG. Each plaintext message is jointly encoded and encrypted by a unique $\boldsymbol{H}_i$ to get the codeword as the ciphertext, which is then jointly decoded and decrypted by the same $\boldsymbol{H}_i$ at the legitimate receiver. The proposed CodeHop may reinforce the security of some traditional communication systems, e.g. GSM. Taking the A5/1 stream cipher of GSM as the PRNG, the CodeHop has been cryptanalyzed under chosen-plaintext attacks to recover the hopping sequence, showing the resistance to the best chosen-plaintext attack effective on A5/1. Moreover, the security of the CodeHop may be enhanced in the case of noisy channels, preserving error correction performance.

Acknowledgements This work was supported by National Basic Research Program of China (Grant No. 2013CB329001) and National Natural Science Foundation of China (Grants Nos. 61132002, 61101072).

Conflict of interest The authors declare that they have no conflict of interest.

References

1 Liang Y, Poor H V, Shamai S. Information theoretic security. Found Trends Commun Inf Theory, 2009, 5: 355–580

2 Wang B, Mu P C, Yang P Z, et al. Two-step transmission with artificial noise for secure wireless SIMO communications. Sci China Inf Sci, 2015, 58: 042308

3 Barkan E, Biham E. Conditional estimators: an effective attack on A5/1. In: Selected Areas in Cryptography. Berlin: Springer, 2006, 3897: 1–19

4 Wyner A D. The wire-tap channel. Bell Syst Tech J, 1975, 54: 1355–1387

5 Thangaraj A, Dihidar S, Calderbank A R, et al. Applications of LDPC codes to the wiretap channel. IEEE Trans Inf Theory, 2007, 53: 2933–2945

6 Wen H, Gong G, Ho P H. Build-in wiretap channel I with feedback and LDPC codes. J Commun Netw, 2009, 11: 538–543

7 Klinc D, Ha J, McLaughlin S W, et al. LDPC codes for physical layer security. In: Proceedings of the 28th IEEE Conference on Global Telecommunications. New York: IEEE Press, 2009. 5765–5770

8 Zúquete A, Barros J. Physical-layer encryption with stream ciphers. In: IEEE International Symposium on Information Theory, Toronto, 2008. 106–110

9 Barkan E, Biham E, Keller N. Instant ciphertext-only cryptanalysis of gsm encrypted communication. J Cryptol, 2008, 21: 392–429

10 Harrison W K, Almeida J, McLaughlin S W, et al. Coding for cryptographic security enhancement using stopping sets. IEEE Trans Inf Foren Secur, 2011, 6: 575–584

11 Schneier B. Applied Cryptography Protocols, Algorithms, and Source Code in C. 2nd ed. New York: John Wiley & Sons, 1996

12 Wei S, Wang J, Yin R, et al. Trade-off between security and performance in block ciphered systems with erroneous ciphertexts. IEEE Trans Inf Foren Secur, 2013, 8: 636–645

13 Mathur C N, Narayan K, Subbalakshmi K. On the design of error-correcting ciphers. EURASIP J Wirel Commun Netw, 2006, 2006: 72

14 National Institute of Standards and Technology, U.S. Department of Commerce. Advanced Encryption Standard (AES). FIPS PUB 197. http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf. 2001

15 Adamo O, Fu S, Varanasi M R. Physical layer error correction based cipher. In: IEEE Global Telecommunications Conference, Miami, 2010. 1–5

16 Chai Q, Gong G. Differential cryptanalysis of two joint encryption and error correction schemes. In: IEEE Global Telecommunications Conference, Houston, 2011. 1–6

17 Anzalchi J, Couchman A, Gabellini P, et al. Beam hopping in multi-beam broadband satellite systems: system simulation and performance comparison with non-hopped systems. In: the 5th Advanced Satellite Multimedia Systems Conference (asma) and the 11th Signal Processing for Space Communications Workshop (spsc), Cagliari, 2010. 248–255

18 Wen H, Gong G, Lv S C, et al. Framework for MIMO cross-layer secure communication based on STBC. Telecommun Syst, 2013, 52: 2177–2185

19 Thorpe J. Low-density parity-check (LDPC) codes constructed from protographs. IPN Progress Report, 2003, 42: 42–154

20 Abbasfar A, Divsalar D, Yao K. Accumulate-repeat-accumulate codes. IEEE Trans Commun, 2007, 55: 692–702

21 Consultative Committee for Space Data Systems. Recommendation for Space Data Systems Standards: TM Synchronization and Channel Coding. CCSDS 131.0-B-2 Blue Book. http://public.ccsds.org/publications/archive/131x0b2ec1.pdf. 2011

22 Brink S, Kramer G. Design of repeat-accumulate codes for iterative detection and decoding. IEEE Trans Signal Process, 2003, 51: 2764–2772

23 Zhang L J, Li B, Cheng L L. Constructions of QC LDPC codes based on integer sequences. Sci China Inf Sci, 2014, 57: 062304

24 Liu X, Xiong F, Zhou Z, et al. Construction of quasi-cyclic LDPC cycle codes over Galois field GF(q) based on cycle entropy and applications on patterned media storage. IEEE Trans Magnetics, 2015, 51: 7209305

25 Andrews K, Dolinar S, Divsalar D, et al. Design of low-density parity-check (LDPC) codes for deep-space applications. IPN Progress Report, 2004, 42: 42–159

26 Andrews K, Dolinar S, Thorpe J. Encoders for block-circulant ldpc codes. In: International Symposium on Information Theory, Adelaide, 2005. 2300–2304

27 Li Z, Chen L, Zeng L, et al. Efficient encoding of quasi-cyclic low-density parity-check codes. IEEE Trans Commun, 2006, 54: 71–81

28 Bogdanov A, Mertens M, Paar C, et al. SMITH-A parallel hardware architecture for fast Gaussian elimination over GF(2). In: Workshop on Special-purpose Hardware for Attacking Cryptographic Systems (SHARCS), Cologne, 2006

29 Richardson T J, Urbanke R L. Efficient encoding of low-density parity-check codes. IEEE Trans Inf Theory, 2001, 47: 638–656

30 Ueng Y L, Yang B J, Yang C J, et al. An efficient multi-standard LDPC decoder design using hardware-friendly shuffled decoding. IEEE Trans Circuits Syst I Reg Papers, 2013, 60: 743–756

31 Otmani A, Tillich J P, Dallot L. Cryptanalysis of two mceliece cryptosystems based on quasi-cyclic codes. Math Comput Sci, 2010, 3: 129–140

Appendix A Proof of (18)

Take the submatrix $\boldsymbol{A}_{11}^{\mathrm{T}}\boldsymbol{B}_{13}$ as an example. Since $\boldsymbol{A}_{11}$ and $\boldsymbol{B}_{13}$ are both T_L-circulant permutation matrices, $\boldsymbol{A}_{11}^{\mathrm{T}}\boldsymbol{B}_{13}$ is also a $\frac{T}{T_L} \times \frac{T}{T_L}$ block matrix with one CPM $\boldsymbol{I}_{T_L}(t_{i,(a,j_2)})$ in each micro-block row or micro-block column, where $\boldsymbol{I}_{T_L}(t_{i,(a,j_2)})$ is the product of two circulant permutation micro-blocks in $\boldsymbol{A}_{11}$ and $\boldsymbol{B}_{13}$, respectively. It is easy to know that $a = 1$ and $j_2 \in [\frac{n}{2T_L} + 1, \frac{3n}{4T_L}]$ for the CPM $\boldsymbol{I}_{T_L}(t_{i,(a,j_2)})$ in each micro-block column of $\boldsymbol{A}_{11}^{\mathrm{T}}\boldsymbol{B}_{13}$. So we have

$$\boldsymbol{I}_{T_L}(t_{i,(1,j_2)}) = \boldsymbol{I}_{T_L}^{\mathrm{T}}(r_{i,j_{\mathrm{A}}(1,j_2)}) \cdot \boldsymbol{I}_{T_L}(r_{i,j_{\mathrm{B}}(j_2)}), \tag{A1}$$

where $j_{\mathrm{A}}(1, j_2)$ and $j_{\mathrm{B}}(j_2)$ are the indices of the shift values in $\boldsymbol{r}_i$ for the two circulant permutation micro-blocks in $\boldsymbol{A}_{11}$ and $\boldsymbol{B}_{13}$, respectively. Given the initialization vector $\boldsymbol{iv}$, such indices can be easily determined. Since the transpose of $\boldsymbol{I}_{T_L}(t)$ is denoted by $x^{T_L}(\frac{1}{x})^t$, then from (A1), we can derive

$$\begin{aligned} x^{t_{i,(1,j_2)}} &= \left(x^{T_L}(1/x)^{r_{i,j_{\mathrm{A}}(1,j_2)}}\right) \cdot x^{r_{i,j_{\mathrm{B}}(j_2)}} \bmod (x^{T_L} - 1) \\ &= x^{r_{i,j_{\mathrm{B}}(j_2)} - r_{i,j_{\mathrm{A}}(1,j_2)}} \bmod (x^{T_L} - 1), \end{aligned} \tag{A2}$$

which gives the result of (18).