Security and Communication Networks

Theoretical Aspects of Cryptography and Their Applications for Data Protection in Emerging 5G Systems

Lead Guest Editor: Andrea Visconti
Guest Editors: Isaac Woungang and Sanjay Kumar Dhurandher

Theoretical Aspects of Cryptography and Their Applications for Data Protection in Emerging 5G Systems

Security and Communication Networks

Theoretical Aspects of Cryptography and Their Applications for Data Protection in Emerging 5G Systems

Lead Guest Editor: Andrea Visconti
Guest Editors: Isaac Woungang and Sanjay Kumar Dhurandher

Chief Editor

Associate Editors

Academic Editors

Contents

Hindawi
Security and Communication Networks
Volume 2020, Article ID 7137480, 12 pages
https://doi.org/10.1155/2020/7137480

Research Article

Decentralized Private Information Sharing Protocol on Social Networks

Shu-Chuan Chu,[1] Lili Chen,[1] Sachin Kumar,[2] Saru Kumari,[3] Joel J. P. C. Rodrigues,[4,5] and Chien-Ming Chen[1]

[1]*College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao, Shandong, China*
[2]*Department of Computer Science and Engineering, Ajay Kumar Garg Engineering College, Ghaziabad, India*
[3]*Department of Mathematics, Chaudhary Charan Singh University, Meerut, India*
[4]*Federal University of Piauí, 64049-550 Teresina, PI, Brazil*
[5]*Instituto de Telecomunicações, Lisboa 1049-001, Portugal*

Correspondence should be addressed to Chien-Ming Chen; chienmingchen@ieee.org

Received 5 November 2019; Accepted 6 January 2020; Published 5 June 2020

Guest Editor: Isaac Woungang

Social networks are becoming popular, with people sharing information with their friends on social networking sites. On many of these sites, shared information can be read by all of the friends; however, not all information is suitable for mass distribution and access. Although people can form communities on some sites, this feature is not yet available on all sites. Additionally, it is inconvenient to set receivers for a message when the target community is large. One characteristic of social networks is that people who know each other tend to form densely connected clusters, and connections between clusters are relatively rare. Based on this feature, community-finding algorithms have been proposed to detect communities on social networks. However, it is difficult to apply community-finding algorithms to distributed social networks. In this paper, we propose a distributed privacy control protocol for distributed social networks. By selecting only a small portion of people from a community, our protocol can transmit information to the target community.

1. Introduction

Social networks are increasing in popularity, and people are sharing information with their friends on social networking sites (SNS). Most of these sites treat all contacts equally by default. For example, if a person does not sort his/her friends into groups, subsequently all of the person's friends can view his/her messages posted on a wall. Even if SNS provide a grouping function, previous works have indicated that sorting friends is inconvenient [1, 2]. In the real world, individuals have distinct types of relationships with different people. The information a user wishes to share with a group of people may not be appropriate for people in other groups, even if they are all the user's friends.

Hence, many privacy protection mechanisms have been proposed [3–6]. These mechanisms, however, require users to set access rights for all their friends in advance. Although this provides accurate solutions for deciding who should have access to certain information, it is inconvenient for a user to manage them, especially when a user has many friends. People may not maintain all the groups that they join in real life on SNS. In addition, even though many social networking sites provide group settings, famous SNS such as Twitter do not have this feature yet. Jones and O'Neill [2] suggested providing group-based privacy using naturally organized groups, which reduces the burden of configurations.

A naturally organized group is a densely connected cluster on a social graph. People in real life tend to form groups. For example, you and your high school classmates form a group; you and your coworkers form another group; members of a club you belong to form yet another group. As indicated in previous studies, people who recognize each other in real life are likely to establish connections on SNS. Mayer and Puller [7] reported that only 0.4% of connections were merely online interactions; therefore, it is safe to assume that the connections on SNS between you and your friends and those between your friends and your friends' friends form clusters. While many ties exist inside a cluster, only a few ties exist across different clusters. These clusters become meaningful groups because connections in a cluster are established for the same reason.

Typical community-finding algorithms only function when a user have access to his/her own ego-network, which includes connections between the user and his/her friends (Level 1 friends) and between the user's friends and their friends (Level 2 friends). However, on many SNS such as Facebook, a user does not have access to other people's relationship paths. In theory, a user may acquire all his/her friends' connection by asking his/her friends to use a Facebook application written to collect data, which is almost impossible to achieve.

Another approach to protect a user's privacy is to establish a decentralized social network, that is, a social network in which a user only knows his/her direct connections. Although this is not yet popular, it has been discussed in Safebook [8] and Helloworld [9]. Furthermore, several studies have presented decentralized social network schemes [10]. In this type of social network, it is impossible to learn other people's connections in advance.

Herein, we first present previous studies regarding private information sharing in social networks; subsequently, we propose a new private-information sharing protocol used on decentralized social networks. Our protocol, which is based on secret sharing, utilizes characteristics of social networks. Our protocol exhibits the following properties. First, to utilize naturally organized groups, communities must be located using only information a user can acquire. We assume that the information a user can acquire is the list of her Level 1 friends. Next, this protocol does not leak the friendship connections of the source to any users. Furthermore, this protocol can be adapted to centralized social networks.

The remainder of this paper is organized as follows. We introduce the background and related studies in Section 2, present the model of our study in Section 3, introduce and analyze our protocol in Section 4, describe our experiments in Section 5, discuss the results in Section 6, and provide the conclusions in Section 7.

2. Background and Related Studies

2.1. Privacy Control on SNS. Security and privacy [11–14] are two important topics that are often discussed in various kinds of applications and environments [15–20]. The privacy problem on SNS has been reported in previous studies. Persona [3] combined attribute-based encryption with the traditional public-key approach to provide user-defined access control on SNS. *flybynight* [4] supported secure one-to-one and one-to-many communications on Facebook by applying RSA and El Gamal to encrypt and decrypt information. NOYB [5] protects private data by partitioning data into atoms and substituting these atoms with another user's atoms pseudorandomly. Lockr [21] provides access control on Flickr. However, these mechanisms require users to define the access ability for each of their friends in advance. By placing each of the user's friends into a predefined community, a user can share private information to only those in the target groups. They use cryptography to protect private information. This results in a complicated key exchange, and it will be difficult to revoke the keys when the connections on SNS are canceled.

2.2. Community-Finding Algorithms. Searching for communities on complex networks is a well-studied topic. Traditional methods based on graph partitioning, such as *Kernighan–Lin*'s algorithm [22], divide a graph into *n* clusters. Modern methods, such as *Newman–Girvan*'s algorithm [23], utilize "*modularity*" to define the stop criterion. Many community-finding algorithms [24, 25] based on modularity demonstrate good partition results when modularity is maximized. CONGA [26] improved the original *Newman–Girvan* algorithm so that overlapping groups could be detected. Other algorithms such as those in [27, 28] have been introduced to detect overlapping groups. The algorithms introduced above require a user to know the entire network data. However, it is infeasible for a user to obtain full network data on SNS or the WWW. Additionally, local community-finding algorithms have been proposed. Clauset [29] and others [30] proposed the local modularity method. Bagrow [31] proposed the "*outwardness*" method.

The algorithms mentioned above require users to know their Level 1 and Level 2 friends. However, on mobile networks, a user cannot easily obtain other people's connections. In addition, SNS such as Facebook restrict users from accessing other people's contextual information, which renders it difficult to apply these methods.

2.3. Group Communication. Applications on social networks are often related to group communication. Some have already utilized the naturally organized community. Grob et al. [1] conducted a survey and concluded that group communication occurred frequently, but grouping functions were rarely used. In their survey, only 16% of users used the built-in grouping functions on mobile phones. They implemented Cluestr and applied CONGA [26] to recommend friends within a community. Jones and O'Neill proposed using implicit communities that appeared on people's social graph for privacy control [2]. They used the SCAN algorithm [32] to detect communities. Li et al. [33] proposed a provably secure group key agreement scheme with privacy preservation for online social networks using extended chaotic maps.

3. Problem Statement and System Model

We model an online social network as a simple graph $G = (V, E)$, in which V is a set of users and E is a set of connections on that online social network. Furthermore, we model a real-life social network as a simple graph $G' = (V', E')$, where V' is a set of people and E' is a set of acquaintance links between each person. We assume that a bijection function $\mathscr{A}: V' \longrightarrow V$ exists. In other words, we ignore people who do not exist on the online social network. We model a real-life community $C' \subset V'$. We assume that a corresponding naturally formed community $C \in G$ exists for every community $C' \in G'$. That is, a bijection function $\mathscr{C}: C' \longrightarrow C$ exists.

We define a friend set $F(u)$ as a set of nodes $v \in G$, in which for each $v \in v$, $e = (u, v)$ exists.

3.1. Problem Statement. The goal of our protocol is to enable $u \in V$ to transmit a secret m to $C' = \{c_1, c_2, \ldots, c_n\} \subset V'$, where a corresponding $C \subset V$ exists. For each $c_i \in C'$, a corresponding node $c_i \in C$ exists. Transmitting m to the nodes in C is equivalent to transmitting it to C'.

3.2. Desired Properties. Our protocol exhibits the following properties.

3.2.1. Decentralized. Our protocol can be applied to decentralized social networks.

3.2.2. Privacy. Our protocol should protect all nodes' identities and link privacy. A node's u's link privacy is the knowledge of $e = (u, v)$, $v \in F(u)$. It is noteworthy that $F(u)$ means the friends of u. Its identity privacy is the knowledge of u's existence.

3.2.3. Robustness. Our protocol should adapt to constantly changing social networks. The set of users who receive the private information should conform to the current social network topology.

3.3. Adversary Model. Herein, we define a semihonest adversary model. In this model, a node g follows the protocol but may wish to discover $e = (u, v) \in E$, $u, v \in V$, where $u \neq g$, $v \neq g$, and g is not the sender on m.

For each $v \in V$, if the adversary g can identify any $e = (u, v) \in E$, $u \neq g$, then the link privacy of v is leaked.

For $v \in V$ who sends a secret in G, if an adversary g can identify the identity of v without acquiring the full secret, then v's identity is leaked. That is, a node should learn the source of a secret if and only if it receives the secret.

An adversary can be an intermediate node, receiver, or stranger that does not receive any tokens.

4. Protocol

In this section, we propose and analyze our privacy control protocol, known as the decentralized private information sharing protocol (DPISP). The DPISP allows a node to distribute private information on social networks to a group of nodes without setting community members in advance. Table 1 describes the notations.

Table 1: Notations.

Symbol	Statements
u_i	Node id
$F(u_i)$	Friends of u_i
U	A set of nodes
(k, n)-SS	(k, n) Secret sharing scheme
t_i	A token
m	Original message
d	TTL (time to live)

In the DPISP, nodes in G are divided into three parties:

(i) A source node sends a secret to $C \subset V$.

(ii) A receiver receives any part of the secret.

(iii) An intermediate node forwards any part of the secret to his/her friends. An intermediate node is also a receiver.

Although a source node u knows only $F(u)$, it can easily identify the role of each $v \in F(u)$ in G'. For example, a node knows who its classmates are and who its coworkers are. To send information to a particular community C on G, u can select representative nodes that belong to the corresponding community C' in G'. The nodes selected are the intermediate nodes.

The set of receivers is controlled by two parameters, n and k, along with the intermediate nodes designated by the source node. n is the number of intermediate nodes plus the source node, and k indicates the number of connections a receiver has between n and him/her to receive the full message. Refer to Figure 1 for an example: the diamond nodes indicate the source node. The four square nodes are the intermediate nodes $(n = 5)$. If we set $k = 4$, only the squares will have access to the full information. If we set $k = 3$, both the squares and circles will have access to the full message. If we set $k = 2$, even the triangles will have access to the message.

4.1. Protocol Overview. To send a private message to a community C, the source node u_s first divides the private message into n partial message; subsequently, u_s sends one token comprising partial information and a TTL tag $d = 1$ (the TTL tag is used to indicate if a token should be propagated further) along with an identity tag, n and k, to each of the intermediate nodes, keeping one for him/herself. The source node sends the token with $d = 0$ to all his/her friends. Tokens with the same identity tag indicate that they are the partial message of the same private message. The intermediate nodes save a copy of the tokens received from u_s, decrease the TTL by 1, and subsequently propagate the tokens to all their friends. Those who receive k or more tokens with the same identity tag can recover the private message.

However, in the case above, the source node's link privacy is leaked. If an intermediate node e receives a token directly from u_s and a token propagated by another intermediate node e' also sent from u, then e can acquire u_s's connections with other people. Assume that u_s sets $k = 3$;

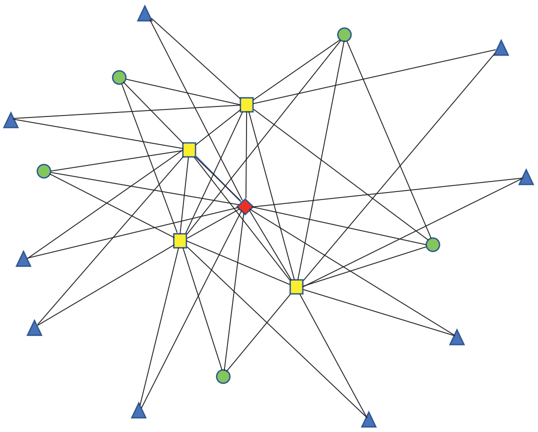

Figure 1: Diamond node is the source node, and square nodes are the intermediate nodes.

although e cannot recover the full message, he/she can still discover that e' has a connection with u_s. Because the token sent directly from u_s has $d = 1$, e instantly knows that the source node of this token is u_s. Furthermore, because u_s is the origin of the token propagated from e', the two tokens' identity tag will be the same; therefore, e knows that the token propagated from e' is also sent from u_s and realizes that a connection exists between u_s and e'.

To solve this privacy leakage problem, the identity of the source node cannot be identified by receivers, unless they can recover the private information.

4.2. DPISP. The DPISP is based on secret sharing. In secret sharing, a secret is divided to n parts; anyone who receives k of n parts can recover the secret, while those who receive fewer than k parts cannot recover the secret and learn anything from the information they have received. We applied Shamir's secret-sharing scheme [34] in our protocol. The DPISP contains two phases: the propagation and recovery phases. The detailed procedures of these two phases are shown in Figures 2 and 3.

4.2.1. Shamir's Secret Sharing. Shamir's secret sharing contains the following two schemes: the distribution scheme (SS) and the reconstruction scheme (SS^{-1}). Figure 4 shows the detailed functions.

A node runs $SS(n, k, s)$ to generate the shares from the secret s. The input n indicates the number of shares it creates, and k indicates the number of shares it has to recover the secret.

In $SS(n, k, s)$, a trusted dealer does the following:

(i) Randomly chooses $k - 1$ coefficients, denoted by $a_1, a_2, \ldots, a_{k-1}$

(ii) Constructs a polynomial $f(x) = s + a_1 x + \cdots + a_{k-1} x^{k-1}$

(iii) Computes shares s_i by evaluating $f(x)$ in n distinct points

```
1. $u_s$ chooses $n-1$ nodes $u_i \in C$, and $k$, then
   generate $n$ shares from $s|h$ by applying the $SS$
   , where $h = hash(s)$.
   For each share $(i, s_i)$, $0 \le i < n$, $u_s$ generates a
   corresponding token $t_i = ((i, s_i)|n|k|d)$,
   where $\forall t_i$, $i \neq 0 : d = 1$, and if $i = 0$, then $d = 0$.

2. $u_s \rightarrow U_{chosen} : t_i$, $1 \le i < n$, for each
           member in $U_{chosen}$, respectively
   $u_s \rightarrow F_{u_s} : t_0$

3. For each $u_i \in F_{u_i}$:
           $u_i$ decreases the tag $d$ of $t_i$ by 1
           if $d > 0$, $u_i \rightarrow F_{u_i} : t_i$
```

Figure 2: Propagation phase.

```
1. Put the tokens with the same $n$, $k$ into the same
   set, creating sets $m_{n,k}$.

2. Then put the tokens with the same $i$ in each set
   $m_{n,k}$ into the same subset, creating subsets $m_{n,k,i}$

Test all the combinations of choosing $k$ groups from
$m_{n,k}$

3. Choose $k$ subsets $m_{n,k,x_1}, m_{n,k,x_2}, \ldots, m_{n,k,x_k}$,
   where $0 \le x_i \le n$.
4. Tested all the combinations of choosing one token
   from each subset, $(y_1, s_{y_1}) \in m_{n,k,x_1}$,
   $(y_2, s_{y_2}) \in m_{n,k,x_2}, \ldots, (y_k, s_{y_k}) \in m_{n,k,x_k}$

5. Apply $SS^{-1}$ on $(y_1, s_{y_1}) \in m_{n,k,x_1}$, $1 \le i < k$,
   to recover $s'$.
   If successful, remove the chosen tokens from each
           group
           goto step 4
   If all the combinations of tokens are tested,
           goto step 3
   Finish if all possible combinations are tested
```

Figure 3: Recovery phase.

```
$SS(n, k, s)$ {
    Generates $k-1$ random numbers,
        denoted by $a_1, a_2, \ldots, a_{k-1}$
    Constructs $f(x) = s + a_1 x + a_2 x^2 + \ldots + a_{k-1} x^{k-1}$
    Computes $n$ pairs of $(x_i, y_i)$ by evaluating $f(x_i)$
    Output: $\{(x_{i_1}, y_{i_1}), (x_{i_2}, y_{i_2}), \ldots, (x_{i_n}, y_{i_n})\}$
}

$SS^{-1}(k, D)$ {
    If $D$ contains less than $k$ shares,
        Fail.
    Otherwise, use interpolation to recover $f'(x)$ by $D$
    Output: the constant $a_0$ of $f'(x)$
}
```

Figure 4: Shamir's secret sharing.

A node runs $SS^{-1}(k, D)$ to recover the secret s. The input k indicates the number of shares it has to recover the secret; the input D is a set of different shares denoted by

$\left\{x_{i_1}, x_{i_2}, \ldots, x_{i_j}\right\}$, where $0 < j < n$. In $\mathrm{SS}^{-1}(D)$, a node adapts Lagrange's interpolation with the set D to reconstruct the polynomial $f'(x)$. If D contains k or more different shares, $f'(x) = f(x)$; otherwise, $f'(x) \neq f(x)$ and no information is revealed from $f'(x)$.

4.2.2. Protocol Description. Figure 2 shows the propagation phase of our protocol. The source node u_s first selects $C \subset G$ that it wishes to share the private information m with. It selects $n - 1$ members that it recognizes in real life from that group as its intermediate nodes, where n is smaller than the group size. Next, u_s applies $\mathrm{SS}(n, k, s)$ to generate n shares (i, s_i), where $s = m|h$, $h = \mathrm{hash}(m)$, by evaluating the polynomial $f(x)$ in i and $0 \leq i < n$. For each s_i, u_s constructs the corresponding tokens $t_i = ((i, s_i) \mid n \mid k \mid d)$ with $d = 1$ for t_i, where $1 \leq i < n$ and $d = 0$ for t_0. The elements of the token are described as follows: (i, s_i) is the share that u_s distributes to u_i; n indicates the total number of different shares that u_s distributes; k represents the number of shares a node has to hold to reconstruct the message; and d is a TTL tag. It sends token $t_1, t_2, \ldots, t_n$ to the corresponding nodes that it selects earlier and sends t_0 to all its friends.

A node that receives any share first verifies d. If $d > 0$, the node decreases d by 1 and sends the token to all its friends. Additionally, the node maintains a copy of the token.

Figure 3 shows the recovery phase of the DPISP. To recover the private information from the tokens a node u receives, he runs the recovery phase of the DPISP. To decrease the calculation cost, u groups all the tokens by their (n, k), creating $|m_{n,k}|$ sets. Subsequently, he puts the tokens with the same i in each $m_{n,k}$ into the same subsets, creating subsets $m_{n,k,i}$, where $0 < i \leq n$. After grouping u's tokens, he runs SS^{-1} with all the combinations of tokens in each set $m_{n,k}$. In other words, u selects k subsets from $m_{n,k}$ and runs SS^{-1} for every possible combination of tokens among those k subsets. If a secret $s'|h$ is recovered successfully, u removes the tokens belonging to that secret. After testing all the possible combinations, u selects another k subset and repeats the same procedure until all the possible combinations of k subsets are tested.

To verify if s' is recovered successfully, a node calculates $\mathrm{hash}(s')$ and verifies if $h = \mathrm{hash}(s')$.

4.2.3. Analysis. First, we examine the privacy of our protocol. As we have described earlier, a node u on decentralized social networks only has knowledge of node set $F(u)$. By the DPISP, u recovers the information if and only if $|F(u) \cap \{u_s \cup I\}| \geq k$, where I is the set containing $n - 1$ intermediate nodes.

The types of privacy involved in this study are as follows:

(1) The Source Node's Privacy. Given that any v receives one or more tokens, v cannot distinguish its source u. In addition, v cannot distinguish if a connection exists between u and any $i \in V$ unless v can read the secret.

(2) The Intermediate Node's Privacy. Given that any v receives one or more tokens, v cannot distinguish if a connection exists between the intermediate node and any $i \in V$ unless v can recover the secret.

(3) The Receiver's Privacy. Any $v \in V$ cannot distinguish the receiver's identity and its connections to other nodes in V.

We discuss the privacy of the three roles. First, we show that DPISP protects the privacy of the source node by demonstrating that the identity of the source node is not revealed to those who cannot recover the secret s. Assume that a receiver cannot reconstruct s; as the elements of the tokens do not reveal the identity of the source node, the origin of the tokens cannot be distinguished. The only exception is that the intermediate nodes know the origin because $d = 1$; however, this is not a privacy leakage because the source node and intermediate nodes are already friends. In addition, knowing the information of one token does not reveal the source of other tokens.

Next, we demonstrate that the link privacy of the intermediate nodes is not revealed. Similar to the above, as the receivers do not know the origin of a token unless they can recover the private information, the identity of the source node is not revealed. Therefore, the receivers cannot acquire any knowledge regarding $e = (u_s, u_i)$ and, hence, the link privacy of the intermediate nodes is protected.

Finally, the privacy of the receivers is not revealed because the receivers do not provide any information to other nodes. The receivers can recover the private information by evaluating s_i using the reconstruction method of Shamir's secret-sharing protocol. With this information, they can identify shares that belong to the same u_s. Because they know the identity of the intermediate nodes that sent these shares to them and they know u_s because they can recover the private information, they know that connections exist between the intermediate nodes and the source node. However, we do not consider this a privacy leakage because we assume that nodes that can decode the message are in the same community as u_s and the intermediate nodes; therefore, the receiver should know that u_s and the intermediate nodes are Level 1 friends.

Next, we analyze the overhead of the DPISP. During the propagation phase of the DPISP, the source node sends tokens to all its friends; subsequently, all the intermediate nodes send tokens to their friends. Assume that the source node u_s has $|F_{u_s}|$ friends; among its friends, it selects $n - 1$ intermediate nodes, denoted by $u_1, \ldots, u_n$, and each of them has $|F_{u_i}|$ friends. The total number of tokens transmitted during the propagation phase is $|F_s| + |F_{u_1}| + \cdots + |F_{u_n}|$.

During the recovery phase of the DPISP, a node places the tokens into the subsets according to its (n, k) and i. Assume that $|m_{n,k}|$ different (n, k) pairs exist; therefore, it has to perform a maximum of SS^{-1} for $|m_{n,k}| \cdot \Sigma \binom{n_i}{k_i} \cdot |m_{n_i,k_i,x_{i_1}}| \cdot |m_{n_i,k_i,x_{i_2}}| \cdot \cdots \cdot |m_{n_i,k_i,x_{i_{k_i}}}|$ times to recover any secrets. Although a node has to perform SS^{-1} times, the time cost is not as large as one might imagine.

4.3. Semidecentralized Protocol. The most pressing problem of the DPISP is that nodes may spend a significant amount of time recovering secrets if they receive many tokens. To avoid an exhaustive search, we propose a semidecentralized information sharing protocol, that is, the SDPISP.

The SDPISP utilizes a server to log tokens. The server provides two functions: register (R, k) and query (R), in which R is a set of integers, and k is the threshold. A source node registers a group of numbers to the server by calling the register function. The server records these numbers into a single entry via an event_{id} in its database. Each entry contains the threshold k. A receiver calls the query function to verify if any set of tokens that is valid for recovery exists.

To send private information using the SDPISP, u_s decides a target community and divides the secret $s = m|h$, where m is the private information and $h = \text{hash}(m)$, into n shares by applying $\text{SS}(n, k, s)$. Subsequently, instead of generating tokens without any identity tags, it generates tokens $t_i = (r_i \,|\, (i, s_i) \,|\, n \,|\, k \,|\, d)$, with $1 \le i \le n$, where r_i is a random number. Next, u_s calls register (R, k), $R = \{r_i \,|\, 1 \le i \le n\}$, to send these random numbers to the server.

To recover secrets, a receiver calls query (R) and inputs all the random numbers he/she received. The server returns the sets of random numbers that get recorded under the same event_{id} if the receiver holds k or more tokens.

Figure 5 illustrates the concept of the SDPISP. A and B wish to send some data to their friends. They set $n = 3$ and $k = 2$. They first create tokens with random numbers $1, 2, 3$ and $4, 5, 6$, respectively. Subsequently, they register these numbers to a server and send the tokens to the intermediate users. Those who receive any tokens, for example, C who receives tokens with $r_i = 1, 2, 3, 4$, call the query function to the server; subsequently, the server returns $\{1, 2, 3\}$ to C. The server does not return 4 because C only obtains one token from B, and C cannot recover the data only by token 4. Take D as another example: D receives tokens $2, 3, 5, 6$ and calls the query function; the server returns $2, 3$ and $5, 6$ to D. Subsequently D knows that he/she can recover two different sets of data from them.

4.3.1. Analysis. Using a server, receivers are not required to calculate all possible token combinations. By the SDPISP, they ask a server if any set of tokens that belongs to the same secret exists.

We examine the privacy of this semidecentralized protocol and ignore the chance of two random numbers colliding. In the SDPISP, the identity and link privacy of the participants are not leaked to each other. Additionally, the link privacy is not leaked to the server. If a node u registers some random numbers to the server and a node v queries the server with any random number that is registered by u, then the server will not know whether v is a Level 1 friend of u or a Level 2 friend of u.

5. Experiments

Assume that a community is an isolated, fully connected network, where all people belonging to the community are connected to each other, while no connections exist between people in different categories. In this case, it is sufficient for a user to set $n = 2$ and $k = 2$. All the members in the community recover the information, but those who do not belong to the community will not receive enough shares to reconstruct the data.

However, on real social networks, two cases can occur. First, two users in the same community may not have a connection with each other. Second, one or more users may have connections to the people who are not in the same community as them. To improve the accuracy, we conducted experiments to obtain adequate settings for n and k for communities with different sizes and clustering coefficients.

5.1. Data Collection. Owing to the lack of ground truth, which is the information of the communities each user belongs to, existing social network graphs such as those in [35, 36] cannot be applied to our experiments. Many previous studies collected data from Facebook. However, querying other people's contextual information is not allowed by the Facebook API unless they agree to provide the information. The only information we can easily acquire is the participants' Level 1 friends. To collect the Level 2 friends, we can develop a Facebook application to collect the information and ask the participants' friends to use it; however, it is unrealistic to expect all of them to use it. Because we require both the Level 1 and Level 2 connections of the participants to perform the experiments, we cannot collect data from Facebook. Therefore, we collected data from Plurk.

Plurk is a famous microblog in Taiwan. According to Alexa [37], on April 4, 2011, 41.5% of Plurk traffic was from Taiwan, where it ranked 27th, as well as 1297th worldwide. The Plurk API allows us to collect other users' data provided that the information is publicly available. Therefore, we asked students at both HIT.SZ and IIIRC to provide their friendship connections on Plurk.

In our experiments, we collected the friendship graph of eight students from HIT.SZ and two students from IIIRC whose *Karma* were all higher than 60—Karma is a value that evaluates the liveness of a user on Plurk. We extracted the links between these participants and their Level 1 and Level 2 friends. Subsequently, we placed their friends into one or more communities that were defined by the participants. For example, Table 2 shows the list of communities given by participant A. He/she defined 4 communities to represent the social network and each of his/her friends can belong to 1 to 4 communities. Similarly, we collected 42 communities from them, in which the minimum community size was 3 and the maximum community size was 61; we denote the community size as β in the following sections. Furthermore, we calculated the clustering coefficient (γ) of each community, which measures the degree of which users in a community tend to cluster together. Equation (1) shows the definition of γ, where l and $l_{\max}$ indicate the actual and maximum number of links between each user in a community, respectively. The maximum number of links between each user in a community is $\binom{n}{2}$, where n is the

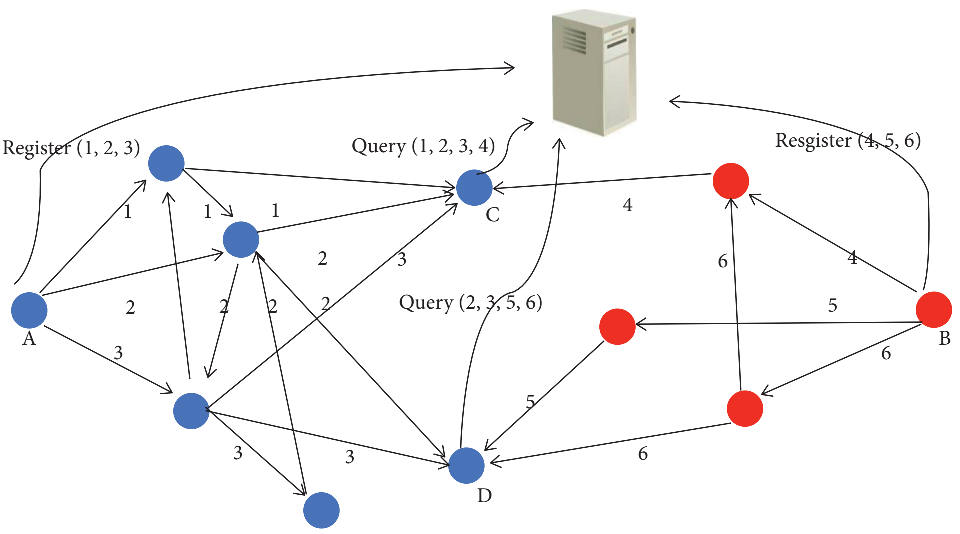

Figure 5: Example of using SDPISP.

Table 2: Communities given by participant A.

Group name	Size
CS13	46
Club	13
Labmate	55
High school	5

community size. Table 3 shows the detailed data that we collected from Plurk.

$$\gamma = \frac{l}{l_{\max}}. \tag{1}$$

After we have collected data from the participants, we performed experiments and recorded users who were not direct friends of the sender but had recovered the token. We sorted the list by the number of appearances each user emerged in the community during the experiments; subsequently, we asked the participants to confirm whether those who appeared on the lists were members of that community. Most participants indicated that they could not accurately determine whether a user on the list belonged to any of their defined communities. However, among the data we collected, we were confident that two participants could correctly identify their Level 2 friends who belonged to one of their communities.

Because it is easier for a user to select community members manually when the community is small, we ignored communities smaller than nine in our subsequent experiments. Hence, our test cases were formed by 31 communities that contain 9 to 61 members.

We only considered the neighbors of participants in a community because the participants cannot accurately tag those who are not their neighbors in the social network. Therefore, in our subsequent experiments, the results involve only the neighbors of each participant.

5.2. Accuracy of the DPISP.

A feature of the DPISP is that the secret-sharing parameter (n, k) can be dynamically adjusted. That is, people may decide if they want more or fewer users to receive the token. The secret-sharing parameter n indicates the number of shares a user has to send to the intermediate users, and k means the number of shares a user has to receive to recover the message. We measured the results of the DPISP by calculating the precision and recall.

The parameters used in this study are defined as follows:

(i) True Positive (TP): users retrieved by the DPISP who are in the community defined by the participant.

(ii) False Positive (FP): users retrieved by the DPISP who are not in the community defined by the participant.

(iii) False Negative (FN): users who are in the community defined by the participant but were not retrieved by the DPISP

(iv) Precision: fraction of users retrieved by the DPISP that belonged to the community tagged by the participant.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{2}$$

(v) Recall: fraction of users retrieved by the DPISP that belonged to the community tagged by the participant.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{3}$$

In our experiments, we divided the test cases based on their sizes (β) and clustering coefficients. For each test case, we tested them by setting n to 3 to 8. For each n, we produced

Table 3: Statistics of the collected communities.

Groups	Min size	Max size	Avg. size	St. dev. size	Min γ	Max γ	Avg. γ	St. dev. γ
42	3	61	19.5476	15.7824	0.1429	1.0	0.5752	0.2095

$(\beta, n-1)$ possible intermediate users sets. Subsequently, we tested the results for k ranging from 2 to n for each intermediate user sets.

An intermediate user set is the $n-1$ intermediate user selected by the source user. Owing to high time cost, if the number of possible combinations of $(\beta, n-1)$ is smaller than or equal to 5000, then we test all the possible intermediate user sets; if the number of possible combinations is larger than 5000, then we randomly select 5000 possible intermediate user sets to test.

5.2.1. Relation between (n, k) and Group Size. We present the result by dividing the test cases into three categories according to their sizes. Categories A, B, and C contain the test cases with community sizes between 9 and 16, between 17 and 32, and larger than 32, respectively. The clustering coefficients of these categories are 0.5878, 0.5126, and 0.5394, respectively.

Figures 6 and 7 show the average precision and recall of each category: in all three categories, when k is fixed, the precision decreases and the recall increases as n increases. This is because if more intermediate users exist, more people will receive the shares, thereby increasing the probability of people who are/are not community members that recover the token.

According to these data, while k increases, recall decreases quickly. For example, in category B, recall is 0.7278 for $k = 2$ and decreases to 0.1646 for $k = 5$ when $n = 5$. This is because users must receive more shares to recover the token; hence, the number of users who can recover the token decreases. Consequently, we recommend that users select a small k. Meanwhile, the precision decreases and the recall increases slightly when n increases. The precision is 0.73 for $n = 6$ and decreases to 0.7008 for $n = 8$ when $k = 3$; the recall increases from 0.57 to 0.66 for the same (n, k) pairs. This implies that users do not have to select a large n to obtain the best result. Instead, they can select a smaller n and the result will still be acceptable.

Additionally, we observed that the average precision increased with the community size. In our opinion, this was caused by an overlap in these communities. For instance, a user's "*good friend*" group might be a subset of his/her "*classmates*" group. Figure 8 illustrates an example of overlapping communities. Suppose a user wishes to send a message to his/her "*good friends*"; therefore, she sends shares to the intermediate users among her "*good friends*." However, some of her "*classmates*" can recover the token because the network is densely connected, thereby reducing the precision.

5.2.2. Relation between (n, k) and the Clustering Coefficient. We divided the test cases into four categories according to their γ, where category A contains test cases with $\gamma < 0.4$, category B contains test cases with $0.4 \leq \gamma < 0.5$, category C contains test cases with $0.5 \leq \gamma < 0.6$, and category D contains test cases with $\gamma \geq 0.6$.

Figure 9 shows the average recalls of the four categories. We observed that the recalls reduced quickly while k increased for communities with $\gamma < 0.6$ but decreased slightly while k increased for communities with $\gamma \geq 0.6$.

Even though the users may not know the clustering coefficient of their desired communities in advance, they can estimate whether the community is densely or loosely connected. For example, if A wishes to send a message to his/her laboratory, he/she can assume that the members of this community are familiar with each other; therefore, the community is highly clustered, and he/she can set k to 4 or 5 to minimize the chances of outliers recovering the token. Meanwhile, if A wishes to send a message to his/her friends in his department, he/she should set k to 2 or 3 to maximize the chances of the members of the department to recover the token.

5.2.3. Token Transmitted during DPISP. The number of tokens that the source and intermediate users must transmit must be equal to that of their friends. Figure 10 shows the average number of tokens transmitted during the protocol; the total number of tokens transmitted during the protocol increases with n.

For each n, regardless of the value of k, the number of tokens transmitted should be the same. However, as shown by the results, the number of tokens differs when k changes. This is because not every round appears during the experiments when a user recovers the token. Occasionally, no user can recover the token because none has received enough shares. We only counted the rounds where one user at the least recovered the message in the experiments.

5.3. Success Rate of the DPISP. Transmitting tokens through different intermediate user sets causes different groups of users to receive the tokens. While some intermediate user sets yield good results, occasionally no user can recover the information sent by the source user.

Herein, we present the success rate of the DPISP. For each test case, we measured the results for a maximum of 5000 rounds. We considered a test round successful if one or more users could recover the token. The results shown in Section 5.2 only incorporated the successful rounds.

We measured the success rate of our protocol by the following formula:

$$p_{(n,k)} = \frac{\sum sr_{(n,k),i}}{\sum r_{(n,k),i}}, \tag{4}$$

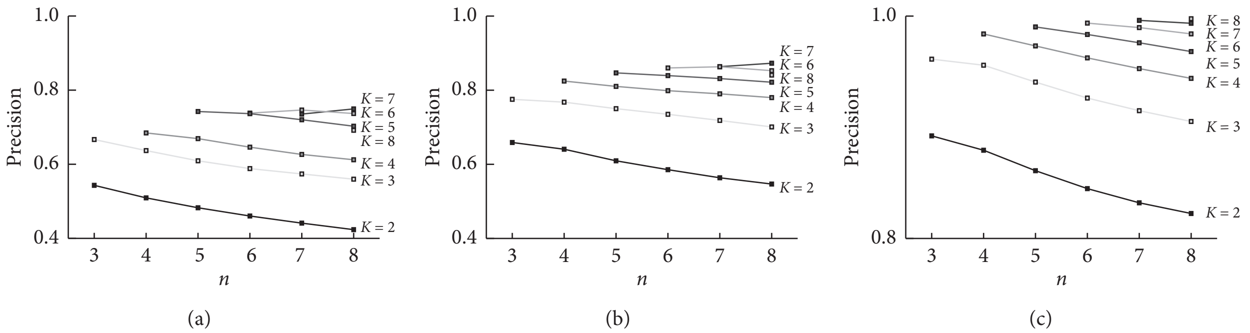

Figure 6: Relation of precision between (n, k) and community size: (a) contains communities with a size between 9 and 15; (b) contains communities with a size between 16 and 31; (c) contains communities with a size larger than 32.

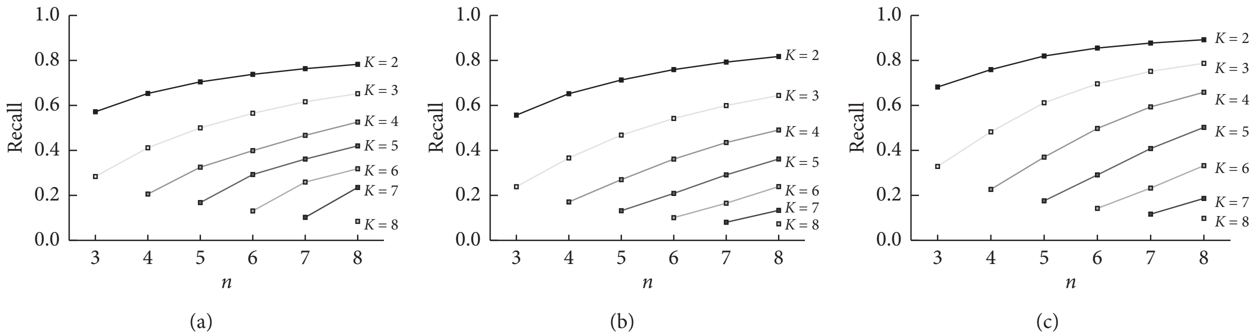

Figure 7: Relation of recall between (n, k) and community size: (a) contains communities with a size between 9 and 15; (b) contains communities with a size between 16 and 31; (c) contains communities with a size larger than 32.

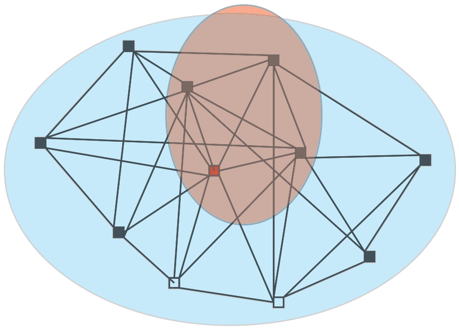

Figure 8: Two overlapped communities: the red area represents a user's "*good friends*," and the blue area represents his/her "*classmates*."

where $p_{(n,k)}$ indicates the average success rate tested in (n, k) for all test cases, $sr_{(n,k),i}$ indicates the number of successful rounds tested in (n, k) for the ith test case, and $sr_{(n,k),i}$ indicates the number of total rounds tested in (n, k) for the ith test case.

Figure 11 depicts the success rates, average number of failure rounds, and average number of total rounds of each (n, k) calculated by the test cases. Although the success rate decreases when k increases, it is near 100% when k is small, which means that even if a user selects intermediate users randomly, the information can still be propagated to someone.

5.4. Choosing Intermediate Users. In the DPISP, a user selects intermediate users from those who belong to that community. If, unfortunately, he/she selects "bad" intermediate users (i.e., users who have only a few links to the community members), the precision will be low, or the recall will be high.

To help users find "good" intermediate users, a user can send a link to all of his/her friends and ask them to register on that link to prove in advance that sufficient connections exist between them. Hence, a user u_s first generates $|F_{u_s}|$ shares from the link and distributes these shares individually to each of his/her friends. Anyone who receives the shares recovers the link by applying the reconstruction method of secret sharing. Subsequently, he/she registers him/herself on that link.

Figure 12 shows the results of selecting those who have many connections with the community members. As shown in previous data, setting $k \geq 4$ yields a low recall and only a few people can recover the private information. The experimental result shows an example of selecting good intermediate users. If a user selects 6 intermediate users who are tightly connected with the community members, even if he/she sets $k = 5$, the recall will be approximately 0.9. Conversely, if he/she chooses intermediate users randomly, the recall will only be 0.6. According to previous results, the precision increases with k; therefore, selecting good intermediate users yields better precisions and recalls.

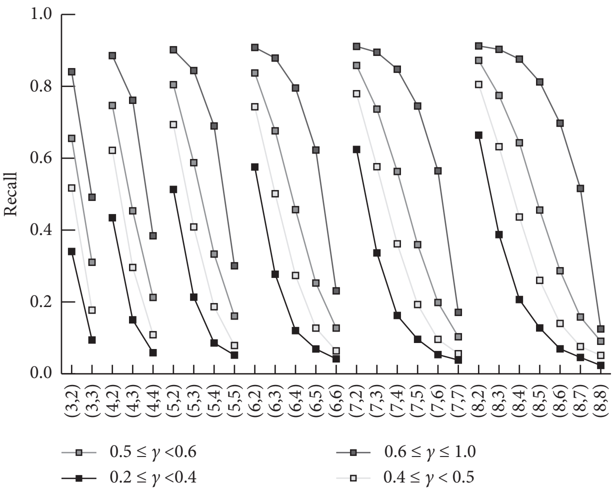

Figure 9: Recall for each (n, k) of each category based on γ.

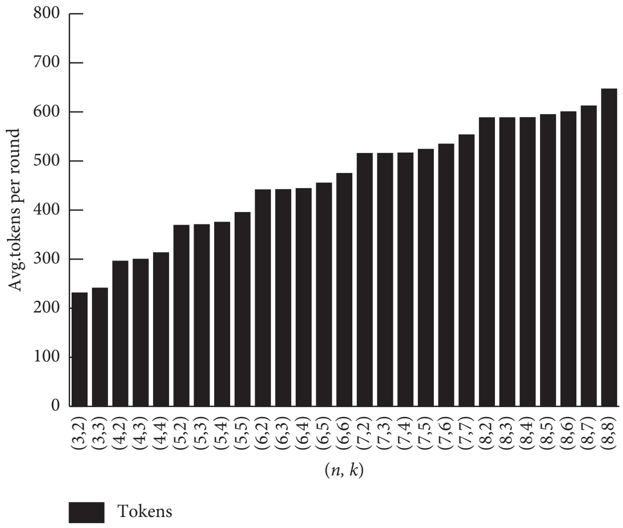

Figure 10: Average tokens transmitted during the protocol for each (n, k).

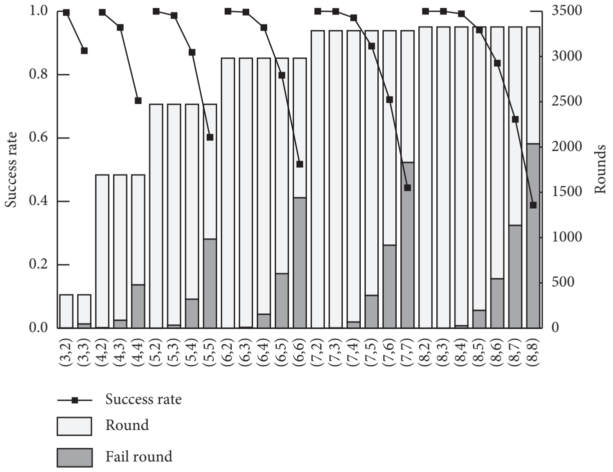

Figure 11: Average success rate, average number of failure rounds, and average number of total rounds for each (n, k).

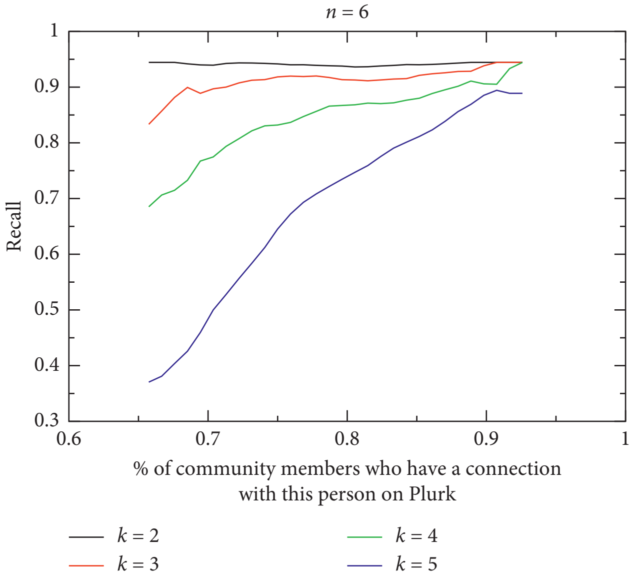

Figure 12: Choosing "good" intermediate users yielding a higher recall.

5.5. Computation Cost of the Recovery Phase. In this section, we discuss the cost of the DPISP's recovery phase. At first glance, it seems that a user must spend a large amount of time to reconstruct the secrets. In theory, a user must perform $|m_{n,k}| \cdot \Sigma \binom{n_i}{k_i} \cdot |m_{n_i,k_i,x_{i_1}}| \cdot |m_{n_i,k_i,x_{i_2}}| \cdot \cdots \cdot |m_{n_i,k_i,x_{i_{k_i}}}|$ times of SS^{-1} to construct all possible secrets, where $|m_{n,k}|$ is the number of tokens with different (n, k) pairs.

To examine the efficiency of the DPISP, we analyzed the number of SS^{-1} a user has to perform with respect to (n, k) and the number of users who have sent secrets. Furthermore, we analyzed the time required by SS^{-1}. Simulations were performed on a PC with a 4-core 3.2 GHz Intel CPU and 4 Gb of RAM. We implemented the secret-sharing functions using the C# SecretSharp library [38] on Microsoft Visual Studio 2010.

In our experiments, we simulated q users simultaneously and sent their secrets with the (n, k) settings in the ranges of $(3, 2)$ to $(8, 2)$ and $(6, 3)$ to $(8, 3)$. In each case, a user can receive a maximum of qn tokens decentralized in n bins, and each bin contains a maximum of q tokens. Therefore, a user has to perform a maximum of $q^k \cdot \binom{n}{k}$ secret sharing to recover all the secrets. Table 4 shows the results of the simulations, with $q = 5, 10, 15, 20$. The size of the coefficients of a polynomial is 1024 bits. In other words, the maximum secret size is 128 bytes. As shown by the results, a user can perform ≈ 6500 times of SS^{-1} per second. If a user sets $(8, 3)$, 448,000 possible combinations exist, which requires slightly more than one minute to recover all the secrets.

6. Discussion

In this section, we discuss the causes of the inaccuracy of the DPISP and a method to improve the efficiency of the DPISP recovery phase. Additionally, we discuss the method of sharing large data.

Table 4: Time cost for the recovery phase: *run*: the maximum possible combinations of tokens a user must test, *time*: the time to test all combinations in milliseconds.

(n, k)	5		10		15		20	
	Run	Time	Run	Time	Run	Time	Run	Time
$(3, 2)$	75	10	300	40	675	90	1200	160
$(4, 2)$	150	20	600	80	1350	180	2400	317
$(5, 2)$	250	33	1000	133	2250	300	4000	527
$(6, 2)$	375	50	1500	199	3375	449	6000	790
$(7, 2)$	525	70	2100	278	4725	630	8400	1110
$(8, 2)$	700	97	2800	372	6300	840	11200	1500
$(6, 3)$	2500	455	20000	3550	67500	12000	160000	28000
$(7, 3)$	4375	785	35000	6250	118125	21000	280000	49200
$(8, 3)$	7000	1280	56000	9970	189000	34600	448000	79140

6.1. Improving DPISP Accuracy.

We discuss possible reasons for the inaccuracy of the protocol in this section.

First, inaccuracy can be caused by users who do not publish their friendship connections. Many social networks allow users to set whether their information can be accessed by other people. If any of the intermediate users do not share their connections during the experiments, the share sent to him/her cannot be further transmitted to other users, thereby decreasing the probability of users related to the corresponding intermediate user recovering the token.

Next, inaccuracy may be caused by robots. Many "*robots*" exist on Plurk. These robots were developed to perform automated message broadcasting or to be an "*oracle*," which allows users to ask questions and provide answers. Almost all users on Plurk have connections with the default account "plurk buddy" and many other robots. Therefore, although these robots are not in user-defined communities, they have a high chance of recovering the token.

This problem can be mitigated by creating a list of ignored users. When we execute the protocol, we can ignore users who are not normal users.

6.2. Sharing Large Data.

Using Shamir's secret sharing, the maximum size of a message is restricted by the coefficient size of a polynomial. For example, if the coefficient size is 1024 bits, the maximum message size is 128 bytes. If the size of a private information is larger than 128 bytes, a user has to partition the message into 128-byte blocks and a receiver has to spend more time recovering the private information.

The constant of the polynomial is insufficient for placing large data, that is, file, photographs, and so forth. Researchers have proposed several multi-secret-sharing schemes [39–42]. For example, Yang et al. [39] proposed a scheme that shares p secrets instead of one secret in a polynomial; however, the threshold of that polynomial is extremely high according to their experiments. The DPISP performs well only when k is small. Hence, it is difficult to apply these algorithms unless a user selects a large k.

Instead of sharing data directly, a user can encrypt data with a session key; furthermore, they can share the key and a path to the data with his/her friends using the DPISP.

7. Conclusion

In this paper, we present DPISP, an information sharing protocol used on social networks. On decentralized social networks or on SNS like Facebook, where users cannot directly access other people's contextual information , our method provides a more realistic way to implement group communication functions using naturally organized communities. We also demonstrate that our method protects users' link privacy. In addition, DPISP runs without using any key or passwords, so it adapts to changes of the networks. One does not have to redistribute keys to all of her friends when she adds or remove friends.

By tuning the parameters (n, k), an information can be sent to different subsets of community members. Our results show that among the users who can recover secrets, about 60% to 80% belong to the target communities; about 50% to 70% of a community can recover the secret correctly.

Data Availability

The experimental data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by national funding from Fundação para a Ciência e a Tecnologia (FCT) through the UID/EEA/50008/2019 Project and by the Brazilian National Council for Research and Development (CNPq) via Grant no. 309335/2017-5.

References

[1] R. Grob, M. Kuhn, R. Wattenhofer, and M. Wirz, "Cluestr: mobile social networking for enhanced group communication," in *Proceedings of the ACM International Conference on Supporting Group Work*, pp. 81–90, Sanibel Island, FA, USA, May 2009.

[2] S. Jones and E. O'Neill, "Feasibility of structural network clustering for group-based privacy control in social networks," in *Proceedings of the 6th Symposium on Usable Privacy and Security*, Redmond, WA, USA, July 2010.

[3] R. Baden, A. Bender, N. Spring, B. Bhattacharjee, and D. Starin, "Persona: an online social network with user-defined privacy," *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 4, pp. 135–146, 2009.

[4] M. Lucas and N. Borisov, "Flybynight: mitigating the privacy risks of social networking," in *Proceedings of the 7th ACM Workshop on Privacy in the Electronic Society, 2008*, pp. 1–8, Alexandria, VA, USA, October 2008.

[5] S. Guha, K. Tang, and P. Francis, "NOYB: privacy in online social networks," in *Proceedings of the 1st Workshop on Online Social Networks, 2008*, pp. 49–54, Seattle, WA, USA, August 2008.

[6] A. K. Das, M. Wazid, N. Kumar, A. V. Vasilakos, and J. J. P. C. Rodrigues, "Biometrics-based privacy-preserving user authentication scheme for cloud-based industrial

internet of things deployment," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 4900–4913, 2018.

[7] A. Mayer and S. Puller, "The old boy (and girl) network: social network formation on university campuses," *Journal of Public Economics*, vol. 92, no. 1-2, pp. 329–347, 2008.

[8] L. Cutillo, R. Molva, and T. Strufe, "Safebook: feasibility of transitive cooperation for privacy on a decentralized social network," in *Proceedings of the IEEE International Symposium on World of Wireless, Mobile and Multimedia Networks & Workshops, 2009*, pp. 1–6, Kos, Greece, June 2009.

[9] M. Ackermann, K. Hymon, B. Ludwig, and K. Wilhelm, "Helloworld: an open source, distributed and secure social network," in *Proceedings of the W3C Workshop on the Future of Social Networking*, Barcelona, Spain, January 2009.

[10] L. Aiello and G. Ruffo, "Secure and flexible framework for decentralized social network services," in *Proceedings of the 8th IEEE International Conference on Pervasive Computing and Communications Workshops, 2010*, pp. 594–599, Mannheim, Germany, March 2010.

[11] C.-M. Chen, Y. Huang, K.-H. Wang, S. Kumari, and M.-E. Wu, "A secure authenticated and key exchange scheme for fog computing," *Enterprise Information Systems*, pp. 1–16, 2020.

[12] K. Renuka, S. Kumar, S. Kumari, and C.-M. Chen, "Cryptanalysis and improvement of a privacy-preserving three-factor authentication protocol for wireless sensor networks," *Sensors*, vol. 19, no. 21, p. 4625, 2019.

[13] T.-Y. Wu, C.-M. Chen, K.-H. Wang, C. Meng, and E. K. Wang, "A provably secure certificateless public key encryption with keyword search," *Journal of the Chinese Institute of Engineers*, vol. 42, no. 1, pp. 20–28, 2019.

[14] H. Xiong, Y. Zhao, L. Peng, H. Zhang, and K.-H. Yeh, "Partially policy-hidden attribute-based broadcast encryption with secure delegation in edge computing," *Future Generation Computer Systems*, vol. 97, pp. 453–461, 2019.

[15] C.-M. Chen, B. Xiang, Y. Liu, and K.-H. Wang, "A secure authentication protocol for internet of vehicles," *IEEE Access*, vol. 7, pp. 12047–12057, 2019.

[16] C.-M. Chen, K.-H. Wang, K.-H. Yeh, B. Xiang, and T.-Y. Wu, "Attacks and solutions on a three-party password-based authenticated key exchange protocol for wireless communications," *Journal of Ambient Intelligence and Humanized Computing*, vol. 10, no. 8, pp. 3133–3142, 2019.

[17] J. M.-T. Wu, J. C.-W. Lin, and A. Tamrakar, "High-utility itemset mining with effective pruning strategies," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 13, no. 6, p. 58, 2019.

[18] J.-S. Pan, C.-Y. Lee, A. Sghaier, M. Zeghid, and J. Xie, "Novel systolization of subquadratic space complexity multipliers based on toeplitz matrix-vector product approach," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 27, no. 7, pp. 1614–1622, 2019.

[19] W. Gan, J. C.-W. Lin, P. Fournier-Viger, H.-C. Chao, V. S. Tseng, and P. S. Yu, "A survey of utility-oriented pattern mining," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–20, 2019.

[20] S.-C. Chu, X. Xue, J.-S. Pan, and X. Wu, "Optimizing ontology alignment in vector space," *Journal of Internet Technology*, vol. 21, no. 1, pp. 15–22, 2020.

[21] A. Tootoonchian, K. Gollu, S. Saroiu, Y. Ganjali, and A. Wolman, "Lockr: social access control for web 2.0," in *Proceedings of the 1st Workshop on Online Social Networks, 2008*, pp. 43–48, Seattle, WA, USA, August 2008.

[22] B. W. Kernighan and S. Lin, "An efficient heuristic procedure for partitioning graphs," *Bell System Technical Journal*, vol. 49, no. 2, pp. 291–307, 1970.

[23] M. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Physical Review E*, vol. 69, no. 2, Article ID 026113, 2004.

[24] E. A. Leicht and M. E. J. Newman, "Community structure in directed networks," *Physical Review Letters*, vol. 100, no. 11, Article ID 118703, 2008.

[25] U. Brandes, D. Delling, M. Gaertler et al., "On finding graph clusterings with maximum modularity," in *Graph-Theoretic Concepts in Computer Science*, pp. 121–132, Springer, Berlin, Germany, 2007.

[26] S. Gregory, "An algorithm to find overlapping community structure in networks," in *Proceedings of the International Conference on Knowledge Discovery in Databases, 2007*, pp. 91–102, Warsaw, Poland, September 2007.

[27] B. Karrer, E. Levina, and M. Newman, "Robustness of community structure in networks," *Physical Review E*, vol. 77, no. 4, Article ID 046119, 2008.

[28] N. Du, B. Wu, X. Pei, B. Wang, and L. Xu, "Community detection in large-scale social networks," in *Proceedings of the 9th WebKDD and 1st SNA-KDD Workshop on Web Mining and Social Network Analysis, 2007*, pp. 16–25, San Jose, CA, USA, August 2007.

[29] A. Clauset, "Finding local community structure in networks," *Physical Review E*, vol. 72, no. 2, Article ID 026132, 2005.

[30] S. Muff, F. Rao, and A. Caflisch, "Local modularity measure for network clusterizations," *Physical Review E*, vol. 72, no. 5, Article ID 056107, 2005.

[31] J. Bagrow, "Evaluating local community methods in networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 5, Article ID P05001, 2008.

[32] X. Xu, N. Yuruk, Z. Feng, and T. Schweiger, "Scan: a structural clustering algorithm for networks," in *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2007*, pp. 824–833, San Jose, CA, USA, August 2007.

[33] C.-T. Li, T.-Y. Wu, and C.-M. Chen, "A provably secure group key agreement scheme with privacy preservation for online social networks using extended chaotic maps," *IEEE Access*, vol. 6, pp. 66742–66753, 2018.

[34] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612-613, 1979.

[35] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel, and B. Bhattacharjee, "Measurement and analysis of online social networks," in *Proceedings of the 5th ACM/USENIX Conference on Internet Measurement, 2007*, San Diego, CA, USA, October 2007.

[36] K. Lewis, J. Kaufman, M. Gonzalez, A. Wimmer, and N. Christakis, "Tastes, ties, and time: a new social network dataset using facebook.com," *Social Networks*, vol. 30, no. 4, pp. 330–342, 2008.

[37] Alexa, http://www.alexa.com/siteinfo/plurk.com.

[38] SecretSharp, http://sourceforge.net/projects/secretsharp/.

[39] C.-C. Yang, T.-Y. Chang, and M.-S. Hwang, "A (t,n) multi-secret sharing scheme," *Applied Mathematics and Computation*, vol. 151, no. 2, pp. 483–490, 2004.

[40] J. Shao and Z. Cao, "A new efficient (t,n) verifiable multi-secret sharing (VMSS) based on YCH scheme," *Applied Mathematics and Computation*, vol. 168, no. 1, pp. 135–140, 2005.

[41] L.-J. Pang and Y.-M. Wang, "A new (t,n) multi-secret sharing scheme based on Shamir's secret sharing," *Applied Mathematics and Computation*, vol. 167, no. 2, pp. 840–848, 2005.

[42] H. Chien, J. Jan, and Y. Tseng, "A practical (t,n) multi-secret sharing scheme," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 83, no. 12, pp. 2762–2765, 2000.

Hindawi
Security and Communication Networks
Volume 2020, Article ID 4180139, 6 pages
https://doi.org/10.1155/2020/4180139

Research Article

Minimizing Key Materials: The Even–Mansour Cipher Revisited and Its Application to Lightweight Authenticated Encryption

Ping Zhang[1] **and Qian Yuan**[2]

[1]*School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing 210023, China*
[2]*School of Economics and Management, Southeast University, Nanjing 211189, China*

Correspondence should be addressed to Ping Zhang; zhgp@njupt.edu.cn

Received 12 December 2019; Accepted 7 February 2020; Published 10 March 2020

Guest Editor: Andrea Visconti

The Even–Mansour cipher has been widely used in block ciphers and lightweight symmetric-key ciphers because of its simple structure and strict provable security. Its research has been a hot topic in cryptography. This paper focuses on the problem to minimize the key material of the Even–Mansour cipher while its security bound remains essentially the same. We introduce four structures of the Even–Mansour cipher with a short key and derive their security by Patarin's H-coefficients technique. These four structures are proven secure up to $\widetilde{O}(2^k/\mu)$ adversarial queries, where k is the bit length of the key material and μ is the maximal multiplicity. Then, we apply them to lightweight authenticated encryption modes and prove their security up to about $\min\{b/2, c, k-\log\mu\}$-bit adversarial queries, where b is the size of the permutation and c is the capacity of the permutation. Finally, we leave it as an open problem to settle the security of the t-round iterated Even–Mansour cipher with short keys.

1. Introduction

In recent years, more and more attention has been paid to lightweight cryptography as smart home, Internet of things (IoT), smart transportation, and 5G/B5G networks are proposed. These new technologies brought convenience to our lives but have introduced a powerful security threat, such as the leakage of the private data in our smart phone. Lightweight cryptography is an effective countermeasure against the security threats in order to achieve the privacy and integrity protections of the sensitive data. Lightweight cryptography is mainly used in resource-constrained devices. The block cipher has become a very vital lightweight symmetric-key cryptography, due to its fast speed, easy implementation, and easy standardization on these devices. It is often used to implement sensitive data encryption, digital signature, message authentication, and key encapsulation schemes in the field of information security and network communication security.

The t-round iterated Even–Mansour cipher is simply described as a pure permutation-based block cipher:

$$y = P_t\left(P_{t-1}\left(\cdots\left(P_1\left(x\oplus K_1\right)\oplus K_2\right)\cdots\oplus K_t\right)\oplus K_{t+1}, \qquad (1)$$

where $(K_1, K_2, \ldots, K_t, K_{t+1})$ is a sequence of n-bit round keys which are usually derived from some master key and $(P_1, P_2, \ldots, P_t)$ is a sequence of t public random permutations. This iterated Even–Mansour cipher, also known as key-alternating ciphers, is of great significance in the design of block ciphers and is also favored in the design of lightweight cryptography. The security of the iterated Even–Mansour ciphers is based on the random permutation model (RPM). In RPM, all permutations are modeled as public random permutation oracles, in other words, anyone can query these permutations and obtain the corresponding responses. The related research includes [1–9].

This paper focuses on the case $t = 1$. Even and Mansour [10] did pioneering work in 1997 and proved that it is birthday-bound secure. That is where the name "Even–Mansour cipher" comes from. The Even–Mansour cipher has some very nice properties, such as simplest structure and strict provable security. Although the research of the Even–Mansour cipher went unnoticed for years, Gold will always shine. Fortunately, it has been a very hot topic in cryptography. In 2012, Dunkelman et al. [11] pointed out that the Even–Mansour cipher is minimal, i.e., any component (either one of the keys or the permutation) is removed; the Even–Mansour cipher becomes trivially breakable. In 2015, Cogliati et al. [12] introduced the tweakable Even–Mansour (TEM) cipher combined by the Even–Mansour cipher and a tweak, and proved its security. Meanwhile, Mouha and Luykx [13] revisited the Even–Mansour cipher and analyzed the multikey security. do Nascimento and Xexeo [14] applied the Even–Mansour cipher to the Internet of Things (IoT) environments and presented a flexible lightweight authenticated encryption mode in 2017. It follows that Cho et al. [15] presented a new family of white-box block ciphers based on the Even–Mansour cipher WEM which achieves balances between performance and security. Farshim et al. [16] analyzed the security of the Even–Mansour cipher under key-dependent messages. In 2018, we described a generalized tweakable Even–Mansour cipher and applied it to authentication and authenticated encryption modes [17].

In the lightweight devices, the storage resources are limited. Therefore, a vital issue is the minimalism and agility of the key material in the design of lightweight ciphers. In this paper, we revisit the Even–Mansour cipher and consider as problem whether we can use the least key material to achieve the same security bound. The Even–Mansour cipher is proven security up to approximately $2^{k/2}$ adversarial queries, where k is the bit-length of the key material. Can we decrease the key material and achieve the same security bound (this bound must be beyond-birthday-bound)?

We answer positively to the question in this paper. We introduce four structures of the Even–Mansour cipher with a short key and present the provable security results. More concretely, we derive their security up to $\widetilde{O}(2^k/\mu)$ adversarial queries using Patarin's H-coefficients technique, where k is the bit-length of the reducing key material and μ is the maximal multiplicity. The Even–Mansour cipher with a short key has many good advantages, such as calculating on-the-fly, avoiding the key schedule, and minimizing the key material. Therefore, it can be widely applied to resource-constrained lightweight devices. Then, we apply its four structures to lightweight authenticated encryption (AE) modes and prove their security up to about $\min\{b/2, c, k - \log\mu\}$-bit adversarial queries, where $b = r + c$ is the size of the permutation and c (resp. r) is the capacity (resp. rate) of the permutation. Finally, we leave it as an open problem to settle the security of the t-round iterated Even–Mansour cipher with short keys.

The rest of this paper is organized as follows. In Section 2, we introduce some preliminaries. In Section 3, we prove the security of the Even–Mansour cipher with a short key. Section 4 describes lightweight AE modes based on four structures of the Even–Mansour cipher with a short key. Section 5 ends up with this paper.

2. Preliminaries

Let $\{0,1\}^b$ be the set of binary strings of length b and $N = 2^b$. For two strings X and Y, let $X\|Y$ or XY be the concatenation of X and Y. Given a string X, we utilize $|X|$ to denote the length in bits of X. Given a nonempty set X, let $x \leftarrow X$ denote an element x drawing from X uniformly at random and $\#X$ be the cardinality of X. Let $\mathrm{Perm}(b)$ stand for the set of permutations on $\{0,1\}^b$. Let $\mathscr{A}^O = 1$ be an event that an adversary $\mathscr{A}$ outputs 1 after interacting with the oracle O. Here, $\mathscr{A}$ never makes a query for which the response is obviously known. Let $\Pr[\mathbf{E}]$ be the probability that the event $\mathbf{E}$ occurs.

2.1. Multiplicity. Let $\{(x_i, y_i)\}_{i=1}^N$ be a set of N evaluations of a permutation P, where $x_i = \overline{x}_i\|\widehat{x}_i$, $y_i = \overline{y}_i\|\widehat{y}_i$. We introduce the total maximal multiplicity as $\mu = \mu_{\mathrm{fwd}} + \mu_{\mathrm{bwd}}$ inspired by [18], where

$$\mu_{\mathrm{fwd}} = \max_a \#\{i = 1,\ldots,N : \overline{x}_i = a \text{ or } \widehat{x}_i = a\}, \tag{2}$$

$$\mu_{\mathrm{bwd}} = \max_a \#\{i = 1,\ldots,N : \overline{y}_i = a \text{ or } \widehat{y}_i = a\}. \tag{3}$$

2.2. H-Coefficients Technique. H-coefficients technique introduced by Patarin [19] is a very important analytical method in the symmetric-key cryptography. We briefly summarize this technique as follows. Consider an information-theoretic adversary $\mathscr{A}$, whose goal is to distinguish a real world X and an ideal world Y and denote the distinguishing advantage of $\mathscr{A}$ as

$$\mathrm{Adv}(\mathscr{A}) = \left|\Pr\left[\mathscr{A}^X = 1\right] - \Pr\left[\mathscr{A}^Y = 1\right]\right|. \tag{4}$$

Without loss of generality, we can assume that $\mathscr{A}$ is a deterministic adversary. The interaction with any of the two worlds X or Y is summarized in a transcript τ. Denote by D_X the probability distribution of transcripts when interacting with X, and similarly, D_Y the distribution of transcripts when interacting with Y. A transcript τ is attainable if $\Pr[D_Y = \tau] > 0$, meaning that it can occur during interaction with Y. Let Γ be the set of attainable transcripts. We denote Γ_1 as a set of good transcripts when interacting with X (Y). Let Γ_2 be a set of bad transcripts such that the probability to obtain any $\tau \in \Gamma_2$ is small in the ideal world $\Gamma = \Gamma_1 \cup \Gamma_2$.

Lemma 1 (H-coefficients lemma [19]). *Fix a deterministic adversary $\mathscr{A}$. Let $\Gamma = \Gamma_1 \cup \Gamma_2$ be a partition of the set of attainable transcripts. Assume that there exists ϵ_1 such that for any $\tau \in \Gamma_1$, one has*

$$\frac{\Pr\left[D_X = \tau\right]}{\Pr\left[D_Y = \tau\right]} \geq 1 - \epsilon_1, \tag{5}$$

and that there exists ϵ_2 such that

$$\Pr\left[D_Y \in \Gamma_2\right] \leq \epsilon_2. \tag{6}$$

Then, the advantage of the adversary $\mathscr{A}$ is

$$\mathrm{Adv}(\mathscr{A}) \leq \epsilon_1 + \epsilon_2. \tag{7}$$

3. The Even–Mansour Cipher with a Short Key

Fix a public permutation $P\colon \{0,1\}^b \longrightarrow \{0,1\}^b$ and integers $r, c,$ and k, such that $b = r + c$ and $k \leq \min\{r, c\}$. Let

$\mathscr{K} = \{0,1\}^k$. The Even–Mansour cipher with a short key, called EM for short, is described in Figure 1. EM takes a uniform random key $K \in \{0,1\}^k$ and a plaintext $x \in \{0,1\}^b$ as inputs and outputs the ciphertext $y = \mathrm{EM}_K^P(x) \in \{0,1\}^b$. Let $\mathrm{pad}_1(K) = 0^{r-k}\|K$ and $\mathrm{pad}_2(K) = 0^{c-k}\|K$. The four structures of EM are, respectively, shown as follows:

$$\begin{aligned}
&(a): \ y = \mathrm{EM}_K^P(x) = P\left(x \oplus \mathrm{pad}_1(K)\|0^c\right) \oplus \mathrm{pad}_1(K)\|0^c,\\
&(b): \ y = \mathrm{EM}_K^P(x) = P\left(x \oplus \mathrm{pad}_1(K)\|0^c\right) \oplus 0^r\|\mathrm{pad}_2(K),\\
&(c): \ y = \mathrm{EM}_K^P(x) = P\left(x \oplus 0^r\|\mathrm{pad}_2(K)\right) \oplus 0^r\|\mathrm{pad}_2(K),\\
&(d): \ y = \mathrm{EM}_K^P(x) = P\left(x \oplus 0^r\|\mathrm{pad}_2(K)\right) \oplus \mathrm{pad}_1(K)\|0^c.
\end{aligned} \tag{8}$$

We consider the security of the Even–Mansour cipher with a short key and obtain the following theorem.

Theorem 1. *For EM_K^P with $K \in \{0,1\}^k$ and $P \leftarrow \mathrm{Perm}(b)$, we have*

$$\mathrm{Adv}_{\mathrm{EM}}\left(q_e, q_p\right) \leq \frac{\mu q_p}{2^k}. \tag{9}$$

The proof of Theorem 1 utilizes the H-coefficients technique. We consider an adversary $\mathscr{A}$ which can interact with $X = (\mathrm{EM}_K^P, P)$ in the real world or $Y = (Q, P)$ in the ideal world, where P and Q are uniform random and independent permutations and K is a (dummy) key. We assume that the adversary $\mathscr{A}$ makes at most q_e construction queries and at most q_p primitive queries. The transcripts can be expressed as this form $\tau = (\mathscr{Q}_e, \mathscr{Q}_p, K)$, where $\mathscr{Q}_e = \{(x_i, y_i)\}_{i=1}^{q_e}$ and $\mathscr{Q}_p = \left\{(u_j, v_j)\right\}_{j=1}^{q_p}$. We start by defining bad transcripts.

Definition 1. We define an attainable transcript $\tau = (\mathscr{Q}_e, \mathscr{Q}_p, K) \in \Gamma$ as bad if one of the two following conditions is fulfilled.

$\mathrm{Bad}_1: \exists (x,y) \in \mathscr{Q}_e$ and $(u,v) \in \mathscr{Q}_p$, such that

$$\begin{aligned}
&(a)\,\&\,(b): \ x \oplus u = \mathrm{pad}_1(K)\|0^c,\\
&(c)\,\&\,(d): \ x \oplus u = 0^r\|\mathrm{pad}_2(K),
\end{aligned} \tag{10}$$

$\mathrm{Bad}_2: \exists (x,y) \in \mathscr{Q}_e$ and $(u,v) \in \mathscr{Q}_p$, such that

$$\begin{aligned}
&(a)\,\&\,(d): \ y \oplus v = \mathrm{pad}_1(K)\|0^c,\\
&(b)\,\&\,(c): \ y \oplus v = 0^r\|\mathrm{pad}_2(K).
\end{aligned} \tag{11}$$

Otherwise we say that τ is good. We denote Γ_{good}, resp. Γ_{bad} the set of good, resp. bad transcripts, and $\Gamma = \Gamma_{\mathrm{good}} \sqcup \Gamma_{\mathrm{bad}}$.

In the real world X, a bad transcript implies that two invocations to P exist with the same input: one directly from querying the primitive oracle P and another one indirectly from querying the construction oracle EM_K^P, while all tuples in $(\mathscr{Q}_e, \mathscr{Q}_p)$ uniquely determine an input-output pair of P for a good transcript. In the ideal world Y, the abovementioned result is clearly established for a bad transcript, while it is not for a good transcript.

We first upper bound the probability of bad transcripts in the ideal world Y by the following lemma.

Lemma 2

$$\Pr\left(D_Y \in \Gamma_{\mathrm{bad}}\right) \leq \frac{\mu q_p}{2^k}. \tag{12}$$

Proof. In the ideal world Y, $(\mathscr{Q}_e, \mathscr{Q}_p)$ is an attainable transcript with a dummy uniform random key $K \in \{0,1\}^k$.

Here, we assume that an adversary $\mathscr{A}$ makes at most q_e construction queries and at most q_p primitive queries. For each $(x,y) \in \mathscr{Q}_e$ and each $(u,v) \in \mathscr{Q}_p$, we obtain at most μ_{fwd} (resp. μ_{bwd}) tuples (x,y) such that $\overline{x} = \overline{u}$ for structures (a) and (b) or $\widehat{x} = \widehat{u}$ for structures (c) and (d) (resp. $\overline{y} = \overline{v}$ for structures (a) and (d) or $\widehat{y} = \widehat{v}$ for structures (b) and (c)) from the property of multiplicity.

It follows that $\Pr(\mathrm{Bad}_1) \leq \mu_{\mathrm{fwd}} q_p / 2^k$ and $\Pr(\mathrm{Bad}_2) \leq \mu_{\mathrm{bwd}} q_p / 2^k$. Hence, the probability of bad transcripts in the ideal world Y is at most $\mu q_p / 2^k$, where $\mu = \mu_{\mathrm{fwd}} + \mu_{\mathrm{bwd}}$.

We then analyze good transcripts and lower bound the ratio $(\Pr[D_X = \tau])/\Pr[D_Y = \tau]$. □

Lemma 3. *For any good transcript τ, one has*

$$\frac{\Pr\left[D_X = \tau\right]}{\Pr\left[D_Y = \tau\right]} \geq 1. \tag{13}$$

Proof. Consider a good transcript $\tau \in \Gamma_{\mathrm{good}}$. Let Ω_X be a nonempty set of all possible oracles in the real world X and Ω_Y be a nonempty set of all possible oracles in the ideal world Y. Therefore, the cardinalities of sets Ω_X and Ω_Y are, respectively, $\#\Omega_X = (2^b)! \cdot 2^k$ and $\#\Omega_Y = (2^b!)^2 \cdot 2^k$. Let $\mathrm{comp}_X(\tau) \subseteq \Omega_X$ and $\mathrm{comp}_Y(\tau) \subseteq \Omega_Y$ be the two sets of oracles compatible with transcript τ. The probabilities appearing in Lemma 1 can be evaluated as follows:

$$\Pr\left(D_X = \tau\right) = \frac{\#\mathrm{comp}_X(\tau)}{\#\Omega_X}, \tag{14}$$

$$\Pr\left(D_Y = \tau\right) = \frac{\#\mathrm{comp}_Y(\tau)}{\#\Omega_Y}. \tag{15}$$

First, we calculate $\#\mathrm{comp}_X(\tau)$. As $\tau \in \Gamma_{\mathrm{good}}$ consists of $q_e + q_p$ query tuples and any query tuple in τ fixes exactly one input-output pair of the underlying permutation oracle, the number of possible oracles in the real world X equals $(2^b - q_e - q_p)!$.

Second, we calculate $\#\mathrm{comp}_Y(\tau)$. The number of possible oracles in the ideal world Y equals $(2^b - q_p)!(2^b - q_e)!$, as P and Q are uniform random and independent permutations.

It follows that

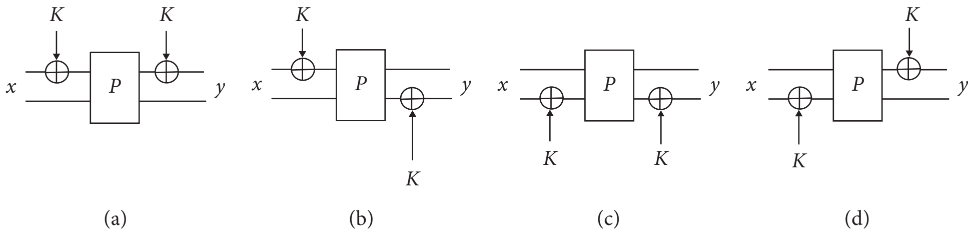

Figure 1: Four structures of EM.

$$\begin{aligned}\Pr\left(D_X=\tau\right)&=\frac{\#\mathrm{comp}_X(\tau)}{\#\Omega_X}\\&=\frac{\left(2^b-q_e-q_p\right)!}{\left(2^b!\right)\cdot 2^k}=\frac{\left(2^b-q_e-q_p\right)!2^b!}{\left(2^b!\right)^2\cdot 2^k}\\&\geq\frac{\left(2^b-q_p\right)!\left(2^b-q_e\right)!}{\left(2^b!\right)^2\cdot 2^k}=\Pr\left(D_Y=\tau\right).\end{aligned}\tag{16}$$

Therefore, we have $(\Pr[D_X=\tau]/\Pr[D_Y=\tau])\geq 1$.

Combining Lemmas 1–3, we can obtain the result of Theorem 1. □

4. Application to Lightweight Authenticated Encryption

With the rises of the smart home, IoT, and 5G/B5G networks, lightweight authenticated encryption (AE) modes are attracting more and more attentions [20–22]. A lightweight AE mode is a lightweight symmetric-key cipher which supports the services of privacy and authenticity of the sensitive data in the devices.

The Even–Mansour cipher with a short key can be directly applied to a lightweight AE mode, which is shown in Figure 2. It consists of an encryption algorithm E and a decryption algorithm D. The encryption algorithm E takes a plaintext M and a key K as inputs and returns a ciphertext C and an authentication tag T, i.e., $C\|T=\mathrm{EM}_K^P(M\|0^c)=E(K,M)$. The decryption algorithm D takes a key K, a ciphertext C, and an authentication tag Tas inputs and returns a plaintext M or a reject symbol $\perp$, i.e., $M/\perp=D(K,C,T)$. If the last c-bit of the EM decryption is 0, then the decryption algorithm D returns M. Otherwise, the decryption algorithm D returns $\perp$.

Let $\Pi=(E,D)$ stand for our lightweight AE modes. We introduce the AE-security model as follows.

Definition 2. (AE security). Let P be a public random permutation. Let $\Pi=(E,D)$ be a P-based AE scheme. Let $\mathscr{A}$ be an adversary which interacts with $X=(E,D,P^{\pm})$ in the real world or $Y=(\$,\perp,P^{\pm})$ in the ideal world. Let $q,p>0$. Then, the AE-security of $\Pi=(E,D)$ is defined as follows:

$$\begin{aligned}\mathrm{Adv}_{\Pi}^{ae}(\mathscr{A})&=\left|\Pr\left[\mathscr{A}^{E,D,P^{\pm}}=1\right]-\Pr\left[\mathscr{A}^{\$,\perp,P^{\pm}}=1\right]\right|,\\\mathrm{Adv}_{\Pi}^{ae}(q,p)&=\max_{\mathscr{A}}\mathrm{Adv}_{\Pi}^{ae}(\mathscr{A}),\end{aligned}\tag{17}$$

where q is the number of queries to the encryption oracle E or the decryption oracle D, p is the number of queries to the random permutation P or its inverse P^{-1}, is a random function which always returns a fresh and random response for each query, and $\perp$ is a symbol which stands for the failure of the decryption oracles.

Theorem 2. *Let $P\leftarrow Perm(b)$ and $b=r+c$. Then,*

$$\mathrm{Adv}_{\Pi}^{ae}(q,p)\leq\frac{\mu p}{2^k}+\frac{q^2}{2^b}+\frac{q}{2^c}.\tag{18}$$

Proof Sketch. Let $\mathscr{A}$ be an adversary with access to the encryption oracle E, the decryption oracle D, and the random permutation P or its inverse P^{-1}. Π can be represented as an EM scheme. We replace the EM modular structure to the random permutation Q. According to Theorem 1, we have

$$\begin{aligned}\mathrm{Adv}_{\Pi}^{ae}(\mathscr{A})&=|Pr\left[\mathscr{A}^{E,D,P^{\pm}}=1\right]-\Pr\left[\mathscr{A}^{\$,\perp,P^{\pm}}=1\right]|\\&\leq\left|\Pr\left[\mathscr{A}^{E,D,P^{\pm}}=1\right]-\Pr\left[\mathscr{A}^{Q,Q^{-1},P^{\pm}}=1\right]\right|\\&\quad+\left|\Pr\left[\mathscr{A}^{Q,Q^{-1},P^{\pm}}=1\right]-\Pr\left[\mathscr{A}^{\$,\perp,P^{\pm}}=1\right]\right|\\&=\frac{\mu p}{2^k}+\left|\Pr\left[\mathscr{A}^{Q,Q^{-1},P^{\pm}}=1\right]-\Pr\left[\mathscr{A}^{\$,\perp,P^{\pm}}=1\right]\right|.\end{aligned}\tag{19}$$

It follows that

$$\begin{aligned}&\left|\Pr\left[\mathscr{A}^{Q,Q^{-1},P^{\pm}}=1\right]-\Pr\left[\mathscr{A}^{\$,\perp,P^{\pm}}=1\right]\right|\\&\leq\left|\Pr\left[\mathscr{A}^{Q,Q^{-1},P^{\pm}}=1\right]-\Pr\left[\mathscr{A}^{Q,\perp,P^{\pm}}=1\right]\right|\\&\quad+\left|\Pr\left[\mathscr{A}^{Q,\perp,P^{\pm}}=1\right]-\Pr\left[\mathscr{A}^{\$,\perp,P^{\pm}}=1\right]\right|\\&\leq\frac{q}{2^c}+\frac{q^2}{2^b},\end{aligned}\tag{20}$$

where $q^2/2^b$ obtained by the PRP-PRF Switch Lemma [23] and $q/2^c$ is from the fact that the successful probability of the adversary is $1/2^c$ for each forgery attempt.

Combining equations (19) and (20), it is easy to draw the result of Theorem 2.

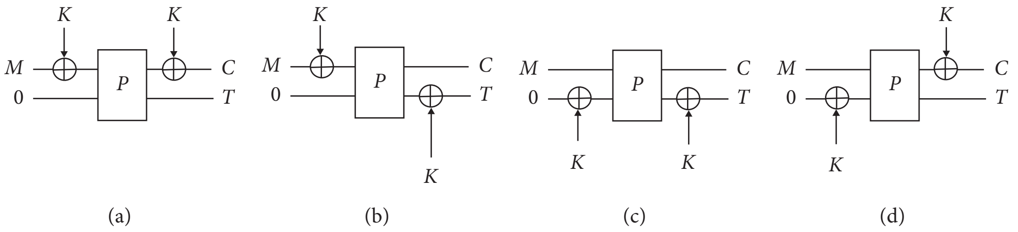

Figure 2: Lightweight authenticated encryption modes.

According to Theorem 2, we can find that these lightweight AE modes ensure about $\min\{b/2, c, k - \log \mu\}$-bit AE-security.

5. Conclusions

The key material is crucial for the secure implementation of cryptographic schemes. Most of devices widely used in smart home, smart transportation, and Internet of Things (IoT) environments are resource constrained. Therefore, in the design of lightweight ciphers, a vital issue is the minimalism and agility of the key material.

In this paper, we revisit the Even–Mansour cipher and discuss this problem whether we can use the least key material to achieve the same (even beyond conventional) security bound in the Even–Mansour cipher. We introduce four structures of the Even–Mansour cipher with a short key and derive security up to $\widetilde{O}(2^k/\mu)$ adversarial queries, where k is the bits of the key material and μ is the maximal multiplicity, using Patarin's H-coefficients technique. Then, we apply them to lightweight authenticated encryption modes and prove their security up to about $\min\{b/2, c, k - \log \mu\}$-bit adversarial queries, where $b = r + c$ is the size of the permutation and c is the capacity of the permutation. Finally, we leave it as an open problem to settle the security of the t-round iterated Even–Mansour cipher with short keys. The Even–Mansour cipher with a short key is proven $(k - \log \mu)$-bit security. It is natural to consider whether our result can be generalized to the t-round iterated Even–Mansour cipher. But the situation of the t-round iterated Even–Mansour cipher with short keys is more complicated. Therefore, it is regarded as an open problem to attract scholars to discuss and analyze it in detail. The Even–Mansour cipher with a short key has many good advantages, such as calculating on-the-fly, avoiding the key schedule, and minimizing the area of the hardware implementation and the key material. Therefore, it can be widely applied to the data security of smart home, Internet of Things, and some lightweight devices.

Data Availability

The data used to support the findings of the study are available within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Research Fund of International Young Scientists (Grant no. 61902195), Natural Science Fund for Colleges and Universities in Jiangsu Province (General Program, Grant no. 19KJB520045), and NUPTSF (Grant no. NY219131).

References

[1] E. Andreeva, A. Bogdanov, Y. Dodis, B. Mennink, and J. P. Steinberger, "On the indifferentiability of key-alternating ciphers," in *Advances in Cryptology–CRYPTO 2013, Lecture Notes in Computer Science*, R. Canetti and J. A. Garay, Eds., vol. 8042, pp. 531–550, Springer, Berlin, Germany, 2013.

[2] A. Bogdanov, L. R. Knudsen, G. Leander, FX. Standaert, J. Steinberger, and E. Tischhauser, "Key-alternating ciphers in a provable setting: encryption using a small number of public permutations," in *Advances in Cryptology–EUROCRYPT 2012, Lecture Notes in Computer Science*, D. Pointcheval and T. Johansson, Eds., vol. 7237, pp. 45–62, Springer, Berlin, Germany, 2012.

[3] S. Chen, R. Lampe, J. Lee, Y. Seurin, and J. Steinberger, "Minimizing the two-round Even-Mansour cipher," *Journal of Cryptology*, vol. 31, no. 4, pp. 1064–1119, 2018.

[4] S. Chen and J. Steinberger, "Tight security bounds for key-alternating ciphers," in *Advances in Cryptology–EUROCRYPT 2014, Lecture Notes in Computer Science*, P. Q. Nguyen and E. Oswald, Eds., vol. 8441, pp. 327–350, Springer, Berlin, Germany, 2014.

[5] B. Cogliati and Y. Seurin, "On the provable security of the iterated Even-Mansour cipher against related-key and chosen-key attacks," in *Advances in Cryptology–EUROCRYPT 2015, Lecture Notes in Computer Science*, E. Oswald and M. Fischlin, Eds., vol. 9056, pp. 584–613, Springer, Berlin, Germany, 2015.

[6] P. Farshim and G. Procter, "The related-key security of iterated Even-Mansour ciphers," in *Fast Software Encryption–FSE 2015, Lecture Notes in Computer Science*, G. Leander, Ed., vol. 9054, pp. 342–363, Springer, Berlin, Germany, 2015.

[7] A. Hosoyamada and K. Aoki, "On quantum related-key attacks on iterated Even-Mansour ciphers," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E102.A, no. 1, pp. 27–34, 2019.

[8] T. Isobe and K. Shibutani, "Meet-in-the-middle key recovery attacks on a single-key two-round Even-Mansour cipher," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E102.A, no. 1, pp. 17–26, 2019.

[9] R. Lampe, J. Patarin, and Y. Seurin, "An asymptotically tight security analysis of the iterated Even-Mansour cipher," in *Advances in Cryptology–ASIACRYPT 2012, Lecture Notes in*

Computer Science, X. Wang and K. Sako, Eds., vol. 7658, pp. 278–295, Springer, Berlin, Germany, 2012.

[10] S. Even and Y. Mansour, “A construction of a cipher from a single pseudorandom permutation,” *Journal of Cryptology*, vol. 10, no. 3, pp. 151–161, 1997.

[11] O. Dunkelman, N. Keller, and A. Shamir, “Minimalism in cryptography: the Even-Mansour scheme revisited,” in *Advances in Cryptology–EUROCRYPT 2012, Lecture Notes in Computer Science*, D. Pointcheval and T. Johansson, Eds., vol. 7237, pp. 336–354, Springer, Berlin, Germany, 2012.

[12] B. Cogliati, R. Lampe, and Y. Seurin, “Tweaking even-mansour ciphers,” in *Advances in Cryptology–CRYPTO 2015, Lecture Notes in Computer Science*, R. Gennaro and M. Robshaw, Eds., vol. 9215, pp. 189–208, Springer, Berlin, Germany, 2015.

[13] N. Mouha and A. Luykx, “Multi-key security: the Even-Mansour construction revisited,” in *Advances in Cryptology–CRYPTO 2015, Lecture Notes in Computer Science*, R. Gennaro and M. Robshaw, Eds., vol. 9215, pp. 209–223, Springer, Berlin, Germany, 2015.

[14] E. M. do Nascimento and J. A. M. Xexeo, “A flexible authenticated lightweight cipher using Even-Mansour construction,” in *Proceedings of the IEEE International Conference on Communications–ICC 2017*, pp. 1–6, IEEE, Paris, France, May 2017.

[15] J. Cho, K. Y. Choi, I. Dinur et al., “WEM: a new family of white-box block ciphers based on the Even-Mansour construction,” in *Cryptographers’ Track at the RSA Conference–CT-RSA 2017, Lecture Notes in Computer Science*, H. Handschuh, Ed., vol. 10159, pp. 293–308, Springer, Berlin, Germany, 2017.

[16] P. Farshim, L. Khati, and D. Vergnaud, “Security of Even-Mansour ciphers under key-dependent messages,” *The IACR Transactions on Symmetric Cryptology*, vol. 2017, no. 2, pp. 84–104, 2017.

[17] P. Zhang and H.-G. Hu, “Generalized tweakable Even-Mansour cipher and its applications,” *Journal of Computer Science and Technology*, vol. 33, no. 6, pp. 1261–1277, 2018.

[18] G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche, “Sponge-based pseudo-random number generators,” in *Cryptographic Hardware and Embedded Systems–CHES 2010, Lecture Notes in Computer Science*, S. Mangard and FX. Standaert, Eds., vol. 6225, pp. 33–47, Springer, Berlin, Germany, 2010.

[19] J. Patarin, “The “coefficients H” technique,” in *Selected Areas in Cryptography–SAC 2008, Lecture Notes in Computer Science*, R. M. Avanzi, L. Keliher, and F. Sica, Eds., vol. 5381, pp. 328–345, Springer, Berlin, Germany, 2008.

[20] A. Chakraborti, N. Datta, M. Nandi, and K. Yasuda, “Beetle family of lightweight and secure authenticated encryption ciphers,” *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2018, no. 2, pp. 218–241, 2018.

[21] G. Hatzivasilis, G. Floros, I. Papaefstathiou, and C. Manifavas, “Lightweight authenticated encryption for embedded on-chip systems,” *Information Security Journal: A Global Perspective*, vol. 25, no. 4–6, pp. 151–161, 2016.

[22] Y. Sasaki and K. Yasuda, “Optimizing online permutation-based AE schemes for lightweight applications,” *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E102.A, no. 1, pp. 35–47, 2019.

[23] M. Bellare and P. Rogaway, “The security of triple encryption and a framework for code-based game-playing proofs,” in *Advances in Cryptology–EUROCRYPT 2006, Lecture Notes in Computer Science*, S. Vaudenay, Ed., vol. 4004, pp. 409–426, Springer, Berlin, Germany, 2006.

Hindawi
Security and Communication Networks
Volume 2019, Article ID 3086975, 10 pages
https://doi.org/10.1155/2019/3086975

Research Article

polarRLCE: A New Code-Based Cryptosystem Using Polar Codes

Jingang Liu,[1] Yongge Wang,[2] Zongxiang Yi,[1] and Zhiqiang Lin [1]

[1] *School of Mathematics and Information Science, Guangzhou University, Guangzhou 510006, China*
[2] *UNC Charlotte, Charlotte, NC 28223, USA*

Correspondence should be addressed to Zhiqiang Lin; linzhiqiang0824@163.com

Received 20 September 2019; Accepted 27 November 2019; Published 26 December 2019

Guest Editor: Andrea Visconti

Security challenges brought about by the upcoming 5G era should be taken seriously. Code-based cryptography leverages difficult problems in coding theory and is one of the main techniques enabling cryptographic primitives in the postquantum scenario. In this work, we propose the first efficient secure scheme based on polar codes (i.e., *polarRLCE*) which is inspired by the RLCE scheme, a candidate for the NIST postquantum cryptography standardization in the first round. In addition to avoiding some weaknesses of the RLCE scheme, we show that, with the proper choice of parameters, using polar codes, it is possible to design an encryption scheme to achieve the intended security level while retaining a reasonably small public key size. In addition, we also present a KEM version of the polarRLCE scheme that can attain a negligible decryption failure rate within the corresponding security parameters. It is shown that our proposal enjoys an apparent advantage to decrease the public key size, especially on the high-security level.

1. Introduction

Cryptography is essential for the security of online communication. However, many commonly used cryptosystems will be completely broken once large quantum computers exist. It is well known that several computation-intensive tasks may be significantly accelerated through algorithms running on a quantum computer, such as Shor's [1] and Grover's [2] algorithm. Current cryptographic protocols, such as RSA and Diffie–Hellman, are proven to be vulnerable under quantum algorithms. This fact pushed cryptographic research to focus on postquantum solutions, i.e., finding new primitives based on more well-suited mathematical problems that may still be difficult to solve for a quantum computer. With this in mind, the US National Institute of Standards and Technology (NIST) is now beginning to prepare for the transition into postquantum cryptography (PQC) and has launched a call for PQC standardization project [3], and this ongoing standardization has moved on to 2^{nd} round thus far. Due to its inherent resistance to attacks by quantum computers, code-based cryptography is one of the main candidates for the PQC standardization call, alongside multivariate and lattice-based schemes.

Code-based cryptography is accepted as quantum computing resistant based on a hard coding theory problem, decoding a random linear code in some metric. Historically, the conservative and well-understood choices for code-based cryptography are the McEliece cryptosystem [4] and its dual variant by Niederreiter [5] using binary Goppa codes. However, they suffer from the disadvantage of having large public key size, in spite of the fast encryption and decryption operations. It is therefore of utmost importance to seek ways to reduce the key sizes for code-based cryptosystems while keeping their security level. After the original proposal of the code-based encryption scheme by McEliece [4] which was based on binary Goppa codes, several variants have been proposed using different codes that allow for smaller keys or more efficient encoding and decoding algorithms, e.g., algebraic geometric (AG) codes [6], generalized Reed–Solomon (GRS) codes [7, 8], low-density parity check (LDPC) codes [9, 10], Reed–Muller (RM) codes [11], low-rank parity check (LRPC) codes [12], and among others. Although the original McEliece cryptosystem remains secure, most of these variants have been successfully cryptanalyzed [13–17]. Despite their promising features, the alternative codes need to be handled carefully due to too much structure.

It is noteworthy that Wang [18, 19] proposed a random linear code-based quantum resistant public key encryption scheme, referred as RLCE, which is a variant of the McEliece encryption scheme. They analyzed an instantiation of the RLCE scheme using GRS codes and introduced randomness in public key, which is based on the juxtaposition of a GRS code with a random linear code. The idea of the RLCE scheme is to use a distortion matrix that mixes some random columns with the structured ones. The advantage of the RLCE scheme is that its security does not depend on any specific structure of underlying linear codes, instead, it is based on the $\mathcal{NP}$-hardness of decoding random linear codes. In such a manner, previous attacks regarding GRS codes based on the technique of filtration distinguisher no longer work. Nevertheless, part of the original parameters was attacked by [20], and RLCE was not selected for the second round of the NIST PQC standardization call.

Polar codes, introduced by Arikan in [21], have received much attention since they are the first class of error-correcting codes that provably achieve the capacity for any symmetric binary discrete memoryless channel (B-DMC) with very efficient encoding and decoding algorithm, whose time complexity scales as $\mathcal{O}(n \log n)$, where n is the length of the code. Because of the good performance and low complexity, polar codes have been adopted for use in future wireless communication systems (e.g., 5G cellular systems). Looking forward, there is a critical need to ensure that 5G techniques, as developed, envision future adoption of PQC for public key cryptosystems.

1.1. Related Work. Within this thread of research, there are two heuristic variants [22, 23] of the McEliece cryptosystem based on polar codes. The first one [23] was broken by Bardet et al. [24] using the structure of the minimum weight codewords. They managed to solve the code equivalence problem for polar codes and thus completely broke the scheme [23] based on polar codes. The second variant was presented by Hooshmand et al. [22] which suggested using the subcode of polar codes. However, we found that the proposal in [22] is useless in practice since 80% ciphertexts could not be decrypted, as discussed in Section 2.4.

1.2. Our Contribution. In this work, we combine the idea of the RLCE scheme by inserting random columns, then propose the first efficient secure scheme based on polar codes (i.e., *polarRLCE*), which can avoid the attack of [24]. Furthermore, possible attacks are outlined and the key size of several choices of parameters is compared to those of known schemes with the same security level. We show that the existing attacks on the proposal scheme do not seem to be effective. More importantly, our proposal enjoys an apparent advantage to decrease the public key size, especially on the high-security level. It allows us to reconsider polar codes as a good candidate for using in code-based cryptography.

The rest of this paper is organized as follows. Some necessary preliminaries such as notation and definitions are given in the Section 2. In Section 3, we present the precise description of the construction of the polarRLCE scheme. Section 4 discusses the known cryptanalytic attacks against our proposal and presents a compact key-encapsulation mechanism (KEM) version regarding polarRLCE. Furthermore, we give the choice of suggested parameters and key size for the achievable security level. Finally, some concluding remarks are made in Section 5.

2. Preliminaries

In this section, we introduce some of the basic background information necessary to follow this paper. Throughout the paper, we will denote vectors by lower-case bold letters, e.g., $\mathbf{m}$. And denote matrices by upper-case bold letters, e.g., $\mathbf{A}$.

2.1. Coding Theory. We begin by briefly reviewing the basic concepts in coding theory and show its application to public-key cryptography.

Definition 1 (linear codes). An $[n, k]$ linear code $\mathscr{C}$ over a finite field $\mathbb{F}_q$ is a k-dimensional linear subspace of $\mathbb{F}_q^n$.

Definition 2 (generator matrix and parity check matrix). A $k \times n$ matrix $\mathbf{G}$ with entries from $\mathbb{F}_q$ having row-span $\mathscr{C}$ is a generator matrix for the $[n, k]$ linear code $\mathscr{C}$. And parity check matrix $\mathbf{H}$ is a $(n-k) \times n$ matrix whose rows generate the orthogonal complement of $\mathscr{C}$.

One can specify a linear code $\mathscr{C}$ via a generator matrix $\mathbf{G} \in \mathbb{F}_q^{k\times n}$ or a parity check matrix $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$ via

$$\mathscr{C} := \left\{ \mathbf{x}\mathbf{G} \in \mathbb{F}_q^n \,\middle|\, \mathbf{x} \in \mathbb{F}_q^k \right\} \text{ or } \mathscr{C} := \left\{ \mathbf{c} \in \mathbb{F}_q^n \,\middle|\, \mathbf{H}\mathbf{c}^T = 0 \right\}. \quad (1)$$

If $\mathbf{G} \in \mathbb{F}_q^{k\times n}$ or $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$, i.e., each matrix entry is chosen uniformly at random from $\mathbb{F}_q$, then we call $\mathscr{C}$ a random linear code.

The code $\mathscr{C}$ can be represented by different generator matrices. An important one is the systematic form, i.e., when each input symbol is directly represented in its first k coordinate positions. For a systematic linear code, the generator matrix $\mathbf{G}$ can always be written as $\mathbf{G} = (\mathbf{I}_k\mathbf{P})$, where $\mathbf{I}_k$ is the identity matrix of size k. And if $\mathbf{G}$ has such a systematic form, then $\mathbf{H} = (-\mathbf{P}^T\mathbf{I}_{n-k})$.

Definition 3 (punctured and shortened codes). Given an $[n, k]$ linear code $\mathscr{C}$, let I be a subset of $\{1, \ldots, n\}$ and the ith entry of a codeword $\mathbf{c} \in \mathscr{C}$ is written as $\mathbf{c}_i$. Then, we define the punctured code $\mathscr{P}_I(\mathscr{C})$ and the shortened code $\mathscr{S}_I(\mathscr{C})$ as

$$\begin{aligned} \mathscr{P}_I(\mathscr{C}) &= \left\{ (\mathbf{c}_i)_{i\notin I} \,\middle|\, \mathbf{c} \in \mathscr{C} \right\}, \\ \mathscr{S}_I(\mathscr{C}) &= \left\{ (\mathbf{c}_i)_{i\notin I} \,\middle|\, \exists \mathbf{c} \in \mathscr{C}, \text{s.t.}\ \forall i \in I, \mathbf{c}_i = 0 \right\}. \end{aligned} \quad (2)$$

Given a subset I of the set of coordinates of a vector $\mathbf{x}$, we denote by $\mathscr{P}_I(\mathbf{x})$ the vector $\mathbf{x}$ punctured at I, that is to say, whose ith entry has been deleted for any $i \in I$.

Lemma 1. *Let $\mathscr{C}$ be a code of dimension k and generator matrix $\mathbf{G}$. Then, the matrix $\mathbf{G}_{\mathscr{P}}$ is a generator matrix for $\mathscr{P}_I(\mathscr{C})$, which can be obtained by deleting the columns from $\mathbf{G}$ index in I.*

We now only consider binary codes, i.e., $q = 2$. The hamming weight $w_H(\mathbf{x})$ of a binary vector in $\mathbf{x} \in \mathbb{F}_2^n$ is the number of nonzero entries in the vector. And the minimum hamming distance of the code $\mathscr{C}$ is defined as $d = \min\{w_H(\mathbf{x} - \mathbf{y})\}$, where $\mathbf{x} \neq \mathbf{y}$.

2.2. McEliece's Public-Key Cryptosystem. In 1978, McEliece presented in his seminal paper [4] the first code-based public key encryption system, which relied on Goppa codes to form the secret key. And a permutation matrix and an invertible matrix are used for scrambling and concealing secret key. It employs an $[n, k]$ linear code $\mathscr{C}$ over $\mathbb{F}_2$, with an error-correcting capability of t errors, for which an efficient decoding algorithm is known. The general key generation, encryption, and decryption steps for the original proposal in [4] work as follows.

Private key: it consists of three matrices $\mathbf{G}$, $\mathbf{S}$, and $\mathbf{P}$, where $\mathbf{G}$ is an $k \times n$ generator matrix of this code, $\mathbf{S}$ is an arbitrary $k \times k$ binary nonsingular matrix (called the scrambling matrix), and $\mathbf{P}$ is an $n \times n$ random permutation matrix.

Public key: it is composed of the $k \times n$ matrix $\mathbf{G}'$ defined by $\mathbf{G}' = \mathbf{SGP}$ and the error-correcting capability with t.

Encryption: to encrypt the message $\mathbf{m} \in \mathbb{F}_2^k$ and choose a random error vector $\mathbf{e} \in \mathbb{F}_2^n$ with weight $w_H(\mathbf{e}) \leq t$, then the corresponding ciphertext is computed as

$$y = \mathbf{m}\mathbf{G}' + \mathbf{e}. \tag{3}$$

Decryption: the decryption procedure consists in computing

$$y\mathbf{P}^{-1} = \mathbf{mSG} + \mathbf{e}\mathbf{P}^{-1}, \tag{4}$$

and using a fast decoding algorithm for Goppa code $\mathscr{C}$ to recover $\mathbf{mS}$. The message is then recovered by $\mathbf{m} = (\mathbf{mS})\mathbf{S}^{-1}$.

Notice that multiplying the error vector by a permutation does not change the weight of the vector. One can easily verify the correctness of the scheme by checking

$$\text{Decrypt}(\text{Encrypt}(\mathbf{m}, \text{pk}), \text{sk}) = \mathbf{m}. \tag{5}$$

A dual version of the McEliece cryptosystem that uses the parity-check matrix instead of the generating matrix has been proposed by Niederreiter in [5]. Following the idea of [25], the Niederreiter system and the McEliece system are equivalent in terms of security.

Knowing the description of the selected Goppa code $\mathscr{C}$ allows efficient decoding, since there are many efficient decoding algorithms for this problem running in polynomial time. However, knowing only the public key $\mathbf{G}'$, the attacker is facing a decoding problem for a code that looks like a random code, which is $\mathscr{NP}$-hardness. The attacker can either try to decode an intercepted ciphertext (message recovery attack) or try to recover the secret matrix $\mathbf{G}$ from the public matrix $\mathbf{G}'$ (key-recovery attack).

The security level of the McEliece system has remained remarkably stable, despite dozens of attack papers over 40 years, regardless of the original McEliece parameters being designed for only 64-bit security level. For instance, as recommended by Bernstein et al. [26], the McEliece scheme with binary Goppa codes using code length $n = 2960$ and code dimension $k = 2288$ and adding $t = 57$ errors can achieve 128-bit security level, Thus, the corresponding public key size is 187.69 KBytes.

2.3. Polar Codes Construction. We first recall the basic facts about polar codes. As shown in the seminal work by Arikan [21], for any B-DMC, there exists a polar code of block length $n = 2^m$ which is characterized by the information bit set $\mathscr{A}$ with exponentially small word-error rate under successive cancellation (SC) decoder. A polar code may be specified completely by $(n, k, \mathscr{F})$, where n is the length of a codeword in bits, k is the number of information bits encoded per codeword, and $\mathscr{F}$ is a set of $n - k$ integer indices called frozen bit locations from $\{0, 1, \ldots, n - 1\}$. The k more reliable subchannels (based on the polarization phenomenon) with indices in set $\mathscr{A}$ carry information bits and the rest subchannels included in the complementary set $\mathscr{A}^c$ (i.e., the set $\mathscr{F}$) can be set to fixed bit values, such as all zeros. Generally, the challenge is to select the information bits set $\mathscr{A}$ or, more precisely, the methods that are proposed for finding the indices of good polarized channels.

For a binary polar code of length $n = 2^m$, the polar encoding of an input vector is carried out by the polarization transformation matrix $\mathbf{G}_n = \mathbf{F}^{\otimes m}$, which is the mth Kronecker power of the 2×2 kernel matrix:

$$\mathbf{F} = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}. \tag{6}$$

For a given noise channel, the generator matrix $\mathbf{G}$ of an $[n, k]$ polar code is defined as the submatrix of $\mathbf{G}_n$ consisting of k rows with indices corresponding to information set $\mathscr{A} = \{i_1, i_2, \ldots, i_k\}$. Roughly speaking these rows are chosen in such a way that it gives good performance for the SC decoder. These codes come equipped with an SC decoder whose decoding complexity scales as $\mathscr{O}(n \log n)$ (see [21] for more details).

The idea of exploiting polar codes in cryptography came in a natural way since polar codes benefit of various interesting properties: can achieve Shannon capacity for the class of binary discrete memoryless channels, attain better performance (lower decoding errors) because of the channel polarization along with the increased block length, posses efficient encoding and decoding procedures, etc. Even though polar codes are closely related to RM codes, the techniques used for the cryptanalysis of RM codes do not work on polar codes.

2.4. The Proposal by Hooshmand et al. [22]. The error-correcting capacity of polar codes [21] does not only depend on the code length but also on other factors such as the code rate and the designed channel. However, the error-correcting capacity was merely set to be a fixed value of $t = 2\sqrt{n} - 1$ in the proposal by Hooshmand et al. [22], they did not consider the error probability of decoding. For instance, they claimed that one can use $[2048, 1750]$-polar code with $t = 89$, which followed Theorem 8 in [27]. Actually, Theorem 8 from [27] is only suitable for the concatenated polar codes with respect to the length of burst-errors as stated in [27]. Nevertheless, the proposal in [22] used random errors through the encryption process. With these parameters in [22], we performed numerical simulation using MATLAB R2018a where 10^5 decoding trails are exploited under SC decoder [21], and the experiment result indicates that the decoding error probability is nearly 0.8, i.e., 80% ciphertexts could not be decrypted and cannot be employed in a practical environment. With respect to our proposal, the error probability is approximately 2^{-14} (see Section 3). Furthermore, we transform the basic polarRLCE into a key-encapsulation mechanism (KEM) version, which can achieve the negligible decryption failure rate (DFR) within the corresponding security parameters.

3. Our Proposed Scheme of polarRLCE

In this section, we describe our new variant of the McEliece cryptosystem by exploiting the method of RLCE [18, 19] scheme. More precisely, the procedures of our polarRLCE are specified as follows.

Key generation: according to the construction of the polar code in Section 2.3,

(i) Choose an $[n, k]$ polar code with the generator matrix $\mathbf{G}$ of length n and dimension k.

(ii) Generate w random column vectors $\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_w$, and let

$$\mathbf{G}_1 = \left(\mathbf{g}_1, \ldots, \mathbf{g}_{n-w}, \mathbf{g}_{n-w+1}, \mathbf{r}_1, \ldots, \mathbf{g}_n, \mathbf{r}_w\right), \tag{7}$$

be the $k \times (n + w)$ matrix obtained by inserting w random $k \times 1$ column vectors $\mathbf{r}_i$ into matrix $\mathbf{G}$.

(iii) To mix the columns, choose w random nonsingular binary 2×2 matrices $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_w$. Denote $\mathbf{A}$ with the $(n + w) \times (n + w)$ block-diagonal matrix:

$$\mathbf{A} = \begin{pmatrix} \mathbf{I}_{n-w} & & & (0) \\ & \mathbf{A}_1 & & \\ & & \ddots & \\ (0) & & & \mathbf{A}_w \end{pmatrix}. \tag{8}$$

(iv) Let $\mathbf{S}$ be a randomly chosen $k \times k$ nonsingular matrix and $\mathbf{P}$ be the $(n + w) \times (n + w)$ permutation matrix.

(v) The public key $k \times (n + w)$ matrix is defined as

$$\mathbf{G}_{\text{pub}} = \mathbf{S}\mathbf{G}_1\mathbf{A}\mathbf{P}. \tag{9}$$

Then, the public key and private key are given, respectively, by

$$\mathbf{G}_{\text{pub}} \text{ and } (\mathbf{G}, \mathbf{S}, \mathbf{P}, \mathbf{A}). \tag{10}$$

Encryption: let $\mathbf{m} \in \mathbb{F}_2^k$ be the message to be encrypted. Then, we randomly generate error vector $\mathbf{e} \in \mathbb{F}_2^{n+w}$ such that the hamming weight $w_{\text{H}}(\mathbf{e}) \leq t$. Compute the corresponding ciphertext:

$$\mathbf{c} = \mathbf{m}\mathbf{G}_{\text{pub}} + \mathbf{e}. \tag{11}$$

Decryption: to decrypt the received ciphertext $\mathbf{c}$,

(i) Calculate $\mathbf{c}\mathbf{P}^{-1}\mathbf{A}^{-1} = \left(c_1', c_2', \ldots, c_{n+w}'\right)$.

(ii) Delete w entries at the $\mathbf{r}_i$ position of row vector $\left(c_1', c_2', \ldots, c_{n+w}'\right)$. We denote the obtained n-length vector by

$$c' = \left(c_1', c_2', \ldots, c_{n-w+1}', c_{n-w+3}', c_{n-w+5}', \ldots, c_{n+w-1}'\right). \tag{12}$$

(iii) It is easy to check that $c' = \mathbf{m}\mathbf{S}\mathbf{G} + e'$ for some error vector $e' \in \mathbb{F}_2^n$, where $w_{\text{H}}(e') \leq t$. Then, using the efficient decoding algorithm, one can recover the corresponding message $\mathbf{m}$.

For the purpose of constructing polar code used in our proposed variant scheme, we consider here the binary symmetric channel (BSC) with crossover probability $\varepsilon = 0.05$. For instance, to achieve 128-bit security, for reliable decoding and keeping reasonably small key size with enough security level, we will set the choice of parameters such that $n = 2^{11}$, $k = 500$, and $w = 50$. Following the method of Dragoi [28], validated through exhaustive simulation, we can choose the error vector weight of $t = 285$ with the reasonable decoding error probability is approximately 2^{-14}.

Remark 1. Please note that our scheme allows occasional decryption failures for valid ciphertexts (similar to some NIST PQC submissions), which is inherited from the decoding algorithm. However, for the good performance of polar codes, one can easily resolve this issue through repeated encryption as presented by Eaton et al. [29] which can reduce the decryption failure rate to a level negligible in the security parameter, without altering the whole parameters.

4. Security Analysis

In this section we will discuss several possible attacks against our proposed polarRLCE scheme in 3. There are two main attacks to thwart, i.e., key structural attack and decoding attack.

Furthermore, if the code $\mathscr{C}$, whose generator matrix is used as a part of the public key, could be distinguished, then an adversary could exploit the structure of $\mathscr{C}$, and this would also possibly allow the adversary to develop faster attacking algorithms. Indeed, most of these variations of the McEliece system are vulnerable to structural attacks because of the algebraic structure of underlying codes.

4.1. Brute Force Attack. A brute force attack is a trial-and-error method used to obtain the correct keys. For our proposed scheme, recall that the private key $(\mathbf{G}, \mathbf{S}, \mathbf{P}, \mathbf{A})$ is obtained randomly. Moreover, the number of candidate invertible scrambling matrix $\mathbf{S}_{k\times k}$ over $\mathbb{F}_2$ is

$$\mathscr{N} = \prod_{i=1}^{k}\left(2^k - 2^{i-1}\right) = 2^{k^2}\prod_{i=1}^{k}\left(1 - 2^i\right) > 2^{k^2-2}. \tag{13}$$

By putting in the suggested parameters (with 128-bit security) $n = 2048$, $k = 500$, and $w = 50$, it turns out that it is as well infeasible to retrieve the other three elements building the private key just by guessing, since there exist $(n + w)! = 2048! \gg 2^{128}$ different matrix $\mathbf{P}$ and nearly $\mathscr{N} = 2^{500^2} \gg 2^{128}$ choices for $\mathbf{S}$. Moreover, the candidate of block-diagonal matrix $\mathbf{A}$ is $6^w = 2^{129.25}$. Hence, the complexity of the exhaustive search attack against our scheme has an exponential time, which indicates this attack is impractical.

4.2. Square Attack. In this section, we study the square attack on our polarRLCE. There has been an increased interest in the square (a.k.a. Schur product) of linear codes in the last years (cf. [30]). Another and more recent application of the Schur product concerns cryptanalysis of code-based public key cryptosystems. In this context, the Schur product is a very powerful operation which can help to distinguish secret codes from random ones.

In fact, the method of inserting random columns or rows in the secret matrix has indeed proposed [7, 8, 31] to avoid structural attacks on similar versions of the McEliece cryptosystem. Although this proposal has effectively avoided the original attack, recent studies [14, 32, 33] have shown that in the case of GRS codes or RM codes, the random columns can be found through the consideration of the dimension of the Schur product code.

Definition 4 (Schur product). Let $\mathbf{x}, \mathbf{y} \in \mathbb{F}_2^n$, then the Schur product of two vectors is denoted by

$$\mathbf{x} * \mathbf{y} = (x_1y_1, x_2y_2, \ldots, x_ny_n). \tag{14}$$

Definition 5 (square code). Let $\mathscr{A}$ and $\mathscr{B}$ be linear codes with length n. The Schur product of the two codes is the vector space spanned by all products $\mathbf{a} * \mathbf{b}$ with $\mathbf{a} \in \mathscr{A}$ and $\mathbf{b} \in \mathscr{B}$:

$$\langle \mathscr{A} * \mathscr{B}\rangle = \langle\{\mathbf{a} * \mathbf{b} \mid \mathbf{a} \in \mathscr{A} \text{ and } \mathbf{b} \in \mathscr{B}\}\rangle. \tag{15}$$

If $\mathscr{A} = \mathscr{B}$, then we call $\langle \mathscr{A} * \mathscr{A}\rangle$ the square code of $\mathscr{A}$ and denote it by $\langle \mathscr{A}^2\rangle$. The impact of the square code on the code-based cryptosystem becomes clear when we study the dimension of these constructions.

Definition 6 (Schur matrix). Let $\mathbf{G}$ be a $k \times n$ matrix, with rows $(g_i)_{1\le i\le k}$. The Schur matrix of $\mathbf{G}$, denoted by $S(\mathbf{G})$ consists of the rows $g_i * g_j$ for $1 \le i \le j \le k$.

We observe that if $\mathbf{G}$ is a generator matrix of a code $\mathscr{C}$, then its Schur matrix $S(\mathbf{G})$ (or the submatrix which contains the linear independent rows of $S(\mathbf{G})$) is a generator matrix of the square code of $\mathscr{C}$. For the $k \times n$ matrix $\mathbf{G}$, the matrix $S(\mathbf{G})$ at most has the size $\begin{pmatrix} k+1 \\ 2 \end{pmatrix} \times n$ (refer to [30]).

It is well known that the square of a linear code $\mathscr{C}[n, k]$ has the dimension

$$\dim\left(\mathscr{C}^2\right) \le \min\left\{n, \frac{1}{2}k(k+1)\right\}, \tag{16}$$

and a random linear code attains this upper bound with high probability.

One of the key features in most of the successful cryptanalysis efforts has been that the proposed codes have small Schur-product dimension which leads to key recovery or distinguishing attacks. In particular, this lends credence to the idea that codes with small Schur-product dimension appear to be unsuitable for use in the McEliece framework. For instance, if the code is generalized Reed–Solomon (GRS) code, then it satisfies

$$\dim\left(\mathscr{C}^2\right) = \min\{n, 2k - 1\}, \tag{17}$$

and fulfills this lower bound with equality, i.e., for $k < (n/2)$, their square dimension is much smaller than one expects from a random code. Actually, this fact is, e.g., utilized by [14, 33] to build an effective distinguisher, yielding a structural attack on the GRS-based McEliece cryptosystem.

Looking at the definition of the square code, we observe that it is generated by all possible Schur-products of every pair of (nonnecessarily distinct) codewords in the given linear code. Therefore, it is natural to expect that the dimension of the square code is "as large as possible." In other words, for a randomly chosen linear code $\mathscr{R}$, we expect that inequality (16) is actually an equality with very high probability.

Let us consider the recent work [34] which reported that it might possibly exist as a heuristic distinguisher, if given two specific weakly decreasing sets. However, in the case of our polarRLCE scheme, such sets could not be found easily because of the extended public codes by inserting random columns.

To illustrate the square attack, we performed simulation by generating 10,000 random sets of the public key matrix. Our experimental result shows that, as in the case of the proposed polarRLCE scheme, the square code of the public code can always reach the maximal dimension bound. Considering the choice of parameters (with 128-bit security) such that $n = 2048$, $k = 500$, and $w = 50$. So, we can obtain the $k \times (n + w)$ public key matrix $\mathbf{G}_{\text{pub}}$. Denote the extended public code as $\mathscr{C}_{\text{pub}}$. Hence, from inequality (16), we have

$$\dim\left(\mathscr{C}_{\text{pub}}^{2}\right) = \dim\left(S\left(\mathbf{G}_{\text{pub}}\right)\right) \le \min\left\{n + w, \frac{1}{2}k(k+1)\right\}. \tag{18}$$

For the proposed parameters, we observed experimentally that the dimension of the public matrix by the square product always reach maximum, that is to say,

$$\dim\left(S\left(\mathbf{G}_{\text{pub}}\right)\right) = n + w = 2098. \tag{19}$$

Furthermore, after perform random puncturing operations, $\mathscr{P}_I(\mathbf{G}_{\text{pub}})$, alternatively, we can obtain

$$\dim\left(S\left(\mathscr{P}_I\left(\mathbf{G}_{\text{pub}}\right)\right)\right) = n + w - |I|. \tag{20}$$

On the basis of the observations made as stated above, we claim that the technique of square attack regarding our polarRLCE could not be used to distinguish from random codes.

4.3. Key-Recovery Attack. The key-recovery attack is one of the important ways of structural attack, consists in recovering the private key from the public key. In this case, the methods are specific to the code family. In order to compute the private key of a given public key, it is often reduced to solve the code equivalence problem.

Definition 7. Let $\mathbf{G}$ and $\mathbf{G}^*$ be the generating matrix for two $[n, k]$ binary linear codes. Given $\mathbf{G}$ and $\mathbf{G}^*$, there exist a $k \times k$ binary invertible matrix $\mathbf{S}$ and $n \times n$ permutation matrix $\mathbf{P}$ such that $\mathbf{G}^* = \mathbf{SGP}$?.

This problem was first studied by Petrank and Roth [35] over the binary field. And the most common algorithm used to solve this problem is the support splitting algorithm (SSA) [36]. SSA is very efficient in the random case, but it cannot be used in the case of codes with large hulls or codes with large permutation group such as Goppa codes and polar codes.

However, a very effective structural attack on the variant [23] using polar codes was introduced by Bardet et al. in [24]. Firstly, they managed to determine exactly the structure of the minimum weight codeword of the original polar codes. Then, they solved the code equivalence problem for polar codes with respect to decreasing monomial codes. Notice that this attack is very specific to the simple usage of polar codes in [23].

Regarding our proposal, there is a really effective way of protecting the scheme since the structure of the private code is someway shattered by inserting random elements. Thus, even though one can find enough low-weight codewords, while they are not subject to the original polar code, because that these codewords possess an extended length $n + w$ which is generated by the public key matrix $\mathbf{G}_{\text{pub}}$. The natural way is to perform puncturing operations, but an exponential number of codewords need to check since there are

$$\binom{n+w}{w} = \binom{2098}{50} = 2^{336.68}. \tag{21}$$

On the other hand, the adversary cannot identify the inserted positions by distinguishing attack or square attack as stated on Section 4.2. So, the code equivalence problem becomes even more complicated to solve in this case. Therefore, the attack by [24, 34] does not apply directly to our proposed polarRLCE scheme.

Finally, we notice that very recently, Couvreur et al. [20] presented a key-recovery attack on half the parameter sets proposed in the RLCE scheme [19]. They showed that it is possible to distinguish some keys from random codes by computing the square of some shortened public codes. The set of positions $\{1, \ldots, n + w\}$ splitted into four parts based on the fact that the entry of any GRS codeword satisfies a specific expression formalization, i.e., $\dim \text{GRS}_k(x, y)^2 = 2k - 1$, and then recognizes the twin positions. While polar codes are used for the aforementioned situation because of the different structure between polar codes and GRS codes.

According to the aforementioned analysis and the fact that we found no other distinguishing methods for our proposal, we claim that it is indeed able to avoid the key-recovery attack.

4.4. Message-Decoding Attack. Message-decoding attack is an important issue in code-based cryptography. The problem of recovering the private message from a ciphertext is directly related to the hardness of generic decoding for the linear code. One possibility attack to recover the message is information set decoding (ISD) algorithm, which means to decode a random linear code without exploiting any structural property of the code. The ISD algorithm searches for an information set such that the error positions are all out of the information set. The work factor of ISD clearly increases with the number of errors added in the encryption process. Thus, when choosing parameters, we will focus mainly on defeating attacks of the ISD family.

This technique was first introduced by Prange [37]. Hereafter, numerous different algorithmic techniques have been explored to improve complexity of ISD algorithm. Among several variants [26, 38–40] and generalizations, it is noteworthy that most modern ISD algorithms are based on Stern's [38] algorithm, which incorporates collision search methods to speed up decoding.

Thus, we will move on to analyze the complexity of Stern's algorithm. Similarly, they try to find a t-weight codeword in an $[n, k]$ linear code $\mathscr{C}$ generated by $\mathbb{F}_2^{k\times n}$. More precisely, apart from the generator matrix $\mathbf{G}$, the algorithm takes as input additional integer parameters p and l such that $0 \le p \le t$ and $0 \le l \le n - k$. Each iteration consists of the steps described in Algorithm 1.

$$\mathscr{P}_{\text{stern}} = \frac{\binom{k/2}{p}^2 \binom{n-k-l}{t-2p}}{\binom{n}{t}}. \tag{22}$$

Then, the probability of success in one single iteration is

And the cost of one iteration of Stern's algorithm is as follows:

```
Input: Generator matrix $\mathbf{G} \in \mathbb{F}_2^{k\times n}$, with parameters $t$, $p$, and $l$.
Output: Codeword $\mathbf{c} \in \mathscr{C}$, s.t. $w_H(\mathbf{c}) = t$.
1 Select a random information set $\mathscr{I}$ from $\{1, 2, \ldots, n\}$ and divide it into two equal size subsets $\mathscr{X}$ and $\mathscr{Y}$. Moreover, select a size-$l$ subset $\mathscr{Z}$ of $\{1, 2, \ldots, n\}\backslash\mathscr{I}$.
2 Permutate $\mathbf{G}$ randomly, and let $\mathbf{G}_{\text{sys}}$ denote the systematic form: $\mathbf{G}_{\text{sys}} = (\mathbf{IQJ})$
  where $\mathbf{I}$ is the $k \times k$ identity matrix, $\mathbf{Q}$ is a $k \times l$ matrix and $\mathbf{J}$ is a $k \times (n - k - l)$ matrix.
3 Let $\mathbf{u}$ run through all $p$-weight vectors of length $k/2$. Then, put all vectors $\mathbf{x} = (\mathbf{u}0)\mathbf{G}_{\text{sys}}$ in a sorted list $\mathscr{L}_1$, sorted according to index $\phi(\mathbf{x})$, with $\phi(\mathbf{x})$ being the value of a vector $\mathbf{x}$ in positions $k + 1$ to $k + l$, i.e., $\phi(\mathbf{x}) = (x_{k+1}, x_{k+2}, \ldots, x_{k+l})$.
4 Then, construct another list $\mathscr{L}_2$ sorted according to $\phi(\mathbf{x})$, containing all vectors $\mathbf{x} = (0\mathbf{u})\mathbf{G}_{\text{sys}}$, where $\mathbf{u}$ run through all $p$-weight vectors of length $k/2$.
5 Add all pairs of vectors $\mathbf{x} \in \mathscr{L}_1$ and $x' \in \mathscr{L}_2$ for which $\phi(\mathbf{x}) = \phi(x')$ and put in a new list $\mathscr{L}$.
6 if there exists $\mathbf{x} \in \mathscr{L}$, s.t. $w_{\text{H}}(\mathbf{x}) = t - 2p$ then
7     return the codeword $\mathbf{c} \in \mathscr{C}$ corresponding to $\mathbf{x}$.
8 else
9     go back to Step 1;
10 end
```

Algorithm 1: Stern's ISD algorithm.

$$(n-k)^2(n+k) + \left(\frac{k}{2} - p + 1\right) + 2l\binom{\frac{k}{2}}{p} + 4p(t - 2p + 1)\binom{\frac{k}{2}}{p}^2 2^{-l}. \tag{23}$$

We refer to the improved version of ISD attack algorithm in [26, 39], which is a generalization of Stern's algorithm [38]. And they pointed out that for small fields (e.g., in our case, $\mathbb{F}_2$), choosing the new algorithm parameters c and r $(1 < r \leq c)$ should be taken into account, which can yield good speedups on the Gaussian elimination cost of each iteration. Furthermore, they offer an improved tool, which allows to compute a rough approximation of the security level of a code-based cryptosystem against information set decoding attacks.

For a practical evaluation of the ISD running times and corresponding security level, similarly, most of the NIST PQCrypto code-based submissions exploited this complexity computation tool to determine the security level of their proposals, and we also use this tool to indicate the security level of our implementation. Note that the ciphertext length should be $n + w$ instead of n in our case. According to this computation tool, we test different input parameters to classify expected bit security level $\kappa \coloneqq 128, 192, 256$, respectively (see Section 4.6 for more details).

4.5. The KEM of polarRLCE Scheme. Key-encapsulation mechanisms (KEMs) are a common stepping stone aiming for the strong security goal, i.e., indistinguishability against adaptive chosen-ciphertext attacks (IND-CCA2). We also suggest a key-encapsulation mechanism (KEM) version of our polarRLCE scheme, consisting of three algorithms: KEM = (KeyGen, Encaps, Decaps), by applying a transformation of Eaton et al.'s [29] observation. A favorable feature of this proposal is that the process of polarRLCE is convenient, and it enables our KEM-DEM version to achieve the negligible decryption failure rate within the corresponding security parameters. Let $\mathscr{G}$, $\mathscr{H}$, and $\mathscr{K}$ be hash functions, typically SHA-3 as advised by NIST. Here, we show the KEM-DEM version below.

KeyGen (pk, sk): exactly the same as polarRLCE key generation (Section 3), and generate a public/secret key pair (pk, sk).

Encaps ($\mathbf{c}$, K): encapsulate a shared key K in ciphertext $\mathbf{c}$ as follows:

(i) Pick a seed $\mathbf{s} \in \{0, 1\}^k$ and set parallel degree P
(ii) For $i \in \{1, 2, \ldots, P\}$, let $\mathbf{e}_i = \mathscr{G}(\mathbf{s} \mid i)$
(iii) Compute $\mathbf{x}_i = \mathbf{s} + \mathscr{H}(\mathbf{e}_i \mid i)$, and the corresponding ciphertext $\mathbf{c}_i = \text{Enc.}(\text{pk}, \mathbf{x}_i, \mathbf{e}_i)$
(iv) Output the shared key $\text{K} = \mathscr{K}(\mathbf{s})$ and $\mathbf{c} = (\mathbf{c}_1, \ldots, \mathbf{c}_P)$

Decaps (K): decapsulate the shared key K from ciphertext $\mathbf{c}$ with sk.

(i) Decrypt ciphertext $\mathbf{c}$ to get $(\mathbf{x}_i^*, \mathbf{e}_i^*)$, where $i \in \{1, 2, \ldots, P\}$.
(ii) Let $j = i$ when the last step successfully decrypt for the current status, then compute $\mathbf{s} = \mathbf{x}_j + \mathscr{H}(\mathbf{e}_j | j)$.
(iii) Compute $\mathbf{c}_i^* = \text{Enc}\cdot(\text{pk}, \mathbf{x}_i^*, \mathbf{e}_i^*)$, obtain $\mathbf{c}^*$ and verify that $\mathbf{c}^* = \mathbf{c}$. If so, return the shared key $\text{K} = \mathscr{K}(\mathbf{s})$.

The same construction was proven to have IND-CCA2 guarantees in the work by [29]. More details regarding the security reduction can be found in [29].

To provide IND-CCA2 for a given security level 2^{κ}, it is required for the decapsulation to have a correctness error $\delta \leq 2^{-\kappa}$. Recall that the DFR of our polarRLCE scheme is no more than 2^{-14}. The resulting ciphertext includes several independent encapsulations with the same key so that a decapsulation failure occurs only if a decryption failure occurs in every instance of the underlying polarRLCE scheme. Willing to target a DFR of $2^{-\kappa}$, we can select the parallel degree $P = 10, 14, 19$, respectively. Thus, our KEM

Table 1: Set of parameters for polarRLCE scheme.

κ	$[n, k, t]$	w	$\mathscr{PK}$	$\mathscr{SK}$	$\mathscr{CT}$	$\mathscr{W}_{\mathrm{ISD}}$
128	$[2^{11}, 500, 285]$	50	97.53	30.54	262.25	130.62
192	$[2^{12}, 585, 760]$	75	256.08	41.81	531.38	193.84
256	$[2^{12}, 960, 610]$	100	379.22	112.55	524.50	257.35

Table 2: Public-key size comparison (in KBytes).

κ	Our	McEliece	RLCE	NTS-KEM	Classic-McEliece
128	97.53	187.69	187.53	312	255
192	256.08	489.4	446.36	907.97	511.88
256	379.22	936.02	1212.86	1386.43	1326

version achieves the desired negligible target DFR value 2^{-140}, 2^{-196}, and 2^{-266}.

4.6. Suggested Parameters for polarRLCE. In this section, we give the suggested parameters in 1 for our polarRLCE scheme, with the three most relevant standard security levels, 128-bit, 192-bit, and 256-bit. Besides, a comparison of the public key size for our suggested parameters with RLCE [19] (the secure second group parameters) and the original McEliece [26] scheme (under binary Goppa codes) is given in 2, together with the state-of-the-art NTS-KEM [41] and Classic-McEliece [42], which are moving on to the 2nd round of the NIST PQC standardization process.

For convenience, we introduce the following notations of each column list in the tables:

(i) κ: security level

(ii) w: the number of inserted columns

(iii) $[n, k, t]$: code length n, code dimension k, and t is the error-correcting ability

(iv) $\mathscr{PK}$: public-key size in kB

(v) $\mathscr{SK}$: private-key size in KBytes

(vi) $\mathscr{CT}$: ciphertext size in Bytes

(vii) $\mathscr{W}_{\mathrm{ISD}}$: the work factor of ISD attack algorithm

We remark that the parameters given in Table 1 may be vulnerable to the attack under the quantum random oracle model (*QROM* [43]). Here, we present the parameters solely to illustrate the rationality of our construction which, to our best knowledge, are secure against current known attacks.

From Table 2, we can see that our scheme can reduce the public key size of the original McEliece scheme by at least 52%. It has the apparent advantage to decrease the key size, especially on the high-security level. However, compared to the candidates based on hamming (rank) quasi-cyclic (QC) codes, the public key size of our proposal is inferior to them. Nevertheless, a new type of statistical analysis, called reaction attacks [44, 45], are threatening these schemes with a specific QC structure of the underlying codes [46, 47]. As a final remark, it would be required to consider the impact of reaction attacks even without the QC structure in our proposal.

5. Conclusion

To summarize, we have proposed a new variant of the code-based encryption scheme by exploring polar codes, benefitting the lower encoding and decoding complexity. We show that, for the proper choice of parameters together with the state-of-the-art cryptanalysis, it is still possible to design secure schemes to achieve the intended security level while keeping a reasonably small key size, using polar code.

However, the disadvantage though is that the information rate is low. We can solve this issue by putting some information data in the error pattern, as shown by Biswas and Sendrier [48]. That is, some additional information bits are mapped into an error vector to be added in the encryption phase. Furthermore, future work in attempting to transfer our scheme to obtain a signature scheme may also be an interesting problem.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

We would like to thank Dr. Vlad Dragoi for insightful discussions. This work was supported in part by the National Natural Science Foundation of China (Grant no. 61702124), Qatar Foundation (Grant no. NPRP8-2158-1-423), and Guang Dong Provincial Natural Science Foundation (Grant no. 2018A030310071).

References

[1] P. W. Shor, “Polynomial time algorithms for discrete logarithms and factoring on a quantum computer,” in *Lecture Notes in Computer Science*, p. 289, Springer, Berlin, Germany, 1994.

[2] L. K. Grover, “A fast quantum mechanical algorithm for database search,” in *Proceedings of the 28th Annual ACM Symposium on Theory of Computing-STOC*, ACM Press, Philadelphia, PA, USA, May 1996.

[3] NIST, *Post Quantum Crypto Project (2017)*, NIST, Gaithersburg, MD, USA, 2017.

[4] R. J. McEliece, “A public-key cryptosystem based on algebraic coding theory,” *Jet Propulsion Laboratory DSN Progress Report*, vol. 4244, pp. 114–116, 1978.

[5] H. Niederreiter, “Knapsack-type cryptosystems and algebraic coding theory,” *Problems Control Inform Theory*, vol. 15, no. 2, pp. 159–166, 1986.

[6] H. Janwa and O. Moreno, “Mceliece public key cryptosystems using algebraic-geometric codes,” *Designs, Codes and Cryptography*, vol. 8, no. 3, 1996.

[7] T. P. Berger and P. Loidreau, “How to mask the structure of codes for a cryptographic use,” *Designs, Codes and Cryptography*, vol. 35, no. 1, pp. 63–79, 2005.

[8] C. Wieschebrink, “Two NP-complete problems in coding theory with an application in code based cryptography,” in

Proceedings of the 2006 IEEE International Symposium on Information Theory, IEEE, Seattle, WA, USA, July 2006.

[9] C. Monico, J. Rosenthal, and A. Shokrollahi, "Using low density parity check codes in the McEliece cryptosystem," in *Proceedings of the 2000 IEEE International Symposium on Information Theory (Cat. No. 00CH37060)*, IEEE, Sorrento, Italy, June 2000.

[10] P. Gaborit, "Shorter keys for code based cryptography," in *Proceedings of the 2005 International Workshop on Coding and Cryptography (WCC 2005)*, pp. 81–91, Bergen, Norway, March 2005.

[11] V. M. Sidelnikov, "A public-key cryptosystem based on binary reed-muller codes," *Discrete Mathematics and Applications*, vol. 4, no. 3, 1994.

[12] P. Gaborit, G. Murat, O. Ruatta, and G. Zémor, "Low rank parity check codes and their application to cryptography," in *Proceedings of the Workshop on Coding and Cryptography WCC*, vol. 2013, Bergen, Norway, April 2013.

[13] M. Baldi and G. F. Chiaraluce, "Cryptanalysis of a new instance of McEliece cryptosystem based on QC-LDPC codes," in *Proceedings of the 2007 IEEE International Symposium on Information Theory*, IEEE, Nice, France, June 2007.

[14] A. Couvreur, P. Gaborit, V. Gauthier-Umaña, A. Otmani, and J.-P. Tillich, "Distinguisher-based attacks on public-key cryptosystems using reed-solomon codes," *Designs, Codes and Cryptography*, vol. 73, no. 2, pp. 641–666, 2014.

[15] A. Couvreur, C. I. Marquez, and R. Pellikaan, "A polynomial time attack against algebraic geometry code based public key cryptosystems," in *Proceedings of the 2014 IEEE International Symposium on Information Theory*, IEEE, Honolulu, HI, USA, June 2014.

[16] J. C. Faugere, A. Otmani, L. Perret, and J. P. Tillich, "Algebraic cryptanalysis of McEliece variants with compact keys," in *Advances in Cryptology–EUROCRYPT 2010*, pp. 279–298, Springer, Berlin, Germany, 2010.

[17] L. Minder and A. Shokrollahi, "Cryptanalysis of the sidelnikov cryptosystem," in *Advances in Cryptology-EUROCRYPT 2007*, pp. 347–360, Springer, Berlin, Germany, 2007.

[18] Y. Wang, "Quantum resistant random linear code based public key encryption scheme RLCE," in *Proceedings of the 2016 IEEE International Symposium on Information Theory (ISIT)*, IEEE, Barcelona, Spain, July 2016.

[19] Y. Wang, *Rlce-key Encapsulation Mechanism*, NIST, Gaithersburg, MD, USA, 2017.

[20] A. Couvreur, M. Lequesne, and J. P. Tillich, "Recovering short secret keys of RLCE in polynomial time," in *Post-quantum Cryptography*, pp. 133–152, Springer International Publishing, Berlin, Germany, 2019.

[21] E. Arikan, "Channel polarization: a method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Transactions on Information Theory*, vol. 55, no. 7, pp. 3051–3073, 2009.

[22] R. Hooshmand, M. K. Shooshtari, T. Eghlidos, and M. R. Aref, "Reducing the key length of mceliece cryptosystem using polar codes," in *Proceedings of the 2014 11th International ISC Conference on Information Security and Cryptology*, IEEE, Tehran, Iran, September 2014.

[23] S. R. Shrestha and Y. S. Kim, "New McEliece cryptosystem based on polar codes as a candidate for post-quantum cryptography," in *Proceedings of the 2014 14th International Symposium on Communications and Information Technologies (ISCIT)*, IEEE, Incheon, Korea, September 2014.

[24] M. Bardet, J. Chaulet, V. Drăgoi, A. Otmani, and J. P. Tillich, "Cryptanalysis of the McEliece public key cryptosystem based on polar codes," in *Post-quantum Cryptography*, pp. 118–143, Springer International Publishing, Berlin, Germany, 2016.

[25] Y. Li, R. H. Deng, and X. Wang, "On the equivalence of mceliece's and niederreiter's public-key cryptosystems," *IEEE Transactions on Information Theory*, vol. 40, no. 1, pp. 271–273, 1994.

[26] D. J. Bernstein, T. Lange, and C. Peters, "Attacking and defending the McEliece cryptosystem," in *Post-quantum Cryptography*, pp. 31–46, Springer, Berlin, Germany, 2008.

[27] H. Mahdavifar, M. El-Khamy, J. Lee, and I. Kang, "Performance limits and practical decoding of interleaved reed-solomon polar concatenated codes," *IEEE Transactions on Communications*, vol. 62, no. 5, pp. 1406–1417, 2014.

[28] V. Dragoi, *Algebraic approach for the study of algorithmic problems coming from cryptography and the theory of error correcting codes*, Ph.D. thesis, University of Rouen, Mont-Saint-Aignan, France, 2017.

[29] E. Eaton, M. Lequesne, A. Parent, and N. Sendrier, "QC-MDPC: a timing attack and a CCA2 KEM," in *Post-quantum Cryptography*, pp. 47–76, Springer International Publishing, Berlin, Germany, 2018.

[30] I. Cascudo, R. Cramer, D. Mirandola, and G. Zemor, "Squares of random linear codes," *IEEE Transactions on Information Theory*, vol. 61, no. 3, pp. 1159–1173, 2015.

[31] C. T. Gueye and E. H. M. Mboup, "Secure cryptographic scheme based on modified reed muller codes," *International Journal of Security and Its Applications*, vol. 7, no. 3, pp. 55–64, 2013.

[32] A. Otmani and H. T. Kalachi, "Square code attack on a modified sidelnikov cryptosystem," in *Lecture Notes in Computer Science*, pp. 173–183, Springer International Publishing, Berlin, Germany, 2015.

[33] C. Wieschebrink, "Cryptanalysis of the niederreiter public key scheme based on GRS subcodes," in *Post-quantum Cryptography*, pp. 61–72, Springer, Berlin, Germany, 2010.

[34] V. Drăgoi, V. Beiu, and D. Bucerzan, "Vulnerabilities of the McEliece variants based on polar codes," in *Innovative Security Solutions for Information Technology and Communications*, pp. 376–390, Springer International Publishing, Berlin, Germany, 2019.

[35] E. Petrank and R. M. Roth, "Is code equivalence easy to decide?," *IEEE Transactions on Information Theory*, vol. 43, no. 5, pp. 1602–1604, 1997.

[36] N. Sendrier, "Finding the permutation between equivalent linear codes: the support splitting algorithm," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1193–1203, 2000.

[37] E. Prange, "The use of information sets in decoding cyclic codes," *IEEE Transactions on Information Theory*, vol. 8, no. 5, pp. 5–9, 1962.

[38] J. Stern, "A method for finding codewords of small weight," in *Coding Theory and Applications*, pp. 106–113, Springer-Verlag, Berlin, Germany, 1989.

[39] C. Peters, "Information-set decoding for linear codes over f q," in *Post-quantum Cryptography*, pp. 81–94, Springer, Berlin, Germany, 2010.

[40] A. May and I. Ozerov, "On computing nearest neighbors with applications to decoding of binary linear codes," in *Advances in Cryptology–EUROCRYPT 2015*, pp. 203–228, Springer, Berlin, Germany, 2015.

[41] M. Albrecht, C. Cid, K. G. Paterson, C. J. Tjhai, and M. Tomlinson, *NTS-KEM*, NIST, Gaithersburg, MA, USA, 2019.

[42] D. J. Bernstein, T. Chou, T. Lange et al., *Classic Mceliece: Conservative Code-Based Cryptography*, NIST, Gaithersburg, MA, USA, 2019.

[43] D. Boneh, O. Dagdelen, M. Fischlin, A. Lehmann, C. Schaffner, and M. Zhandry, "Random oracles in a quantum world," in *International Conference on the Theory and Application of Cryptology and Information Security*, pp. 41–69, Springer, Berlin, Germany, 2011.

[44] Q. Guo, T. Johansson, and P. Stankovski, "A key recovery attack on mdpc with cca security using decoding errors," in *International Conference on the Theory and Application of Cryptology and Information Security*, pp. 789–815, Springer, Berlin, Germany, 2016.

[45] A. Nilsson, T. Johansson, and P. S. Wagner, "Error amplification in code-based cryptography," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 1, pp. 238–258, 2019.

[46] N. Aragon, P. Barreto, S. Bettaieb et al., *Bike: Bit Flipping Key Encapsulation*, NIST, Gaithersburg, MD, USA, 2019.

[47] M. Baldi, A. Barenghi, F. Chiaraluce, G. Pelosi, and P. Santini, *Ledacrypt*, NIST, Gaithersburg, MD, USA, 2019.

[48] B. Biswas and N. Sendrier, "McEliece cryptosystem implementation: theory and practice," in *Post-quantum Cryptography*, pp. 47–62, Springer, Berlin, Germany, 2008.