Hybrid Encryption Scheme based on Polar Codes

Reza Hooshmand

Department of Electrical Engineering, Shahid Sattari Aeronautical University of Science and Technology, Tehran, Iran

Abstract

This paper introduces a secure and efficient hybrid scheme based on polar codes, called as HES-PC. The proposed HES-PC contains of two other mechanisms: (i) a key encapsulation mechanism based on polar codes, called as KEM-PC; (ii) a data encapsulation mechanism based on polar codes, called as DEM-PC. In fact, the symmetric key is exchanged between the legitimate partners by exploiting the KEM-PC. Also, secure polar encoding/successive cancelation (SC) decoding is enhanced between the honest parties by using DEM-PC. Moreover, by exploiting the characteristics of polar codes, the key sizes of KEM-PC and DEM-PC are reduced significantly. To decrease the key sizes of KEM-PC and DEM-PC, new approaches are proposed to store a little sub-channel indices in lieu of storing the generator matrix of applied polar code. Also, the security analyses demonstrate that the polar code-based HES has a proper security level opposed to the usual attacks. In reality, reducing the length of secret key in KEM-PC and DEM-PC has no influence on the level of security in the HES-PC. The aforementioned properties makes HES-PC to be usable in the secure physical layer communication devices and protocols.

Keywords: Hybrid Encryption Scheme, Polar Codes, Security

*Corresponding author

Email address: rhooshmand@ssau.ac.ir (Reza Hooshmand)

1. Introduction

A hybrid encryption scheme (HES) exploits public-key encryption methods to exchange a symmetric key that is then applied for encryption the actual plaintexts using the secret key method of encryption [1]. Hybrid encryption scheme incorporates a combination of two mechanisms, i.e. key encapsulation mechanism (KEM) and data encapsulation mechanism (DEM), to benefit from enough security and performance levels. Therefore, HES is noticed a highly secure encryption scheme as long as the KEM and DEM are fully secure. However, commonly-used HES will be vulnerable to attacks based on the large scale quantum computers. In fact, designing and planning the quantum-impregnable HES that exploits DEM and KEM, simultanously with suitable tradeoff between security and performance is an important labor [2].

Also, polar codes [3] are a category of linear codes that can attain the channel capacity by using efficient polar encoder and fast polar decoder. Most lately, polar codes are exploited in distinct cryptographic schemes such as in public key method of encryption [4], physical layer method of encryption [5], secure channel method of coding [6], secret key method of encryption [7], identification plan [8] and key encapsulation mechanism [9]. Hence, due to the capabilities of polar codes, it is argumentative to use polar codes in the hybrid encryption schemes. In this work, a secure polar code-based hybrid encryption scheme, named as HES-PC, is introduced in which enough levels of security and performance are enhanced by using the characteristics of polar codes. The security of HES-PC is based on the difficulty of general decoding problem (GDP) [10]. Therefore, HES-PC is categorized in the class of post-quantum cryptography that can oppose quantum computer-based attacks. An important aim in designing HES-PC is to acquire a proper exchange between security level and key size. This aim can be acquired by exploiting polar codes in the structure of HES-PC.

The proposed HES-PC contains of two mechanisms: (i) a key encapsulation mechanism based on polar codes, named as KEM-PC, in which symmetric key encapsulation/decapsulation are performed; (ii) a data encapsulation mechanism based on polar codes, named as DEM-PC, where data encapsulation/decapsulation are executed. In fact, in the proposed HES-PC, the secret key is exchanged between the legitimate partners by exploiting the KEM-PC. Also, secure polar encoder/successive cancelation (SC) decoder is enhanced between the honest parties by using DEM-PC. In fact, in the DEM-PC, the encryption and polar encoding algorithms are carried out concurrently to provide the security and reliability of the exchanged data. Moreover, merging the encryption/polar encoding and decryption/polar decoding algorithms in the DEM-PC can result faster and easier execution. Moreover, by exploiting the characteristics of polar codes, the key sizes of KEM-PC and DEM-PC are reduced significantly in comparison to the previous schemes. To reduce the key sizes of KEM-PC and DEM-PC, new approaches are proposed by which a few sub-channel indices are saved in lieu of the generator matrix of used polar codes.

The remainder of this paper is organized as follows. The structure of polar codes and polar encoder/decoder are explained in Section 2. We describe the construction of introduced HES-PC, including the constructions of KEM-PC and DEM-PC, in Section 3. We present the HES-PC examination containing of the security and performance resolutions in Sections 4 and 5, respectively. Finally, the result of this work is explained in Section 6.

2. Polar Codes

Polar codes [3] are a category of capacity attaining linear block codes that exploit low complexity non systematic encoder and successive cancelation (SC) decoder. Let us suppose $W : \mathcal{X} \rightarrow \mathcal{Y}$ as a BMS channel whose input alphabet is $\mathcal{X} = \{0, 1\}$ and output alphabet is $\mathcal{Y} = \{0, 1\}$ and the transition probabilities are noticed as $\{W(y|x), x \in \mathcal{X}, y \in \mathcal{Y}\}$. Let us notice $I(W)$ and $Z(W)$ as equations (1) and (2), respectively,

$$I(W) \triangleq \sum_{y\in\mathcal{Y}} \sum_{x\in\mathcal{X}} \frac{1}{2} W(y|x) \log \frac{W(y|x)}{\frac{1}{2}W(y|0) + \frac{1}{2}W(y|1)}, \tag{1}$$

$$Z(W) \triangleq \sum_{y \in \mathcal{Y}} \sqrt{W(y|0)W(y|1)}, \tag{2}$$

where $I(W) \in [0,1]$ is the symmetric bilateral data among input and output of channel W with consistent distribution on its input. When W is a binary memoryless symmetric (BMS) channel, $I(W)$ is defined as the capacity of W and hence can be applied as the rate measurement. Also, Bhattacharyya parameter of W is shown as $Z(W) \in [0,1]$ and can be applied as a reliability evaluation scale [3].

2.1. Creating the Generator Matrix of Polar Codes

Let us notice $n = 2^m, n \geq 1$, $F = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$, rate $R < I(W)$, dimension $k = nR$ and error correction capability t. A $k \times n$ generator matrix $G_{\mathcal{A}}$ is acquired for any (n,k) polar code as follows:

1. Compute the m-th kronecker product $G_n = F^{\otimes m}$ which acquires an $n \times n$ matrix. Then, label the rows of G_n from top to bottom as $i = 1, 2, \cdots, n$.
2. Calculate the Bhattacharyya parameters of all n sub-channels for $k = 1, 2, \cdots, 2^{m-1}$ with primary amount $Z(W_1^{(1)})$. In fact, create the parameters $Z(W_{2k}^{(i)}) = 2Z(W_k^{(i)}) - Z(W_k^{(i)})^2$ for $1 \leq i \leq k$. Also, calculate the parameters $Z(W_{2k}^{(i)}) = Z(W_k^{(i-k)})^2$ for $k+1 \leq i \leq 2k$. If W is a BEC(ϵ), we have $Z(W_1^{(1)}) = \epsilon$.
3. Form a permutation $\pi_n = (i_1, \cdots, i_n)$ for the indices set of n sub-channel $\mathcal{I}_n = \{1, 2, \cdots, n\}$ such that $Z(W_n^{(i_j)}) \leq Z(W_n^{(i_k)}), 1 \leq j < k \leq n$ is confirmed.
4. Calculate the data set $\mathcal{A} \subset \mathcal{I}_n$ whose sub-channels' indices are consistent with k farthest indices to right of the permutation π_n, i.e., $i_1, i_2, \cdots, i_k$. Afterwards, compute the frozen set $\mathcal{A}^c \subset \mathcal{I}_n$ whose sub-channels' indices depend on $(n-k)$ farthest indices to right of π_n, i.e. $i_{k+1}, i_{k+2}, \cdots, i_n$.
5. Aqcuire the generator matrix $G_{\mathcal{A}}$ by picking out k rows of the matrix G_n which are relevant to the sub-channels' indices of $\mathcal{A}$. If the sub-channel

$W_n^{(i)}$ is noticed, then choose the i-th row of G_n. Also, create the $(n-k)\times n$ matrix $G_{\mathcal{A}^c}$ by picking out $(n-k)$ rows of G_n relevant to the sub-channels' indices of of $\mathcal{A}^c$.

In general, the Bhattacharya parameters $\{Z(W_n^{(i)}), 1 \le i \le n\}$ of all sub-channels $\{W_n^{(i)}, 1 \le i \le n\}$ are created by the recursive formula presented in Step 2. The k rows of the matrix G_n are selected such that whose indices are relevant to the sub-channels with minimum conceivable Bhattacharya parameters to construct the generator matrix $G_{\mathcal{A}(s)}$,.

2.2. Polar Encoder

The input vector $u = (u_1, u_2, \cdots, u_n) = (u_{\mathcal{A}(s)}, u_{\mathcal{A}^c(s)})$ of non-systematic polar encoder is constituted of the data bits, i.e. a k-bit subvector $u_{\mathcal{A}(s)} = (u_i, i \in \mathcal{A}(s))$, and the frozen (fixed) vector, i.e. an $(n-k)$-bit subvector $u_{\mathcal{A}^c(s)} = (u_i, i \in \mathcal{A}^c(s))$. The data vector $u_{\mathcal{A}(s)}$ comprises information bits that may vary in each step of communication. However, the frozen vector constitutes the fixed parameters which is clear to polar decoder. The information vector u is converted to an n-bit codeword x as equation (3),

$$x = u_{\mathcal{A}(s)} G_{\mathcal{A}(s)} + u_{\mathcal{A}^c(s)} G_{\mathcal{A}^c(s)} = u_{\mathcal{A}(s)} G_{\mathcal{A}(s)} + c. \tag{3}$$

Since $c \triangleq u_{\mathcal{A}^c(s)} G_{\mathcal{A}^c(s)}$ has non-variable bits, such polar encoding is determined as non-systematic. The rate of code is computed as $R = |u_{\mathcal{A}(s)}|/|x| = |\mathcal{A}(s)|/n$ and is set by the dimension of $\mathcal{A}(s)$. Via the noiseless sub-channels, the data vector can be transmitted at the rate near to 1. Also, the frozen vector can be also transported at the rate near 0 by means of noisy sub-channels [3].

2.3. Polar Decoder

Suppose that $\mathbf{x}$ is an n-bit polar codes' codeword and suppose that y be its relevant channel output vector. The estimation of input vector when the data set $\mathcal{A}(s)$, the frozen vector $u_{\mathcal{A}^c(s)}$ and the channel output vector y are familiar is the main goal of SC decoding as equation (4),

$$\hat{u}_i = \begin{cases} u_i, & if\ i \in \mathcal{A}^c(s) \\ h_i(y_1^n, \hat{u}_1^{i-1}), & if\ i \in \mathcal{A}(s) \end{cases}. \tag{4}$$

In this way, the decision subordinates $h_i : \mathcal{Y}^n \times \mathcal{X}^{i-1} \to \mathcal{X}, i \in \mathcal{A}(s)$, are calculated as equation (5) for all $y_1^n \in \mathcal{Y}^n, \hat{u}_1^{i-1} \in \mathcal{X}^{i-1}$,

$$h_i(y_1^n, \hat{u}_1^{i-1}) \triangleq \begin{cases} 0, & if\ \dfrac{W_n^{(i)}(y_1^n, \hat{u}_1^{i-1}|0)}{w_n^{(i)}(y_1^n, \hat{u}_1^{i-1}|1)} \geq 1 \\ 1, & otherwise \end{cases}. \tag{5}$$

When the vector $\mathbf{y}$ is received and with the knowledge of previous estimated data bits $\hat{u}_1^{i-1}$, the information bits $u_i, i \in \mathcal{A}(s)$ are approximated one by one by exploiting the i-th decision element. As mentioned before, the SC decoder knows the parameter of constant bits, $u_i, i \in \mathcal{A}^c(s)$. The error probability under SC decoding for any given B-DMC W is upper bounded as inequality (6) [3],

$$P_e \leq \sum_{i \in A} Z_{N,i}. \tag{6}$$

To attain the trustworthy communications using SC decoding, the inequality (7) should be convinced [10],

$$R \leq I(W) - n^{-1/\mu}. \tag{7}$$

In this case, μ is named as scaling exponent and its dimension for BEC is equal to 3.627. The biggest dimension of R that satisfies such inequality is named R_0. In [11], it is indicated that for the polar codes with rate $R \leq I(W) - n^{-1/3.627}$ under successive cancelation (SC) decoding, reliable communication can be obtained over a BMS channel. The maximum rate in such inequality is named as cutoff rate R_0.

3. The Proposed Polar Code-based Hybrid Encryption Scheme

As mentioned in Sec. 1, the proposed HES-PC contains of KEM-PC and DEM-PC. The presented KEM-PC contains of three parts: (i) KEM-PC key generation algorithm, called as KEM-PC.Gen; (ii) KEM-PC encapsulation algorithm, called as KEM-PC.Encaps; and (iii) KEM-PC decapsulation algorithm, called as KEM-PC.Decaps. Also, the presented DEM-PC contains of two algorithms: (i) DEM-PC encapsulation algorithm, called as DEM-PC.Encaps; and (ii) DEM-PC decapsulation algorithm, called as DEM-PC.Decaps. In this section, first we describe the overal construction of the proposed HES-PC. Then, we describe the algorithms of KEM-PC and DEM-PC, respectively. Figures 1 and 2 illustrate the block diagram and overal construction of the proposed HES-PC between legitimate parties, i.e. Alice and Bob, respectively. As shown in these figures, Alice wants to send a k-bit message vector m to Bob. In this case, first the symmetric key $\mathcal{K}$ should be shared between Alice and Bob by using KEM-PC. The KEM-PC.Gen obtains the seed (s) as input and generates a public key (PK) and a secret key (SK). The KEM-PC.Encaps, obtains PK as input and outputs the symmetric key $\mathcal{K}$ and ciphertext $c^{'}$, i.e. the encapsulation of $\mathcal{K}$ via a key derivation function (KDF). Also, the DEM-PC.Encaps, takes symmetric key $\mathcal{K}$ and the plaintext m as input and outputs the ciphertext c. Actually, the ciphertext c disguises m indistinguishably from a random string. Then, the pair $(c, c^{'})$ is sent through an insecure channel. At the receiver side, given SK and $c^{'}$, $\mathcal{K}$ can be obtained by executing the KEM-PC.Decaps. The ciphertext c is decoded and decrypted in a unique stage using the DEM-PC.Decaps. By this way, the plaintext m is obtained. Table 1 shows the data bandwidth of HES-PC.

The KEM-PC.Gen is explained as Algorithm 1 in which the seed (s) of applied counter mode-based deterministic random bit generator (CTR-DRBG) [12] is considered as its input. Moreover, to coceal the generator matrix $G_{\mathcal{A}'(\mathsf{s})}$,

Table 1: Data Bandwidth of HES-PC.

Data Flow	Data	Size (bits)
Bob→Alice	PK	$k(n-k)$
Alice→Bob	$(c, c^{'})$	$2n$

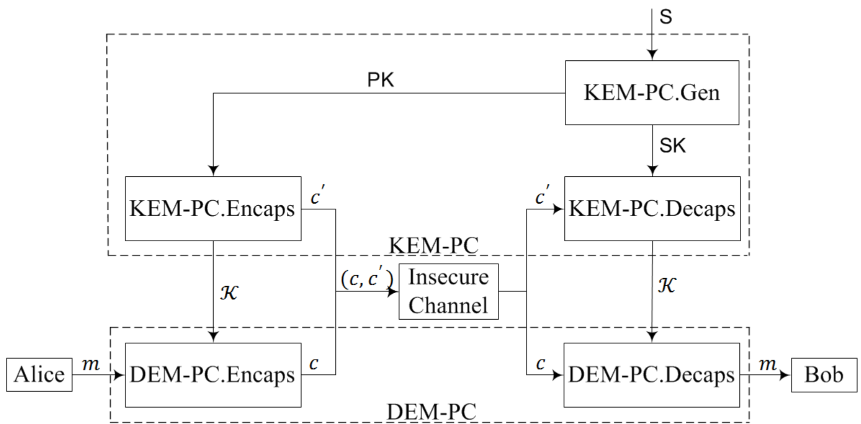

Figure 1: The block diagram of the proposed HES-PC.

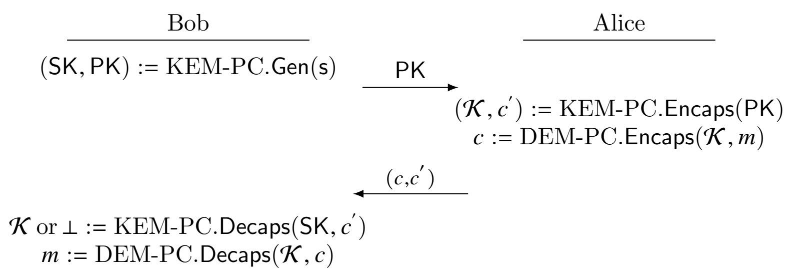

Figure 2: The overal construction of the proposed HES-PC.

k indices are raised at random from $\mathcal{I}_\setminus$ shown by $\mathcal{A}'(\mathsf{s})$. In fact, concealing the generator matrix $G_{\mathcal{A}'(\mathsf{s})}$ is necessary for secure polar coding.

Algorithm 1: KEM-PC.Gen

Data: the polar code parameters n, k and t.

Input: the seed (S) of applied CTR-DRBG.

Output: the public key (PK) and the secret key (SK).

1. Calculate the m-th kronecker product $G_n = F^{\otimes m}$ as an $n \times n$ matrix and denote its rows as $i = 1, 2, \cdots, n$ from top to bottom.
2. For $k = 1, 2, \cdots, 2^{m-1}$ with initial condition $Z(W_1^{(1)})$, calculate all n sub-channels Bhattacharyya parameters as $Z(W_{2k}^{(i)}) = 2Z(W_k^{(i)}) - Z(W_k^{(i)})^2$, $1 \leq i \leq k$ and $Z(W_{2k}^{(i)}) = Z(W_k^{(i-k)})^2$, $k+1 \leq i \leq 2k$.

3. Construct a permutation $\pi_n = (i_1, \cdots, i_n)$ for the set of n sub-channels' indices $\mathcal{I}_n = \{1, 2, \ldots, n\}$ such that $Z(W_n^{(i_j)}) \leq Z(W_n^{(i_k)}), 1 \leq j < k \leq n$.
4. Generate a secret data set $\mathcal{A}'(\mathsf{s}) \subset \mathcal{I}_n$ through a CTR-DRBG by which its k indices are picked out at random from n indices in π_n, i.e., $i_1, i_2, \cdots, i_n$.
5. Construct a secret generator matrix $G_{\mathcal{A}'(\mathsf{s})}$ for an (n, k) polar code by picking out k rows of G_n respective to $\mathcal{A}'(\mathsf{s})$.
6. Complement the $\mathcal{A}'(\mathsf{s})$ to obtain $\mathcal{A}'^c(\mathsf{s})$ and consider it as SK.
7. Generate P_1' as an $n \times k$ submatrix with k 1's, each placed in j-th, $j \in \mathcal{A}'(\mathsf{s})$ row of its k columns, respectively.
8. Generate P_2' as an $n \times (n-k)$ submatrix with $(n-k)$ 1's putted in j-th, $j \in \mathcal{A}'^c(\mathsf{s})$ row of $(n-k)$ columns.
9. Consider the interpolation of submatrices P_1' and P_2' as a permutation matrix $P' = [P_1' | P_2']$.
10. Construct the nonsingular submatrix $S'_{k \times k} = (G_n)_{\mathcal{A}'(\mathsf{s})\mathcal{A}'(\mathsf{s})}$ whose k rows and k columns are picked out from G_n corresponding to $\mathcal{A}'(\mathsf{s})$.
11. Construct the encapsulation matrix $G'_{k \times n} = S'^{-1} G_{\mathcal{A}'(\mathsf{s})} P' = [I_k | Q]$ and extract the submatrix $Q_{k \times (n-k)}$ as PK.
12. Return SK and PK.

In Algorithm 2, the KEM-PC.Encaps is presented in which the symmetric key $\mathcal{K}$ is acquired via a key derivation function (KDF). A used KDF : $\{0, 1\}^n \rightarrow \{0, 1\}^{l_{\mathcal{K}}}$ in the KEM-PC is a cryptographic hash function with a changeable output symmetric key length $l_{\mathcal{K}}$. The symmetric key $\mathcal{K} = \{\alpha, \beta, I_0\}$, contains of random integers α, β and I_0 such that $\{\lceil \alpha \rceil, \lceil \beta \rceil, \lceil I_0 \rceil\} \leq n^{\gamma}$, where $\lceil \alpha \rceil$ denotes the upper bound of α. Also, γ is an integer which is picked out depending on the requested symmetric key length $l_{\mathcal{K}}$.

Algorithm 2: KEM-PC.Encaps

Data: polar code parameters n, k and t.

Input: the public key PK.

Output: the symmetric key $\mathcal{K}$ and the ciphertext c.

1. Generate the encryption matrix $G' = [I_k | Q]$.

2. Pick out random error vector $e^{'} \in F_2{}^n$, $w_H(e^{'}) \leq t$ and a random data $m^{'} \in F_2{}^k$.
3. Generate the symmetric key $\mathcal{K} = \mathrm{KDF}(m^{'} \| e^{'}, l_{\mathcal{K}})$.
4. Generate the ciphertext as $c^{'} = m^{'} G^{'} + e^{'}$.
5. Return $c^{'}$ and $\mathcal{K}$.

In Algorithm 3, the KEM-PC.Decaps is explained in which the symmetric key $\mathcal{K}$ is recovered by using SC decoding.

Algorithm 3: KEM-PC.Decaps

Data: polar code parameters n, k and t.

Input: the ciphertext $c^{'}$ and the secret key SK.

Output: the symmetric key $\mathcal{K}$ or failure symbol $\perp$.

1. Complement $\mathcal{A}^{'c}(\mathsf{s})$ to acquire $\mathcal{A}^{'}(\mathsf{s})$.
2. Generate a secret generator matrix $G_{\mathcal{A}^{'}(\mathsf{s})}$ for an (n, k) polar code by picking out k rows of G_n corresponding to $\mathcal{A}^{'}(\mathsf{s})$.
3. Generate $P_1^{'}$ as an $n \times k$ submatrix with k 1's, each located in j-th, $j \in \mathcal{A}^{'}(\mathsf{s})$ row of its k columns, respectively.
4. Generate $P_2^{'}$ as an $n \times (n-k)$ submatrix with $(n-k)$ 1's, each located in j-th, $j \in \mathcal{A}^{'c}(\mathsf{s})$ row of $(n-k)$ columns.
5. Set the concatenation of submatrices $P_1^{'}$ and $P_2^{'}$ as a permutation matrix $P^{'} = [P_1^{'} | P_2^{'}]$.
6. Compute $c^{''} = c^{'} P^{'-1} = m^{'} S^{'-1} G_{\mathcal{A}^{'}(\mathsf{s})} + e^{'} P^{'-1}$.
7. Compute SC Decoding$\{c^{''}, \mathcal{A}^{'}(\mathsf{s})\}$ and acquire its outputs, i.e, $e^{''} = e^{'} P^{'-1}$ and $u = m^{'} S^{'-1}$
8. If SC decoding fails or $w_H(e^{''}) \neq t$, output $\perp$ and halt.
9. Generate the nonsingular submatrix $S_{k \times k}^{'} = (G_n)_{\mathcal{A}^{'}(\mathsf{s}) \mathcal{A}^{'}(\mathsf{s})}$ whose k rows and k columns are picked out from G_n corresponding to $\mathcal{A}^{'}(\mathsf{s})$.
10. Obtain $e^{'} = e^{''} P^{'}$ and $m^{'} = u S^{'}$.
11. Generate the symmetric key $\mathcal{K} = \mathrm{KDF}(m^{'} \| e^{'}, l_{\mathcal{K}})$.
12. Return $\mathcal{K}$

In algorithm 4, DEM-PC.Encaps is described in which the plaintext m is encapsulated to the ciphertext c by using the symmetric key $\mathcal{K}$.

Algorithm 4: DEM-PC.Encaps

Data: the polar code parameters n, k and t.

Input: the symmetric key $\mathcal{K} = \{\alpha, \beta, I_0\}$ and the message m.

Output: the ciphertext c.

1. Compute k indices as $I_j = (\alpha I_{j-1} + \beta)\, mod n$, $1 \leq j \leq k$.
2. Sort k produced indices $I_j, 1 \leq j \leq k$ from the minimum parameter to the maximum parameter.
3. Generate a secret data set $\mathcal{A}(s) = \{a_1(s), a_2(s), \ldots, a_k(s)\}$ such that for $1 \leq i < l \leq k$, we have $a_i(s) < a_l(s)$, $a_1(s) = minI_j$ and $a_k(s) = maxI_j$.
4. Generate a secret generator matrix $G_{\mathcal{A}(\mathsf{s})}$ by picking out k rows of G_n corresponding to $\mathcal{A}(\mathsf{s})$.
5. Generate P_1 as an $n \times k$ submatrix with k 1's, each located in j-th, $j \in \mathcal{A}(\mathsf{s})$ row of its k columns, respectively.
6. Generate P_2 as an $n \times (n-k)$ submatrix with $(n-k)$ 1's, each located in j-th, $j \in \mathcal{A}^c(\mathsf{s})$ row of $(n-k)$ columns.
7. Set the concatenation of submatrices P_1 and P_2 as a permutation matrix $P = [P_1|P_2]$.
8. Generate the nonsingular submatrix $S_{k\times k} = (G_n)_{\mathcal{A}(\mathsf{s})\mathcal{A}(\mathsf{s})}$ whose k rows and k columns are picked out from G_n corresponding to $\mathcal{A}(\mathsf{s})$.
9. Select random perturbation vector $e \in F_2{}^n$, $w_H(e) \leq t$.
10. Generate the ciphertext as $c = (mSG_{\mathcal{A}(s)} + e)P$.
11. Return c.

In Algorithm 5, the DEM-PC.Decaps is proposed in which the message m is acquired from the ciphertext c via a data decapsulation mechanism.

Algorithm 5: DEM-PC.Decaps

Data: the polar code parameters n, k and t.

Input: the ciphertext c and the symmetric key $\mathcal{K}$.

Output: the message m.

1. Compute k indices $I_j = (\alpha I_{j-1} + \beta)\, mod n$, $1 \le j \le k$.
2. Sort k produced indices $I_j, 1 \le j \le k$ from the minimum parameter to the maximum parameter.
3. Generate a secret data set $\mathcal{A}(s) = \{a_1(s), a_2(s), \dots, a_k(s)\}$ such that for $1 \le i < l \le k$, we have $a_i(s) < a_l(s)$, $a_1(s) = min I_j$ and $a_k(s) = max I_j$.
4. Generate P_1 as an $n \times k$ submatrix with k 1's, each located in j-th, $j \in \mathcal{A}(\mathsf{s})$ row of its k columns, respectively.
5. Generate P_2 as an $n \times (n-k)$ submatrix with $(n-k)$ 1's, each located in j-th, $j \in \mathcal{A}^c(\mathsf{s})$ rows of $(n-k)$ columns.
6. Set the concatenation of submatrices P_1 and P_2 as a permutation matrix $P = [P_1|P_2]$.
7. Compute $c^{'} = cP^{-1} = mSG_{\mathcal{A}(\mathsf{s})} + e$.
8. Compute SC Decoding$\{c^{'}, \mathcal{A}(\mathsf{s})\}$ and acquire its outputs, i.e, $u = mS$
9. Generate the nonsingular submatrix $S_{k\times k} = (G_n)_{\mathcal{A}(\mathsf{s})\mathcal{A}(\mathsf{s})}$ whose k rows and k columns are picked out from G_n corresponding to $\mathcal{A}(\mathsf{s})$.
10. Obtain $m = uS^{-1}$.
11. Return the message m.

4. Security Analysis

In this section, we notice various attacks to investigate the security level of the described HES-PC.

4.1. The Exhaustive Search (ES) Attack

In the exhaustive search attack, an adversary checks and tests all possible states of symmetric key $\mathcal{K}$ and secret key SK in the HES-PC until $\mathcal{K}$ and SK are found. In the proposed HES-PC, the possible numbers of symmetric key $\mathcal{K}$ and secret key SK are computed as $\mathcal{N}_{\mathcal{K}} = \mathcal{N}_{\alpha} = \mathcal{N}_{\beta} = \mathcal{N}_{I_0} = n^{\gamma}$ and $\mathcal{N}_{\mathsf{SK}} = \mathcal{N}_{\mathcal{A}^c(\mathsf{s})} = 2^{n-k}$, respectively. On the other hand, to acquire the symmetric key $\mathcal{K}$, the adversary must search α, β and I_0, at the same time. Hence, the estimation of required time for obtaining the parameters of symmetric key $\mathcal{K}$

and secret key SK of HES-PC are computed as $O(n^{3\gamma})$ and $O(2^{n-k})$, respectively. In Table 2, the estimation of required time (work factor) for ES attack, shown by $\mathsf{WF}_{\mathsf{ES}}$, is computed for distinct HES-PC instances. It is obvious that all HES-PC instances are secure opposed to ES attack.

4.2. The Message-Resend Attack

In the message-resend attack [13], the adversary deletes, adds or intercepts the ciphertext pair $(c, c^{'})$ of HES-PC and his goal is to acquire the perturbation vectors $(e, e^{'})$. Let us suppose that the sender encapsulates the plaintext pair $(m, m^{'})$ as $c_1 = (mSG_{\mathcal{A}(s)} + e_1)P$ and $c^{'}_1 = m^{'}S^{'-1}G_{\mathcal{A}^{'}(\mathsf{s})}P^{'} + e^{'}_1$, respectively. Then, the ciphertext pair $(c_1, c^{'}_1)$ is sent to the legitimate receiver and it is assumed that attacker changes it. In such case, the errors will be appeared in the decapsulated message pair $(m, m^{'})$ and Bob figures out that the ciphertext pair $(c_1, c^{'}_1)$ are changed. Hence, the receiver demands the sender to encapsulate the plaintext pair $(m, m^{'})$ again. Now, the sender encapsulates $(m, m^{'})$ using DEM-PC.Encaps and KEM-PC.Encaps again as $c_2 = (mSG_{\mathcal{A}(s)} + e_2)P$ and $c^{'}_2 = m^{'}S^{'-1}G_{\mathcal{A}^{'}(\mathsf{s})}P^{'} + e^{'}_2$ and sends $(c_2, c^{'}_2)$ to the receiver. At this time, the attacker has two distinct ciphertext pairs $(c_1, c^{'}_1)$ and $(c_2, c^{'}_2)$ of the same plaintext pair $(m, m^{'})$ and his aim is to acquire $(e_1, e^{'}_1)$ and $(e_2, e^{'}_2)$ from $c_1 + c_2 = (e_1 + e_2)P$ and $c^{'}_1 + c^{'}_2 = e^{'}_1 + e^{'}_2$. In the proposed HES-PC, because of exploiting the characteristics of polar codes, the perturbation vectors $(e_1, e^{'}_1)$ and $(e_2, e^{'}_2)$ are chosen randomly and their Hamming weights are changeable. Hence, the Hamming weight of $c_1 + c_2 = (e_1 + e_2)P$ and $c^{'}_1 + c^{'}_2 = e^{'}_1 + e^{'}_2$ do not necessarily have distinguished parameters. In this case, attacker cannot recognize a message-resend

Table 2: Work Factor of ES attack ($\mathsf{WF}_{\mathsf{ES}}$) for various HES-PC instances.

Instance	(n, k)	Finding $\mathcal{K}$	Finding SK
HES-PC I	$(1024, 816)$	$O(2^{120})$	$O(2^{208})$
HES-PC II	$(2048, 1632)$	$O(2^{132})$	$O(2^{416})$
HES-PC III	$(4096, 3264)$	$O(2^{144})$	$O(2^{832})$
HES-PC IV	$(8192, 6528)$	$O(2^{156})$	$O(2^{1664})$

condition by using the Hamming weight of $c_1 + c_2$ or $c_1^{'} + c_2^{'}$. Moreover, Oscar cannot recognize the entries of $(e_1, e_1^{'})$ and $(e_2, e_2^{'})$ even if he can discover when the sender resends $(c_2, c_2^{'})$. Therefore, the adversary faces several challenges in the HES-PC to acquire $(e_1, e_1^{'})$ and $(e_2, e_2^{'})$ from c_1+c_2 and $c_1^{'}+c_2^{'}$ and therefore this attack fails.

4.3. The Rao-Nam (RN) Attack

In the RN attack [14], the main goal is to acquire the encryption matrix $SG_{\mathcal{A}(s)}P = [g_{i,\mathcal{A}(s)}]$, $i = 1, 2, \ldots, k$ from the plaintext-encapsulated ciphertext pairs (m_i, c_i) of DEM-PC.Encaps, where $g_{i,\mathcal{A}(s)}$ is the i-th row of matrix $SG_{\mathcal{A}(s)}P$. Let us assume that m_1 and m_2 be two plaintext vectors which differs in the i-th, $i = 1, 2, \ldots, k$ bit. Let $e_1 \in F_2{}^n$, $w_H(e_1) \leq t$ and $e_2 \in F_2{}^n$, $w_H(e_2) \leq t$ be two perturbation vectors which are picked out randomly. Also, let $c_1 = (m_1 SG_{\mathcal{A}(s)} + e_1)P$ and $c_2 = (m_2 SG_{\mathcal{A}(s)} + e_2)P$ be two distinct ciphertexts which are encapsulated from the plaintexts m_1 and m_2, respectively. The distinction vector $c_1 - c_2$ is then calculated as $c_1 - c_2 = (m_1 - m_2)SG_{\mathcal{A}(s)}P + (e_1 - e_2)P$. Also, the i-th row of $SG_{\mathcal{A}(s)}P$, i.e., $g_{i,\mathcal{A}(s)}$, can be acquired through the equation (8),

$$g_{i,\mathcal{A}(s)} = c_1 - c_2 - (e_1 - e_2)P. \tag{8}$$

If $(w_H(e_1 - e_2))P/n \ll 1$, the distinction vector $c_1 - c_2$ shows the estimation of $g_{i,\mathcal{A}(s)}$. In the proposed HES-PC, since the perturbation vectors e_1 and e_2 are random n-bit vectors, it is obvious that the upper bound on the Hamming weight of $(e_1 - e_2)P$ is calculated as $2n$. Therefore, the distinction vector $c_1 - c_2 - (e_1 - e_2)P$ does not necessarily show the estimation of $g_{i,\mathcal{A}(s)}$. In this way, $SG_{\mathcal{A}(s)}P = [g_{i,\mathcal{A}(s)}]$, $i = 1, 2, \ldots, k$ cannot be acquired. In the sequel, we also calculate the estimation of required time for obtaining the matrix $SG_{\mathcal{A}(s)}P$ from the same plaintext m. Let $c_j = m(SG_{\mathcal{A}(s)} + e_j)P$ and $c_k = m(SG_{\mathcal{A}(s)} + e_k)P$ be two distinct n-bit encapsulated ciphertexts of HES-PC acquired from the same plaintext m. The distinction of c_j and c_k is calculated as $c_j - c_k = (e_j - e_k)P$. This stage should be iterated until one of the parameters calculated for

$(e_j - e_k)P$ satisfies the equation (8). In such case, the complete construction of $SG_{\mathcal{A}(s)}P$ must be confirmed, as the accuracy of each row vector $g_{i,\mathcal{A}(s)}$ cannot be confirmed independently. Because of the quantity of distinct perturbation vectors of DEM-PC is calculated as $\mathcal{N}_e = \binom{n}{t}$, the quantity of all possible pairs (e_j, e_k) is calculated as $\binom{\mathcal{N}_e}{2} = (\mathcal{N}_e^2 - \mathcal{N}_e)/2$. In addition, the vector $g_{i,\mathcal{A}(s)}$ should be acquired for each of the k rows of $SG_{\mathcal{A}(s)}P$. Hence, the estimation of required time is calculated as WF= $\Omega(\mathcal{N}_e^k)$. The estimation of required time on RN attack for $(2048, 1632)$ polar code is too large and hence such attack is infeasible.

4.4. Indistinguishability under Chosen-Plaintext Attack (IND-CPA)

In this section, it is shown that the HES-PC is IND-CPA secure if the used KDF of KEM-PC is noticed as a random oracle. In fact, we define the IND-CPA of KEM-PC using two games 1 and 2 [15]. In the game 1, the security parameter λ is fed into the KEM-PC.Gen and then the secret key $\mathcal{A}^c(\mathsf{s})$ and public key Q are noticed as its output. Then, the KEM-PC.Encaps is run using the public key Q and its outputs, i.e. the ciphertext c and the encapsulated key $\mathcal{K}$, are acquired. Then, the outputs of honest Game 1 are noticed as $(Q, c, \mathcal{K})$. In the dishonest game 2, the challenger, replaces the key $\mathcal{K}$ by a random parameter $\mathcal{K}^{'}$. The outputs of Game 2 are $(Q, c, \mathcal{K})$. In fact, the only distinction between Game 1 and Game 2 is the parameter of $\mathcal{K}$ which is replaced by $\mathcal{K}^{'}$. In this case, the KEM-PC is IND-CPA secure if $\mathcal{K}$ and $\mathcal{K}^{'}$ are computationally indistinguishable. In Game 1, the encapsulated key is acquired as $\mathcal{K} = \mathrm{KDF}(m \| e, l_{\mathcal{K}})$. The output of KDF is pseudorandom and hence the only method to distinguish $\mathcal{K}$ and $\mathcal{K}^{'}$ is acquiring the vector $m^{'} \| e^{'}$. This is equivalent to decode the ciphertext c which is noticed to be hard. Therefore, since the output of used KDF is pseudorandom and obtaining $m^{'} \| e^{'}$ is infeasible, KEM-PC is IND-CPA secure.

4.5. Indistinguishability under Adaptive Chosen Ciphertext Attack (IND-CCA2)

In the proposed HES-PC, KEM-PC has a systematic encapsulation matrix $G^{'} = [I_k | Q]$ by which the PK size needs $k(n-k)$ bits in lieu of kn bits. However, the systematic form of $G^{'}$ helps the adversary to extract $m^{'}$ from $c^{'}$, without regaining SK. Therefore, appropriate transformations should be applied to acquire an IND-CCA2 mode of KEM-PC. In this section, we apply the Kobara-Imai (KI) conversion [16] to have an IND-CCA2 secure KEM-PC.Encaps and KEM-PC.Decaps as algorithms 4 and 5, respectively. In the step 4 of algorithm 5, because of XOR operation between r and $\overline{m}\|const$, the random vector y_1 is acquired. Also, the seed S is XORed by $\mathrm{KDF}(y_1)$ to acquire the secret seed and satisfying the IND-CCA2 secure property. The vector $y_2\|y_1$ is splitted into y_3, y_4 and y_5, where y_3 is k LSBs of $y_2\|y_1$ and y_4 is noticed as $\lfloor log_2 C\,(n,t)\rfloor$ LSBs of remaining bits. By using the bijective subordinate Conv, y_4 is converted to its corresponding perturbation vector $\overline{e}$. The $|\overline{m}| + |const| + |\mathsf{s}| - |y_4| - k$ MSBs of $y_2\|y_1$ is remarked as y_5 and is prefixed to $\overline{c}$ to attain $c^{'}$. In Algorithm 6, similar inverse steps are executed to acquire the decapsulated key $\mathcal{K}$ from the ciphertext $c^{'}$.

Algorithm 5: IND-CCA2 Secure KEM-PC.Encaps

Data: polar code parameters n, k and t.

Input: the seed (s) and the public key PK.

Output: the encapsulated key $\mathcal{K}$ and the ciphertext $c^{'}$.

1. Generate a pseudorandom sequence $r = \text{CTR-DRBG}(\mathsf{s})$.
2. Select a random plaintext $m \in {F_2}^k$.
3. Generate $\overline{m} = \mathrm{Prep}(m)$ and $y_1 = r \oplus (\overline{m}\|const)$.
4. Generate $y_2 = \mathsf{s} \oplus \mathrm{KDF}(y_1)$ and set $y_5\|y_4\|y_3 = y_2\|y_1$.
5. Generate $\overline{e} = \mathrm{Conv}(y_4\|\mathrm{KDF}(y_3\|r))$ and $G^{'} = [I_k|Q]$.
6. Compute the encapsulated key $\mathcal{K} = \mathrm{KDF}(m\|\overline{e}, l_{\mathcal{K}})$.
7. Compute $\overline{c} = (y_3\|r)G^{'} + \overline{e}$ and the ciphertext $c^{'} = y_5\|\overline{c}$.
8. Return $c^{'}$ and $\mathcal{K}$.

Algorithm 6: IND-CCA2 Secure KEM-PC.Decaps

Data: polar code parameters n, k and t.

Input: the ciphertext c and the secret key (SK) $\mathcal{A}^c(\mathsf{s})$.

Output: the decapsulated key $\mathcal{K}$ or failure symbol $\perp$.

1. Generate $y_5 = \mathrm{MSB}_{\mathrm{len}(c)-n}(c^{'})$ and $\overline{c} = \mathrm{LSB}_n(c^{'})$.
2. Generate P and complement $\mathcal{A}^c(\mathsf{s})$ to acquire $\mathcal{A}(\mathsf{s})$.
3. Compute $c^{''} = \overline{c}P^{-1} = (y_3\|r)S^{-1}G_{\mathcal{A}(\mathsf{s})} + \overline{e}P^{-1}$.
4. Compute SC Decoding$\{c^{''}, \mathcal{A}(\mathsf{s})\}$ and acquire its outputs.
5. If decoding was successful, $e^{''} = \overline{e}P^{-1}$ and $\overline{u} = (y_3\|r)S^{-1}$.
6. If decoding fails or $w_H(e^{'}) \neq t$, output $\perp$ and halt.
7. Generate $S^{'}$ and acquire $y_3\|r = \overline{u}S^{'}$ and $\overline{e} = e^{''}P$.
8. Compute $(y_4\|\mathrm{KDF}(y_3\|r)) = \mathrm{Conv}^{-1}(\overline{e})$.
9. Obtain $y_2\|y_1 = y_5\|y_4\|y_3$ and $\mathsf{s} = y_2 \oplus \mathrm{KDF}(y_1)$.
10. Generate $r = \text{CTR-DRBG}(\mathsf{s})$ and $(\overline{m}\|const^{'}) = y_1 \oplus r$.
11. If $const^{'} = const$, obtain $m^{'} = \mathrm{Prep}^{-1}(\overline{m})$, else, reject $c^{'}$.
12. Compute $\mathcal{K} = \mathrm{KDF}(m^{'}\|\overline{e}, l_{\mathcal{K}})$ and return $\mathcal{K}$.

4.6. Information Set Decoding (ISD) Attack

In the information set decoding (ISD) attack, the main goal is to discover the perturbation vector pair $(e, e^{'})$ in the ciphertext pair $(c, c^{'})$ by finding the codewords with minimum Hamming weight for an (n, k) polar code. We exploit the Stern algorithm [17] to analyse the security of HES-PC opposed to the information set decoding (ISD) attack. Let us notice $H = H_{(j,m)}, 1 \leq j \leq n-k, 1 \leq m \leq n$ as a parity check matrix of (n, k) polar code. In fact, the main goal of Stern algorithm is to solve the equation $Hx = 0$ in which $w_H(x) = \omega$. The estimation of required time for ISD attack is acquired as equation (9),

$$WF_{ISD}(n, k, \omega) = Cost_{ST}/P_{ST}, \tag{9}$$

where, $Cost_{ST}$ is the cost of executing the binary calculation for each repetition of ISD algorithm opposed to the HES-PC and obtained as equation (10),

$$Cost_{ST} = 1/2(n-k)^2(n+k) + 2plC\,(k/2, p) + 2^{-l+1}p(n-k)C\,(k/2, p)^2. \tag{10}$$

Also, P_{ST} is defined as the possibility of finding a codeword with Hamming weight ω as equation (11),

$$P_{ST} = C\,(k/2, p)^2 C\,(n-k-1, \omega-2p)\,C\,(n, \omega)^{-1}, \tag{11}$$

where, $0 \le p \le \omega$ and $0 \le l \le n-k$ are integers. In Table 3, the estimation of required time for ISD attack, shown as $\mathsf{WF}_{\mathsf{ISD}}$, is described for distinct HES-PC instances. As shown in this table, all HES-PC instances have proper estimation of required time to secure opposed to the ISD attack.

5. Performance Analysis

We assess the key length, computational complexity and decapsulation failure rate (DFR) of the described HES-PC. The performance level of HES-PC is investigated according to the key size, decapsulation failure rate and computational complexity.

5.1. Key Size

In the proposed HES-PC, the size of the symmetric key $\mathcal{K} = \{\alpha, \beta, I_0\}$ is computed as $\mathcal{M}_{\mathcal{K}} = \mathcal{M}_\alpha + \mathcal{M}_\beta + \mathcal{M}_{I_0} \le 3\gamma log_2 n$ bits. Also, the size of public key $\mathsf{PK} = Q$ is acquired as $\mathcal{M}_{\mathsf{PK}} = k(n-k)$ bits. Also, each index of secret key $\mathsf{SK} = \mathcal{A}^c(\mathsf{s})$ may be needs at most $\lceil log_2(n) \rceil$ bits to be saved. Hence, the maximum memory to store $\mathcal{A}^c(\mathsf{s})$ is acquired as $\mathcal{M}_{\mathsf{SK}} = (n-k)\lceil log_2(n) \rceil$ bits. Table 4 shows the key lengths of the described HES-PC for various samples.As illustrated in this table, enlarging the polar code lengths has a little effect on the key size of the described HES-PC.

5.2. Computational difficulty

For the proposed HES-PC, the computational difficulty comprises DEM-PC's difficulty and KEM-PC's difficulty. The difficulty of DEM-PC.Encaps is

Table 3: ISD attack work factor ($\mathsf{WF}_{\mathsf{ISD}}$) for various HES-PC instances.

Instance	(n, k)	(p, l)	$\mathsf{WF}_{\mathsf{ISD}}(log_2)$
HES-PC I	$(1024, 816)$	$(5, 39)$	162.581
HES-PC II	$(2048, 1632)$	$(7, 59)$	274.035
HES-PC III	$(4096, 3264)$	$(9, 79)$	$\gg 80$
HES-PC IV	$(8192, 6528)$	$(11, 99)$	$\gg 80$

computed as $C_{DEM-PC.\mathsf{Encaps}} = C_{\mathsf{mul}}(mSG_{\mathcal{A}(\mathsf{s})}P) + C_{\mathsf{mul}}(eP) + C_{\mathsf{add}}(eP)$, where $C_{\mathsf{mul}}(mSG_{\mathcal{A}(\mathsf{s})}P) = O(kn)$ is the difficulty of multiplication between m and encryption matrix $SG_{\mathcal{A}(\mathsf{s})}P$; $C_{\mathsf{mul}}(eP) = O(n)$ is the difficulty of multiplication between e and permutation matrix P and $C_{\mathsf{add}}(eP) = O(n)$ is the quantity of binary subordinates for adding n-bit perturbation vector eP. The difficulty of DEM-PC.Decaps is calculated as $C_{DEM-PC.\mathsf{Decaps}} = C_{\mathsf{mul}}(cP^{-1}) + C_{\mathsf{SC}}(c^{'}) + C_{\mathsf{mul}}(uS^{-1})$, where $C_{\mathsf{mul}}(cP^{-1}) = O(n)$, is the essential binary subordinates to multiply n-bit vectors c by the matrix P^{-1}. Also, $C_{\mathsf{SC}}(c^{'}) = O(nlogn)$ is the SC decoding's difficulty of $c^{'}$ and $C_{\mathsf{mul}}(uS^{-1}) = O(k^2)$ is the quantity of demanded operations for multiplying u by S^{-1}.

The difficulty of KEM-PC.Encaps can be represented as $C_{KEM-PC.\mathsf{Encaps}} = C_{\mathsf{mul}}(m^{'}G^{'}) + C_{\mathsf{add}}(e^{'}) + C_{\mathsf{KDF}}(m^{'}\|e^{'})$, where $C_{\mathsf{mul}}(m^{'}G^{'}) = O(n(n-k))$ is the difficulty of multiplication between $m^{'}$ and encapsulation matrix $G^{'} = [I_k|Q]$; $C_{\mathsf{add}}(e^{'}) = O(n)$ is the quantity of binary subordinates for adding n-bit perturbation vector $e^{'}$ and $C_{\mathsf{KDF}}(m^{'}\|e^{'}) = O(n + k + l_k)$ is the time difficulty of used KDF. The difficulty of KEM-PC.Decaps is calculated as $C_{KEM-PC.\mathsf{Decaps}} = C_{\mathsf{mul}}(c^{'}P^{'-1}) + C_{\mathsf{SC}}(c^{''}) + C_{\mathsf{mul}}(uS^{'}) + C_{\mathsf{mul}}(e^{''}P^{'}) + C_{\mathsf{KDF}}(m^{'}\|e^{'})$, where $C_{\mathsf{mul}}(c^{'}P^{'-1}) = C_{\mathsf{mul}}(e^{''}P^{'}) = O(n)$, is the essential binary subordinates to multiply n-bit vectors $c^{'}$ and $e^{''}$ by the matrices $P^{'-1}$ and $P^{'}$, respectively [18]. $C_{\mathsf{SC}}(c^{''}) = O(nlogn)$ is the SC decoding's difficulty of $c^{''}$; $C_{\mathsf{mul}}(uS^{'}) = O(k^2)$ is the quantity of demanded operations for multiplying u by $S^{'}$.

Table 4: The key lengths of various HES-PC instances.

Instance	(n, k)	$\mathcal{M}_{\mathcal{K}}$(Bits)	$\mathcal{M}_{\mathsf{PK}}$(KBytes)	$\mathcal{M}_{\mathsf{SK}}$(KBytes)
HES-PC I	$(1024, 816)$	≤ 23.9	20.72	0.254
HES-PC II	$(2048, 1632)$	≤ 52.59	82.88	0.559
HES-PC III	$(4096, 3264)$	≤ 114.75	331.5	1.22
HES-PC IV	$(8192, 6528)$	≤ 248.625	1326	2.64

5.3. Decapsulation Failure Rate

In the proposed HES-PC, the probability of failed SC decoding in the KEM-PC.Decaps and DEM-PC.Decaps algorithms is defined as the decapsulation failure rate (DFR). Let $\mathcal{A}_{min} = \{i \in \mathcal{I}_n : \pi_n(i) \in \{i_1, i_2, \ldots, i_k\}\}$, the least DFR with SC decoder in the HES-PC is computed as $\text{DFR}_{min} = \Sigma_{i \in \mathcal{A}_{min}} Z(W_n^{(i)})$. Let $\mathcal{A}_{max} = \{i \in \mathcal{I}_n : \pi_n(i) \in \{i_{n-k+1}, i_{n-k+2}, \ldots, i_n\}\}$, the most DFR with SC decoder in the KEM-PC is acquired as $\text{DFR}_{max} = \Sigma_{i \in \mathcal{A}_{max}} Z(W_n^{(i)})$. Hence, the DFR in the proposed HES-PC with SC decoder can alter from DFR_{min} to DFR_{max} depending on k randomly sub-channels. Figure 3 illustrates the DFR_{min} and DFR_{max} for various polar codes of HES-PC. As seen in this figure, HES-PC with distinguished code lengths gives DFR values from DFR_{min} to DFR_{max}. Also, both DFR_{min} and DFR_{max} are reduced by aggrandizing the sizes of codes in the HES-PC instances. For instance, for polar code length $n = 2^{12}$, we compute $\mathcal{A}_{max} = \{i \in \mathcal{I}_{4096} : \pi_{4096}(i) \in \{i_{1312}, i_{1313}, \ldots, i_{3359}\}\}$ and $\mathcal{A}_{min} = \{i \in \mathcal{I}_{4096} : \pi_{4096}(i) \in \{i_{1271}, i_{1272}, \ldots, i_{3318}\}\}$. In fact, the upper bound on the DFR can change from $\text{DFR}_{min} = \sum_{i \in \mathcal{A}_{min}} Z(W_{4096}^{(i)}) \simeq 1.35 \times 10^{-7}$ to $\text{DFR}_{max} = \sum_{i \in \mathcal{A}_{max}} Z(W_{4096}^{(i)}) \simeq 9.39 \times 10^{-7}$.

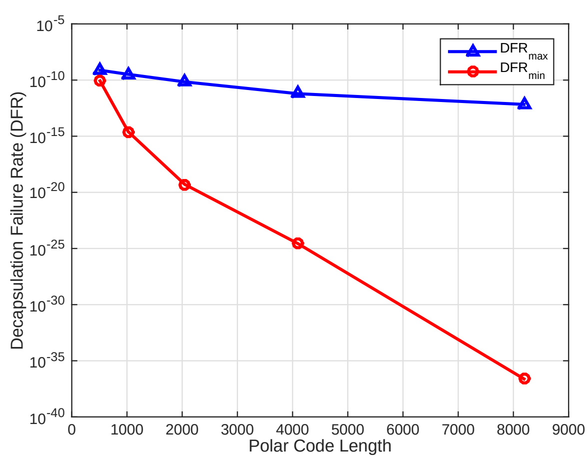

Figure 3: DFR_{min} and DFR_{max} for various code lengths of HES-PC instances.

6. Conclusion

This work describes a polar code-based hybrid encryption scheme, named as HES-PC, that makes secure and efficient communication between Alice and Bob. It is demonstrated that the proposed HES-PC has IND-CPA security. Also, by exploiting the Kobara-Imai conversion, it is shown that HES-PC is also IND-CCA2 and IND-CPA secure. Moreover, security analyses demonstrate that the proposed HES-PC is secure opposed to the conventional attacks such as brute-force attack, message-resend attack and Rao-Nam attack. The performance investigation results show that DFR of HES-PC for distinguished code lengths alters from DFR_{min} to DFR_{max} according to the picked out sub-channels. In fact, the security and performance of HES-PC appertain on different agents such as pseudorandomness of applied KDF, Hamming weight of perturbation vector, length and rate of used polar code. Hence, such agents should be picked out such that a desirable balance takes place between security and performance of HES-PC.

References

[1] K. Kurosawa, Y. Desmedt "A New Paradigm of Hybrid Encryption Scheme," *Proc. CRYPTO 2004*, Santa Barbara, USA (2004) 426-442.

[2] National Institute of Standards and Technology: Call for proposals (2016). http://csrc.nist.gov/groups/ST/post-quantum-crypto/documents/call-for-proposals-final-dec-2016.pdf.

[3] E. Arikan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Trans. Inf. Theory* 55 (2009) 3051-3073.

[4] R. Hooshmand, M. Koochak Shooshtari, and M. R. Aref, "PKC-PC: A variant of the McEliece Public Key Cryptosystem based on Polar Codes, " *IET Communications* 12 (2020) 1883-1893.

[5] R. Hooshmand, and M. R. Aref, “Efficient polar code-based physical layer encryption scheme, ” *IEEE Wireless Communications Letters* 6 (2017) 710-713.

[6] R. Hooshmand, and M. R. Aref, “Polar code-based secure channel coding scheme with small key size,” *IET Communications* 11 (2017) 2357-2361.

[7] R. Hooshmand, M. R. Aref, T. Eghlidos, “Secret key cryptosystem based on non-systematic polar codes,” *Wireless Personal Communications* 84 (2015) 1345-1373.

[8] R. Hooshmand, A. Jafari and G. Karamali, “Id-PC: An Identification Scheme based on Polar Codes,” *Information Security Journal: A Global Perspective* (2022) 1-15.

[9] R. Hooshmand, and M. Khoshfekr, “Key encapsulation mechanism based on polar codes,” *IET Communications* 16 (2022) 2438-2447.

[10] T. Johansson, and F. Jonsson, “On the complexity of some cryptographic problems based on the general decoding problem,” *IEEE Trans. Information Theory* 48 (2002) 2669-2678.

[11] A. Goli, S. H. Hassani, R. Urbanke, “Universal bounds on the scaling behavior of polar codes,” *IEEE Int. Symp. Inf. Theory* (2012) 1957-1961.

[12] E. Barker, J. Kelsey, “Recommendation for random number generation using deterministic random bit generators,” *DRAFT NIST Special Publi- cation 800-90A, Revision 1* (2014).

[13] T. A. Berson, “Failure of the McEliece public-key cryptosystem under message-resend and related-message attack,” *Proc. CRYPTO 97* (1997) 213-220.

[14] T.N.R. Rao, K.H. Nam, “Private key algebraic coded cryptosystem, ” *Proc. CRYPTO 87* (1987) 35-48.

[15] P. S. L. M. Barreto, S. Gueron, T. Guneysu, R. Misoczki, E. Persichetti, N. Sendrier, and J. P. Tillich, "CAKE: Code-based Algorithm for Key Encapsulation," *Proc. IMACC 2017*, Oxford, United Kingdom (2017) 207-226.

[16] K. Kobara, H. Imai, "Semantically secure McEliece public-key cryptosystems conversions for McEliece PKC," *Proc. Int. Workshop on Practice and Theory in PKCs*, Cheju Island, Korea (2001) 19-35.

[17] J. Stern, "A method for finding codewords of small weight," *Proc. Int. Colloquium on Coding Theory and Applications*, Toulon, France (1998) 106-113.

[18] R. Hooshmand, and M. R. Aref, "Public Key Cryptosystem Based on Low Density Lattice Codes," *Wireless Personal Communications*, 92 (2017) 1107-1123.