IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 11, 2020, accepted February 22, 2020, date of publication February 27, 2020, date of current version March 11, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2976753

A Centralized Key Management Scheme Based on McEliece PKC for Space Network

JIE LIU[1,3], XIAOJUN TONG[1], ZHU WANG[2], MIAO ZHANG[1], AND JING MA[4]

[1]School of Computer Science and Technology, Harbin Institute of Technology at Weihai, Weihai 264209, China
[2]School of Information and Electrical Engineering, Harbin Institute of Technology at Weihai, Weihai 264209, China
[3]School of Rongcheng, Harbin University of Science and Technology, Rongcheng 264300, China
[4]Science and Technology on Information Assurance Laboratory, Beijing 100072, China

Corresponding author: Xiaojun Tong (tong_xiaojun@163.com)

This work was supported in part by project ZR2019MF054 supported Shandong Provincial Natural Science Foundation, in part by the Foundation of Science and Technology on Information Assurance Laboratory under Grant KJ-17-004, in part by the Equip Pre-Research Projects of 2018 under the Foundation of China Academy of Space Technology under Grant WT-TXYY/WLZDFHJY003, in part by the Fundamental Research Funds for the Central Universities under Grant HIT.NSRIF.2020099, in part by the National Natural Science Foundation of China under Grant 61902091, in part by the 2017 Weihai University Co-Construction Project, and in part by the Engineering Technology and Research Center of Weihai Information Security.

ABSTRACT Constrained by the limited resource, high-latency and high bit error rate, the existing group key management schemes for the space network are inefficient. To solve this problem, we proposed a centralized and identity-based key management scheme by using McEliece public-key cryptosystem (PKC). In this scheme, the node identity is used as the parameter to generate the public key. Therefore, the authentication can be embedded into the verification of the public key without needing the PKI. The group key is distributed with the protection of the public key so that the group key management scheme can be implemented safely. Furthermore, the McEliece public-key cryptosystem can resist the quantum attack and provide error correction capacity. It improves the efficiency of the group key distribution over the noisy channel. The proposed key management scheme is simulated on OPNET. The security of public-key generation, forward secrecy, backward secrecy and performance are analyzed. The results show that our scheme can provide confidentiality, integrity, authentication, non-repudiation, failure tolerance and error correction. In addition, the computation overhead and rounds of interaction are lower than former work.

INDEX TERMS Centralized key management, failure tolerance, McEliece PKC, space network, verifiable public key.

I. INTRODUCTION

The space network (SN) [1] consists of various satellite constellations and control centers. Due to the openness and complexity of space network, its key management faces great challenges. To ensure integrity and confidentiality, the cryptosystem is widely used in SN communication. Various key management schemes are designed to support these cryptosystems. They provide identification, authentication, access control, initialization and update of the group key [2] for space communication.

The key management schemes can be summarized into three categories by their characteristic [3]: centralized key management, distributed key management and decentralized key management. In centralized key management, a centralized key distribution center (KDC) is employed for controlling the key management. It is fully trusted and responsible for generation, distribution and update of the key. The centralized key management schemes take a high efficiency [4]. The major drawback is that it needs high storage overhead and may cause a single-point failure. In the distributed key management schemes, there is no central controller. The group key is generated through a threshold secret sharing scheme (TSSS). When the group key needs to be updated, all the members publish their secret shares to other members. Each member generates or recovers the group key by the received secret shares. The advantage of the distributed key management is that there is no single-point failure and key escrow. But the generation of group key takes a multi-round of interaction. In the decentralized key management, the group

The associate editor coordinating the review of this manuscript and approving it for publication was Chien-Ming Chen.

is split into several subgroups or multiple clusters [5]. The cluster head is used to manage each subgroup or cluster. The failure of one cluster head will not affect the other clusters. However, the communication between different clusters needs the delivery of the cluster head, which increases the delay.

Recently, many centralized key management schemes based on LKH are presented. In [6], the consumption of rekeying is reduced by improving the original LKH scheme. A modification of the LKH-based key management scheme is proposed in [7]. A one-encryption key and multi-decryption key protocol is used to generate and distribute the group key. However, the computation overhead and interaction round of initialization and rekeying are higher than other schemes. In the key management presented in [8], the public cryptosystem with public key infrastructure (PKI) is used to provide authentication for the GNSS system. It is inefficient in large-scale space network. The key management based on multi-servers is presented in [9] to avoid single-point failure. But it doesn't provide identification. The certificateless group key management scheme is present in [10]. However, it needs many rounds of interaction to verify and distribute the group key, which costs a long time. The decentralized key management scheme is usually put forward in large-scale networks, Ad hoc networks and Internet of Things (IoT). The hierarchical key management scheme is presented to reduce the computation overhead and bandwidth overhead of rekeying in the satellite networks [11]. The efficiency is improved by extending the tree-based key management to multi-cluster. But the communication between different hierarchies increases the delay and node load. Many ECC-based distributed key management schemes are presented by the researchers. In [12], the ECC cryptosystem is used to provide integrity, confidentiality and verification without the requirement of PKI. Another ECC-based key management scheme is presented by Hsiao *et al.* in 2019 [13]. But it needs the support of PKI. A certificateless public-key cryptosystem based on ECC is employed to design the key management scheme in [14]. In this scheme, the nodes in the wireless sensor network (WSN) are divided into different clusters. Four kinds of keys are used to guarantee the security of the communication. The flaw is that it needs multiple rounds of interaction, which leads to greater delay. In 2018, Ali *et al*. proposed a distributed key management based on ECC [15]. However, it takes a huge message overhead and interaction round in the initialization phase. More serious is that it cannot identify the malicious node. Harbi *et al*. proposed an ECC-based key management for the IoT [16]. However, it needs multiple rounds to authenticate the new nodes. Therefore, most of the key management schemes used in WSN or IoT are unsuitable for space network since the multi-round of interactivity is inefficient in the noisy and delayed environment.

From the above, we can see that the centralized key management scheme either does not provide authentication or provides authentication with the support of PKI. The distributed key management scheme takes many rounds of interaction which leads to a large delay. The decentralized key management scheme increases the transfer delay and node load. All of them are inefficient in long-distance and large-scale space networks. To design a key management scheme that can meet the above requirement, the McEliece cryptosystem is introduced. The McEliece cryptosystem is a public-key cryptosystem based on algebraic coding theory, which provides error correction and resistance against quantum attacks. The application of the original McEliece cryptosystem is constrained by its large public key size. In [17], the McEliece cryptosystem based on a quasi-cyclic moderate density parity check (QC-MDPC) is proposed to against all of the known attacks while decreasing the key size. Another advantage is that it is a lightweight cryptosystem [18] and secure than QC-LDPC with the same code length. Thus, an identity-based group key management scheme is designed by using the McEliece cryptosystem in this work, which provides authentication, secure group key management and verifiable public key management without a certificate.

The contributions of this paper are shown as follows: (I) A verifiable public key generation scheme, which is based on QC-MDPC code and the node identity, is designed. The public key is verified without the requirement of a certificate. What's more, it can resist the quantum attack and provide error correction capacity. (II) A group key management scheme is proposed by using the Hash function. The method of generation, distribution and update of the group key is designed. (III) The proposed scheme is simulated and its security is analyzed. (IV) The proposed scheme is compared with other schemes.

The rest of this paper is organized as below: In section 2, the QC-MDPC code-based McEliece cryptosystem is described. In Section 3, the identity-based and verifiable public key management scheme is presented. The generation and distribution of the group key are designed. The procedure of the KDC election is described. In Section 4, the security is analyzed. The simulation test and the performance are analyzed in section 5. The paper is concluded in Section 6.

II. PRELIMINARY

A. CONSTRUCTION OF QC-MDPC CODE

We gather here a few basic definitions which are used in this paper.

Definition 1 (Quasi-Cyclic Code): An (n, r)-linear code is quasi-cyclic (QC) if there is some integer n_0 such that every cyclic shift of a codeword by n_0 places is again a codeword.

Definition 2 (LDPC/MDPC Codes): An (n, r, w)-LDPC or MDPC code is a linear code of length n, codimension r which admits a parity-check matrix of constant row weight w.

The MDPC McEliece cryptosystem was proposed to resist the attacks caused by the sparse of LDPC code. The difference is that the MDPC is denser than LDPC, which the row weight is approximately $O(\sqrt{n \log n})$. When these codes

are also quasi-cyclic, we call them QC-MDPC or QC-LDPC codes.

To achieve a flexibility code rate, we usually construct the parity check H as $H = [H_0|H_1|\cdots|H_{m-1}] \in F_2^{r\times n}$, where $n = mr$ for $m, r \in Z^*$, r is the block size of H_i. Each H_i has row weight w_i so that $w = \sum_{i=0}^{m-1} w_i$. Thus the generator matrix G can be computed by formula (1):

$$G = \begin{bmatrix} & (H_{m-1}^{-1}H_0)^T \\ & (H_{m-1}^{-1}H_1)^T \\ I_{(m-1)\times r} & \vdots \\ & (H_{m-1}^{-1}H_{m-2})^T \end{bmatrix} \tag{1}$$

The construction of (n, r, w)-QC-MDPC code is as following:

Step1: Choose a code length n and block size r. Generate the vector $h_i \in F_2^r$ with length r and weight w/m at random, where $m = n/r$ is the number of blocks.

Step2: Generate the circulant sub-matrix $H_i \in F_2^{r\times r}$, in which the first row is defined by vector h_i. The other $r-1$ rows of H_i are obtained from the $r-1$ cyclic shifts of h_i.

Step3: Obtain the check matrix $H = [H_0|H_1|\cdots|H_{m-1}]$.

B. MCELIECE CRYPTOSYSTEM BASED ON QC-MDPC CODE

The McEliece cryptosystem based on QC-MDPC code consists of key generation, encryption algorithm and decryption algorithm.

1) KEY GENERATION

The keys include public key (G, t) and private key H. The private key is a parity-check matrix $H \in F_2^{r\times n}$ of a t-error-correcting (n, r, w)-QC-MDPC code. The public key is the generator matrix $G \in F_2^{(n-r)\times n}$ in a row reduced echelon form. It can be computed by formula (1).

2) ENCRYPTION

Firstly, An error vector $e \in F_2^n$ of Hamming weight $wt(e) \le t$ is chosen randomly. Here, $t = n/w$ is the error correction capacity, $wt(e)$ is the Hamming weight of e. Then the ciphertext can be obtained by $C = MG+e$, where M is the plaintext.

3) DECRYPTION

Compute MG by using the decoding algorithm. Then extract the plaintext M from the first $n-r$ columns of MG.

III. THE CENTRALIZED KEY MANAGEMENT SCHEME BASED ON MCELIECE PKC

As mentioned above, a certificateless key management scheme that provides authentication, integrality, public key management and group key management is required eagerly in the space networks. Furthermore, it is better tolerant of single-point failure. In this section, an identity-based key management scheme based on McEliece PKC is presented, which meets the requirements all above. It consists of the establishment of the public key and group key, rekeying for nodes' joint, rekeying for nodes' leaving and KDC election algorithm. The flow chart of the proposed key management is shown in FIGURE. 1.

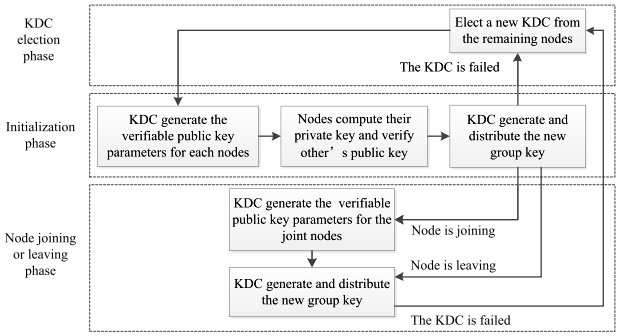


FIGURE 1. The flow chart of the proposed key management scheme.

As shown in FIGURE. 1, the flow chart of proposed key management for satellite network consists of The establishment of keys, rekeying for nodes' joint or leaving and KDC election. The establishment of keys includes the design of verifiable McEliece public key generation scheme, the generation of the public key and the generation of the group key. The detail is as below.

A. DESIGN OF VERIFIABLE MCELIECE PUBLIC KEY GENERATION SCHEME

As the QC-MDPC code-based cryptosystem is lightweight, it is suitable to be used in the space network for key management. Thus a QC-MDPC code-based public key generation scheme is designed, which embeds identity in the verification of public key so that the space nodes can authenticate each other by computing their public key. The detail is as following.

1) GENERATION OF THE PUBLIC KEY AND PRIVATE KEY FOR KDC

The procedure of public key generation and private key generation for KDC is as below:

Step1: The KDC randomly chooses vector $h_{kdc_0}, h_{kdc_1}, \cdots, h_{kdc_{m-1}}$ with weight w/m. Here, w is the weight of (n, r, w)-QC-MDPC code, m is the number of sub-matrix.

Step2: The KDC constructs a check matrix $H_{kdc} = \left[H_{kdc_0}H_{kdc_1}\ldots H_{kdc_{m-1}}\right]$ by the method in subsection A of section 2.

Step3: The KDC computes the generator matrix G_{kdc} by using formula (1) and publishes it.

Without loss of generality, we set $m = 2$ and $w = \sqrt{n\log n}$ in this work.

2) GENERATION OF THE PUBLIC KEY AND PRIVATE KEY FOR SPACE NODES

The generation and verification of the public-private key-pairs for space nodes are as below:

Step 1: For each node i, the KDC randomly chooses a vector s_i with a weight of $\sqrt[3]{w/2}$ and publishes it.

Step 2: Node i generates its circulant sub-matrix S_i by s_i and chooses vector $h_{i,2}$ and $h_{i,3}$. Here, $wt(h_{i,2}) = \sqrt[3]{w/2}$, $wt(h_{i,3}) = w/2$. Then generate the circulant sub-matrixes $H_{i,2}$ and $H_{i,3}$. Since the number of sub-matrix is $m = 2$, the Hamming weight of left and right sub-matrix in H should less than or equal to $w/2$.

Step 3: Node i maps its ID to a binary sequence h_{ID_i} by using formula (2).

$$h_{ID_i} = hash(ID_i) \tag{2}$$

Here, the collision-free hash function is $hash : \{0, 1\}^* \rightarrow \{0, 1\}^*$. Then h_{ID_i} is transformed to a binary sequence hb_{ID_i} so that $wt(hb_{ID_i}) = \sqrt[3]{w/2}$. The right-hand side of hb_{ID_i} is filed with "0" to generate the vector $h_{i,1}$. The matrix $H_{i,1}$ is generated by shifting $h_{i,1}$ cyclically, which is shown in formula (3):

$$H_{i,1} = \begin{bmatrix} h_{i,1_1} & \cdots & h_{i,1_r} \\ \vdots & \ddots & \vdots \\ h_{i,1_2} & \cdots & h_{i,1_1} \end{bmatrix} \tag{3}$$

Step 4: Node i computes its check matrix H_i by formula (4). The private key is H_i. The vector $h_{i,2}$ and $h_{i,3}$ are kept secretly.

$$H_i = \left[H_{i,3} | H_{i,2} H_{i,1} S_i\right] \tag{4}$$

Step 5: Node i computes a witness value $R_i = H_{i,2}^{-1} H_{i,3}$ and publishes the first row r_i. Then node i obtains its public key G_i as formula (5):

$$G_i = \left[I | ((H_{i,2} H_{i,1} S_i)^{-1} H_{i,3})^T\right] = \left[I | (S_i^{-1} H_{i,1}^{-1} H_{i,2}^{-1} H_{i,3})^T\right] \tag{5}$$

Step 6: KDC and other nodes compute and verify the public key of node i by formula (6):

$$G_{i_{verify}} = \left[I | (S_i^{-1} H_{i,1}^{-1} R_i)^T\right] \tag{6}$$

When KDC and other nodes need to transmit secret to node i, they encrypt the secret by using $G_{i_{verify}}$. It's obvious that $G_{i_{verify}} = G_i$. Note that S_i, R_i and all of the sub-matrixes $H_{i,j}$ are overcoming probability invertible since they are circulant.

3) ENCRYPTION AND DECRYPTION ALGORITHM

The message can be encrypted by the McEliece public key through formula (7):

$$C = MG + e \tag{7}$$

where M is the message, e is a random vector with Hamming weight $wt(e) \leq t$ and $t = n/w$. When the ciphertext is received by nodes or KDC, they obtain MG by using the decoding algorithm and extract the plaintext M from the first $n - r$ columns of MG.

B. GENERATION AND DISTRIBUTION OF THE PUBLIC KEY AND GROUP KEY

Generate and distribute the public key and group key is the main work in the initialization stage of the proposed key management scheme. The parameters n, r, w, m and t need to be set before the proposed key management scheme starts to work. The symbols used in the following algorithms are described below: $s_i, h_{i,2}, h_{i,3}$ are random vectors, ID_i is the identity of node i, R_i is the witness value, G_i is the public key of node i, H_i is the private key of node i, $G_{i_{verify}}$ is the verifiable public key of node i, DEK is the group key, a_1 is a random vector, DEK_{pk_i} is the encrypted group key which using the public key of node i. The procedure is described as Algorithm 1.

Algorithm 1 Generation and Distribution of the Public Key and Group Key

Input: $n, r, w, m, t, s_i, h_{i,2}, h_{i,3}, ID_i, a_1$
Output: $R_i, G_i, H_i, G_{i_{verify}}, DEK$
/* Generate the verifiable public key and private key for all of the nodes. Generate and distribute the group key */
1 KDC: Publishes (n, r, w, m, t);
2 for $i \leftarrow 1$ to N
3 All nodes: Computes and stores $G_{i_{verify}}$ by formula (6);
4 KDC: Computes $DEK \leftarrow hash(a_1)$
5 KDC: Computes DEK_{pk_i};
6 KDC: Sends DEK_{pk_i} to node i;
7 end for

The receiver decrypts DEK_{pk_i} by using its private key and obtains the group key DEK.

C. REKEYING FOR NODES' JOINT

When a new node is jointing, it sends a request to KDC. After KDC and other members' verification, the rekeying is triggered. The procedure is described as Algorithm 2.

The rekeying is usually periodic. When a request for jointing or leaving is sent to KDC, the operation of rekeying is triggered and the timer is reset.

D. REKEYING FOR NODES' LEAVING

If a node is leaving, it sends a request to KDC. Then the rekeying is triggered. The procedure is as below:

Step 1: KDC removes the public key of the leaving node from its table.

Step 2: KDC randomly chooses a vector a_1 and generates the new group key by $DEK = hash(a_1)$.

Step 3: KDC encrypts the new group key with each nodes' public key and sends it to them.

Step 4: Each node decrypts the new group key by its private key and updates the group key.

Since the public key of any node is re-computed when it joints to a new group or returns to the original group, there is no need to revoke its public key.

Algorithm 2 Rekeying for Nodes' Joint

Input: $n, r, w, m, t, s_i, h_{i,2}, h_{i,3}, ID_i, a_1$
Output: $R_i, G_i, H_i, G_{i_{verify}}, DEK$
/* Node i generates its public key and private key. KDC and other nodes verify the public key of node i. KDC generates and distributes the new group key */
1 Node i: Sends a request to KDC;
2 KDC: Randomly chooses a vector s_i for node i and publishes it;
3 Node i: Generates $H_{i,2} \leftarrow h_{i,2}$ and $H_{i,3} \leftarrow h_{i,3}$, computes H_i and G_i by formula (4) and (5).
4 Node i: Computes $R_i = H_{i,2}^{-1} H_{i,3}$ and publishes the first row r_i;
5 KDC and other nodes: Compute and store $G_{i_{verify}}$;
6 for $i \leftarrow 1$ to $_N$
7 KDC: Computes $DEK \leftarrow hash(a_1)$;
8 KDC: Computes DEK_{pk_i};
9 KDC: Sends DEK_{pk_i} to node i;
10 end for

E. KDC ELECTION ALGORITHM

The proposed scheme is tolerant for the single-point failure by using the KDC election mechanism. When the KDC is failed, the group member can elect a new KDC. The detail is as below:

Step 1: One of the remaining node i computes a challenge message $(ID_i, (ID_i)_{pk_i})_{E_{DEK}}$ and sends it to other nodes in the group. Here, $(ID_i)_{pk_i}$ is the encryption of ID_i by using the public key pk_i, $(x)_{E_{DEK}}$ denotes that encrypt x by using the group key DEK.

Step 2: Each of the nodes except node i decrypts the challenge message by using $((ID_i, (ID_i)_{pk_i})_{E_{DEK}})_{D_{DEK}}$. Then obtains ID_i and $(ID_i)_{pk_i}$. Here, $(x)_{D_{DEK}}$ denotes decrypt x by using the group key DEK.

Step 3: Each of the nodes except node i encrypts ID_i by using the verifiable public key of node i. They will obtain an encrypted value ID_i'.

Step 4: Each of the nodes except node i compares ID_i' with $(ID_i)_{pk_i}$ which is obtained in step 2. If the identity of the sender equals to ID_i and $ID_i' = (ID_i)_{pk_i}$, keeps silent. Otherwise, computes $num_{disagree} = num_{disagree} + 1$ and broadcasts $num_{disagree}$.

Step 5: All of the nodes check that if $num_{disagree} \leq N/3$. If it is true, update its KDC to node i. Else, start the next round of the election.

The procedure is described as Algorithm 3.

By using the KDC election algorithm, the new KDC can be elected so that the single point failure can be avoid absolutely.

IV. SECURITY ANALYSIS

In this section, the security of McEliece public key generation scheme, backward and forward secrecy and collusion attack are analyzed. The security analysis of McEliece public key generation scheme consists of the key recover attack, the information set attack and the GJS attack.

Algorithm 3 KDC Election Algorithm

Input: $ID_i, (ID_i)_{pk_i}$
Output: node i
/* elect the new KDC from the remaining nodes */
/* $(ID_i)_{pk_i}$ is the encrypted identity of node i by using its public key, $(x)_{E_{DEK}}$ is the encryption of x with group key DEK, $(x)_{D_{DEK}}$ is the decryption of x with group key DEK, ID_i' is the encrypted identity of node i by using its verifiable public key. */
/*$num_{disagree}$ is the number of disagreement, N is the total number of nodes */
1 for $i \leftarrow 1$ to N
2 Node i: Sends a challenge message $(ID_i, (ID_i)_{pk_i})_{E_{DEK}}$;
3 for $j \leftarrow 1$ to N and $j \neq i$
4 Node j: Computes $((ID_i, (ID_i)_{pk_i})_{E_{DEK}})_{D_{DEK}}$ and $ID_i' \leftarrow (ID_i)_{G_{i_{verify}}}$;
5 if the identity of the sender equals to ID_i and $ID_i' = (ID_i)_{pk_i}$, then node j keeps silent;
6 else
7 Node j : $num_{disagree} = num_{disagree} + 1$;
8 end if
9 end for
10 if $num_{disagree} \leq N/3$, then node i is the new KDC;
11 break;
12 else continue;
13 end if
14 end for

A. KEY RECOVERY ATTACK

In this work, we designed a verifiable public key generation scheme by using McEliece cryptosystem. Its security equals the work factor that obtains the private key from the public information. The public information for each node i includes s_i, r_i and $h_{i,1}$.

Theorem 1: The probability that deduces the private key from published parameters is negligible if the block size of the generator sub-matrix is larger than 512 bits.

Proof: As the security of McEliece cryptosystem has been proved in [17], [18]. Thus the security of public key generation scheme equals the security of the private key. There are two approaches to obtain the private key H_i: guessing H_i directly or computing the secret vector $h_{i,2}$ and $h_{i,3}$ from known information. For the first method, the probability that guessing H_i correctly can be obtained by formula (8):

$$P_H = \frac{1}{\binom{n}{w}} \tag{8}$$

When $r = 160$, probability P_H is about $2^{-198.46}$. The probability P_H increases to $2^{-471.21}$ when $r = 512$. The work fact of this method is increasing exponentially with the increase of block size r.

For the second method, the attacker needs to work out $H_{i,2}$ and $H_{i,3}$ from $H_{i,2}^{-1} H_{i,3}$. Suppose that $h_{i,2}^{-1} = (x_1, x_2, \cdots, x_r)$,

$h_{i,3} = (y_1, y_2, \cdots, y_r)$, the first row of $H_{i,2}^{-1}H_{i,3}$ is $(z_1, z_2, \cdots, z_r)$. This problem can be transformed to solve the equation set of formula (9):

$$\begin{cases} x_1y_1 + x_2y_r + \cdots + x_ry_2 = z_1 \\ x_ry_2 + x_1y_1 + \cdots + x_{r-1}y_3 = z_1 \\ \vdots \\ x_2y_r + x_3y_{r-1} + \cdots + x_1y_1 = z_1 \\ \vdots \\ x_1y_r + x_2y_{r-1} + \cdots + x_ry_1 = z_r \\ x_ry_1 + x_1y_r + \cdots + x_{r-1}y_2 = z_r \\ \vdots \\ x_2y_{r-1} + x_3y_{r-2} + \cdots + x_1y_r = z_r \end{cases} \tag{9}$$

The addends of the r equations which equal to z_i are shifted cyclically, where $i \in \{1, 2, \cdots, r\}$. Thus we have the formula (10):

$$\begin{cases} x_1y_1 + x_2y_r + \cdots + x_ry_2 = z_1 \\ x_1y_2 + x_2y_1 + \cdots + x_ry_3 = z_2 \\ \vdots \\ x_1y_r + x_2y_{r-1} + \cdots + x_ry_1 = z_r \end{cases} \tag{10}$$

Obviously, the equations have no solution because only z_i is known. Thus the only way to obtain $H_{i,2}$ and $H_{i,3}$ from $H_{i,2}^{-1}H_{i,3}$ is that guessing $H_{i,2}$ and $H_{i,3}$ separately. For a (n, r, w)-QC-MDPC code, $H_{i,2}$ and $H_{i,3}$ are $r \times r$ circulant matrixes. As $m = 2$, $r = n/2$, $w = \sqrt{nlog\,n}$, the probability that guessing $H_{i,2}$ and $H_{i,3}$ can be computed by formula (11):

$$P_h = \frac{1}{\binom{r}{\sqrt[3]{w/2}}\binom{r}{w/2}} \tag{11}$$

When $r = 512$ the probability that successfully guessing $H_{i,2}$ and $H_{i,3}$ is $2^{-177.37}$. The probability will decrease with the increasing of code length. It decreases to $2^{-267.13}$ when $r = 1024$.

Therefore, in both of the two conditions, the work factor to recover the secret key is higher than 2^{128} when $r \geq 512$. The security level is higher than 2^{192} when $r \geq 1024$.

B. INFORMATION SET DECODING (ISD) ATTACK

ISD attack is a critical message recovery attack against McEliece cryptosystem based on QC-MDPC code [19]. The lowest work fact of ISD is shown as formula (12):

$$WF = \frac{\binom{n}{t}}{\binom{n-k-l}{t-p}\binom{k+l}{p}} \tag{12}$$

Here, l and p are parameters which relate to the code length n, $t = n/w$ is the maximum weight of the error vector e. We compute the minimum work fact for different (n, r, w) QC-MDPC codes, where $p = 4$ and $l = 0.013n$. The results are shown in Table 1.

TABLE 1. The ISD work fact for different code length.

Proposed Scheme	ISD work fact
(2048,1024,40)-QC-MDPC with $t = 104$	$2^{81.82}$
(9602,4801,63)-QC-QMDPC with $t = 148$	$2^{130.13}$
(22054,11027,103)-QC-QMDPC with $t = 214$	$2^{192.12}$

From Table 1, we can see that the proposed scheme is secure when the block size $r \geq 1024$. The security is improving with the increase of the code length.

Therefore, to achieve a higher level of security to resist both key recovery attacks and ISD attacks, the block size should larger than 1024 bit.

C. GJS ATTACK

GJS attack is proposed by Guo et al. in [20] to recover the secret key. In this attack, the attacker generated a special error vector set φ_d. All the vectors in set φ_d have t ones which are placed as random pairs with distance d in the first half of the vector. Then the attacker encrypts a message by using the error vector in φ_d and sends it to Bob, Bob's reactions (in terms of decoding success or failure) is collected and analyzed by the attacker to reconstruct the distance spectrum. It is further used to recover the secret key of Bob. The attacker goes through the following steps:

Step1: Compute the distance spectra of all the error vectors $D(e_0), D(e_1), \ldots, D(e_{n_01})$.

Step 2: If a distance d is contained in at least one spectrum $D(e_i)$, increment the counter $b(d)$ by 1.

Step 3: When the decoding of Bob is failed and the distance d is contained in at least one spectrum $D(e_i)$, increment the counter $a(d)$ by 1.

Step 4: If a distance d is contained in the spectrum of the first error vector $D(e_0)$, increment counter $v(d)$ by 1.

Step 5: When the decoding of Bob is failed and the distance d is contained in the spectrum of the first error vector $D(e_0)$, increment the counter $u(d)$ by 1.

After processing all the T ciphertexts, the four vectors are used to construct the distance spectrum of the secret key H. For a QC-MDPC code with 80 bit security ($n_0 = 2, n = 9602, r = 4801, w = 90, t = 84$), it needs 356 million ciphertexts to recover the secret key. This attack is further analyzed in [21] by Baldi et al. The result is that a key pair of QC-MDPC based cryptosystem has a lifetime of $\hat{T}$ before it is broken. The lifetime is computed by formula (13):

$$\hat{T} = \hat{a}DFR^{-1} \tag{13}$$

Here, $\hat{a}$ is the threshold of the expectation $E[a(d)]$, which corresponding to the lower bound of the successful attack. DFR is the decoding failure rate. The GJS attack is simulated for the proposed scheme. The result is that $\hat{a}$ is about 10^4 when $DFR = 10^6$. From formula (13), we can conclude that

the key pairs of the proposed scheme with $(n_0 = 2, n = 2048, r = 1024, w = 40, t = 104)$ QC-MDPC code is secure before it is used to encrypt 10^{10} ciphertexts. That is to say, the proposed scheme is secure against the GJS attack when the public key pair is updated for each 10^{10} ciphertexts.

D. BACKWARD SECRECY AND FORWARD SECRECY

Backward secrecy is used to prevent the new member from decrypting messages exchanged before it joint to the group. In this work, the group key will be updated by the KDC when a new member joins. The parameter a_1 is randomly chosen by KDC to generate the new group key. Suppose that the old group key is DEK_{old} and the new group key is DEK_{new}. The message exchanged before new member joint to group denotes as $E_{DEK_{old}}(M)$. As the new member only has new group key DEK_{new}, The decryption process of the new member is shown in formula (14):

$$M' = D_{DEK_{new}}(E_{DEK_{old}}(M)) \tag{14}$$

Here, $D_{DEK_{new}}(x)$ denotes the decryption of x by using secret key DEK_{new}. Since the properties of hash function guarantee $DEK_{new} \neq DEK_{old}$, the new member cannot decrypt $E_{DEK_{old}}(M)$ correctly. Thus the proposed scheme keeps backward secrecy.

Forward secrecy is used to prevent the leaving member from decrypting the group's communication after its leaving. In this work, the group key will be updated by KDC as soon as the member leaving. The leaving member cannot decrypt the messages because KDC does not send the new group key to it. Therefore, the proposed scheme holds backward secrecy and forward secrecy in key management.

E. COLLUSION ATTACK

Collusion attack is that the evicted members work together and share their pieces of information to compute the new group key. Suppose that the leaving node i collects all other leaving nodes' group keys, which denotes as DEK_k, $k \in \{1, 2, \cdots, v\}$. In this work, the group key is generated by using $DEK = hash(a_1)$. Thus we can obtain formula (15):

$$\begin{cases} DEK_k = hash(a_{1k}),\ k \in \{1, 2, \cdots, v\} \\ DEK_n = hash(a_{1,n}) \end{cases} \tag{15}$$

Here, DEK_n is the new group key, the parameter $a_{1,n}$ is randomly chosen by KDC. From the properties of the hash function, we can obtain formula (16):

$$DEKE_n \neq DEK_k, \quad k \in \{1, 2, \cdots, v\},\ n \neq k \tag{16}$$

The old group keys DEK_k are absolutely useless for the computation of the new group key DEK_n. Thus the proposed scheme can resist collusion attack.

V. SIMULATION TEST AND PERFORMANCE ANALYSIS

In this section, we simulated the proposed scheme on OPNET and compared it with other key management schemes in terms of storage overhead, computation overhead, communication overhead, interaction round and robustness. The compared key management schemes are AGKM [7], the scheme proposed in [10] which denotes as CGKEP, the scheme proposed in [12] which denotes as DEKM, CL-EKM [14], SGKMP [15] and MAKA [16]. Furthermore, we simulated the error correction capacity of the proposed group key distribution scheme over the noisy channel. In order to facilitate comparison, some parameters are given as below: K denotes a unit of key and key material with $1K$ bit, N denotes the number of the group member, l is the cluster number, n_t is the threshold value of secret share scheme.

A. SIMULATION TEST

To estimate the performance of the proposed scheme, we simulated it by using OPNET. In the simulation, the key management model is designed and simulated with different number of nodes. The key management model is shown in FIGURE. 2:

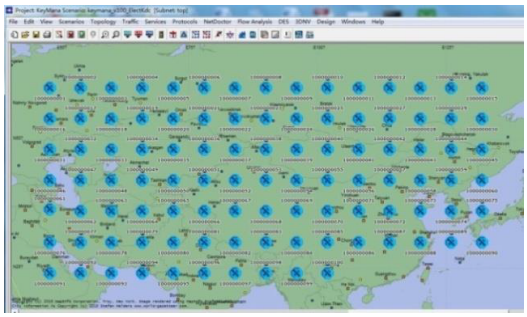


FIGURE 2. The opnet nodul of proposed scheme with 100 nodes.

In FIGURE. 2, the number of satellite nodes is $N = 100$. They communicated with each other by using wireless radio transmitter and receiver. We compared the message cost in different phases. The result is shown in moving_agerage style in FIGURE. 3 so that it can be seen more clearly:

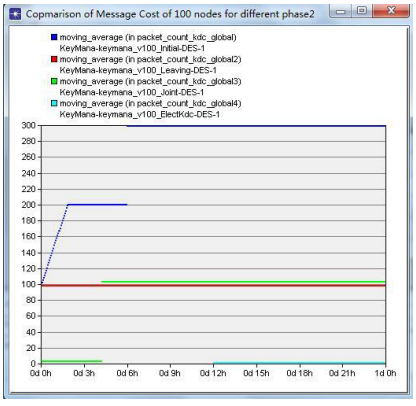


FIGURE 3. The comparison of message cost in different phase with 100 nodes.

In FIGURE. 3, the horizontal axis is the simulation time, the vertical axis is the message number. The message cost in the initialization phase is 300. It is triple of the node number.

TABLE 2. Comparison of storage overhead for different key management schemes.

Scheme	KDC/BS	CH	Member/CM
Proposed scheme	*(N+2)K*	-	$(N+2)K$
Ref [7]	*(2N-1)K*	-	$(log_2 N+1)K$
Ref [10]	-	-	*2K*
Ref [12]	-	-	*4K*
Ref [14]	*(N+l+1)K*	*(N/l+3)K*	*(N/l+3)K*
Ref [15]	*NK*	-	*NK*
Ref [16]	*(N+1)K*	*2K*	*3K*

The message cost of the joint phase and leaving phase are 102 and 100 correspondingly. The message cost of the KDC election is 1 when all of the nodes agree to the new challenger. Furthermore, to estimate the message cost of different network scale, we compared the message cost of 100 nodes and 200 nodes. The result is shown in FIGURE. 4:

As shown in FIGURE. 4, the red color is the message cost of network with 200 nodes while the blue color is for the network with 100 nodes. The message overhead for the initialization, join and leave phases of the 200-node network is twice that for the 100-node network. They are increasing linearly with the increase of network scale. The message cost of different network scales in the KDC election is very low, which equals to 1.

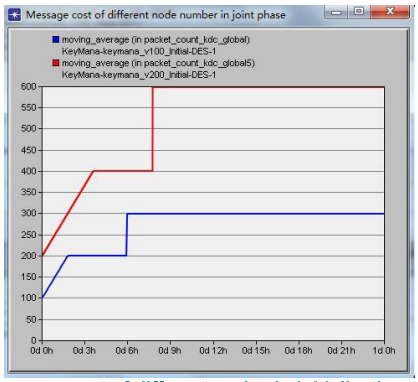
B. STORAGE OVERHEAD

In the proposed scheme, in order to verify the other nodes and elect new KDC, all of the nodes need to store its public-private key pairs, the group key and $(N-1)$ other nodes' public keys. So the storage overhead of the proposed scheme is $(N+2)K$. The comparison is shown in Table 2.

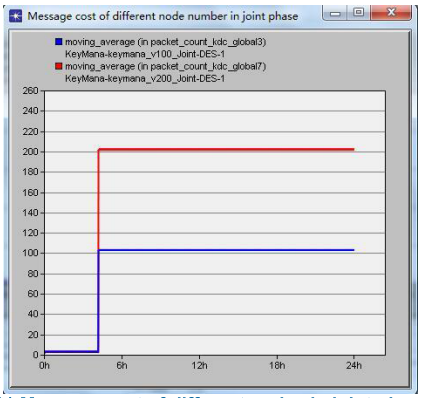
Here, l is the cluster number. As shown in Table 2, the storage overhead of the proposed scheme is about $2N$ while the storage overhead of distributed key management schemes such as CGKEP and DEKM is a constant value. Its storage overhead is approximately twice that of the decentralized key management scheme such as CL-EKM and MAKA. The character of centralized key management determines that its storage overhead is higher than the decentralized key management. Furthermore, the KDC election is used to avoid the single-point failure in the proposed scheme, which increases the storage overhead of the member node. The storage cost of the proposed scheme is lower than the traditional centralized key management schemes such as AGKM.

C. COMPUTATION OVERHEAD

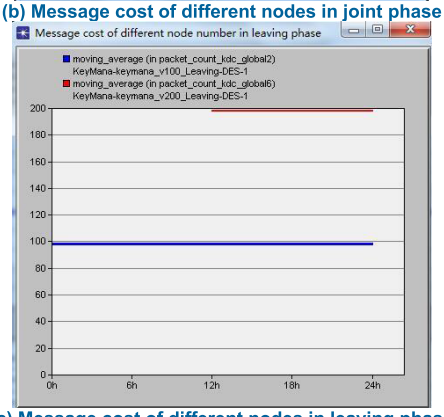
In this work, KDC and each member need $N+2$ operations in the initialization phase. When a new member is jointing, the KDC needs $N+3$ operations while the computation cost of each member is 2. When a member is leaving, KDC generates and encrypts the new group key with each member's public key. Its computation cost is N. The comparison of computation cost is shown in Table 3. Here, l is the cluster number.



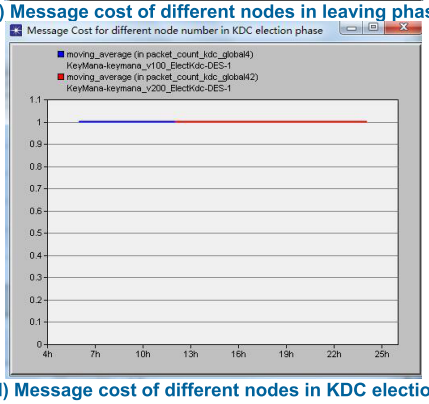
(a) Message cost of different nodes in initialization phase



(b) Message cost of different nodes in joint phase



(c) Message cost of different nodes in leaving phase



(d) Message cost of different nodes in KDC election

FIGURE 4. Comparison of message cost for different network scale in different phase.

The total computation cost of the proposed scheme is compared with other schemes. The number of nodes is changing from 1 to 500. For convenience, the computation cost is

TABLE 3. Comparison of computation overhead for different key management schemes.

Scheme	Initialization			Joining			Leaving		
	KDC/BS	CH	Member/CM	KDC/BS	CH	Member/CM	KDC/BS	CH	Member/CM
Proposed Scheme	N+2	-	N+2	N+3	-	2	N	-	0
Ref [7]	-	-	-	2(N-2)	-	2(N-2)	$3N-4-\log_2 N$	-	$3N-4-\log_2 N$
Ref [10]	N+3	-	N+3	N+2	-	N+3	N+2	-	0
Ref [12]	3N	-	N+1	-	-	3	0	-	0
Ref [14]	3N+1	$20\frac{N}{l}(\frac{N}{l}-1)+2$	$16\frac{N}{l}(\frac{N}{l}-1)+5$	1	-13	$16(\frac{N}{l}-1)+2$	0	$\frac{N}{l}+1$	1
Ref [15]	2	-	2	0	-	N	0	-	N+1
Ref [16]	-	-	-	3N+2l+2	N+4l+5	7	-	-	-

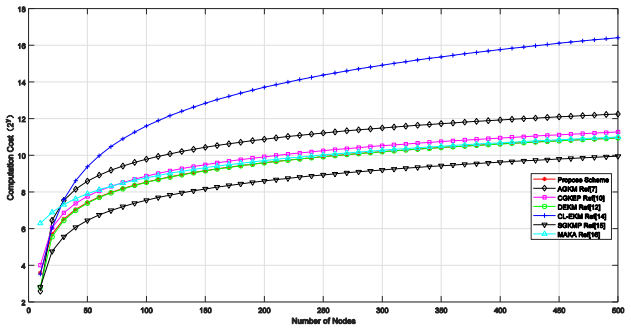


FIGURE 5. Comparison of total computation overhead.

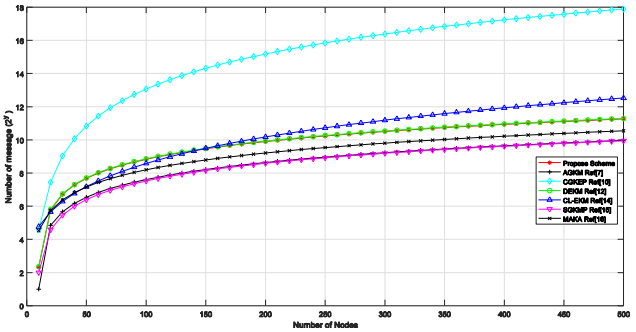


FIGURE 6. Comparison of total message overhead.

TABLE 4. Comparison of the message overhead for different key management schemes.

Scheme	Initialization	Joining	Leaving
Proposed scheme	3N	N+2	N
Ref [7]	2N	$\log_2 N$	$\log_2 N$
Ref [10]	N^2+l	3N+l	l
Ref [12]	3N	3 n_t	-
Ref [14]	$2\frac{N}{l}(\frac{N}{l}-1)+2N+2l$	$2(\frac{N}{l}-1)+5$	$\frac{N}{l}+2$
Ref [15]	2N-1	1	2
Ref [16]	-	3N+2l	-

TABLE 5. Comparison of interaction round and robustness.

Scheme	Interaction Round			No Single-Point Failure
	Initialization	Joining	Leaving	
Proposed scheme	1	1	1	Y
Ref [7]	1	1	1	N
Ref [10]	4	1	1	N
Ref [12]	4	2	0	Y
Ref [14]	5	5	2	Y
Ref [15]	2N-1	2	1	Y
Ref [16]	-	8	-	N

transformed into logarithmic form base 2. The cluster number is $l = 10$. The result is shown in FIGURE. 5.

From Table 3 and FIGURE. 5, we can see that the total computation cost of the proposed scheme is lower than the other schemes except for SGKMP. As the SGKMP only provides group key management, its computation cost is the lowest. In our scheme, the computation used for the rekeying of joining and leaving is less than AGKM, CGKEP and MAKA. Hence, the computation overhead advantage of the proposed scheme will increase with the nodes' joining or leaving.

D. MESSAGE OVERHEAD

In our scheme, the DKC publishes each member's public key parameter and sends the group key to them in initialization. Each member publishes its witness value so that it can be verified. So the total message cost is $3N$. In member joining event, the KDC sends a random vector to the new member and the new member publish its witness value. After verification, the KDC sends the new group key to each member. Its message cost is $N + 2$. When a member is leaving, the KDC only needs to send the new group key to each member. So the message cost is N. We compared the message overhead for different key management scheme. In the comparison, l is the cluster number, n_t is the threshold value of the secret sharing scheme. The result is shown in Table 4.

To validate the comparison in Table 4, we compute the total message for different schemes on the different number of nodes. The range of the node number changes from 0 to 500. Here, we set $l = 10$ and $n_t = 0.7N$. The result is shown in FIGURE. 6.

The Table 4 and FIGURE. 6 imply that the message cost of the proposed scheme is higher than classical centralized schemes but lower than the distributed scheme CGKEP

TABLE 6. qualitative comparison between the different key management schemes.

Scheme	Proposed scheme	Ref [7]	Ref [10]	Ref [12]	Ref [14]	Ref [15]	Ref [16]
Storage Cost scheme	High	High	Low	Low	Middle	High	Middle
Computation Cost	Low	Middle	Middle	Low	High	Low	Low
Message Cost	Middle	Low	High	Middle	High	Low	Low
Authentication	Y	N	Y	N	Y	N	Y
Public Key Management	Y	N	N	N	Y	Y	Y
Group Key Management	Y	Y	Y	Y	Y	Y	Y
No Single-Point Failure	Y	N	N	Y	Y	Y	N
Error Correction	Y	N	N	N	N	N	N

and the decentralized key management schemes CL-EKM. Though the message cost of the decentralized key management scheme such as SGKMP and MAKA is lower than the proposed scheme, they either do not provide authentication or do not provide dynamic key management. The proposed scheme takes a large number of messages to generate and verify the public key of nodes while the traditional centralized schemes such as AGKM only provide group key management. The security of their group key distribution is granted by the extra cryptosystem. So the proposed scheme achieves higher robustness and security by sacrificing the message cost.

E. INTERACTION ROUND AND ROBUSTNESS

The interaction round is another indicator to estimate the key management scheme. In this work, KDC needs to generate the nodes' public key and group key in the initialization stage. In joining event and leaving event, only needs KDC to generate and distribute a new group key. Thus, only one round is needed in each of these stages. Furthermore, the proposed scheme is robust against single-point failures. The comparison is shown in Table 5.

From Table 5 we can see that the proposed scheme provides tolerance of single-point failure with fewer interaction rounds. Though AGKM also has lower interaction rounds, it is unavailable if its KDC is failed. The distributed key management schemes and the decentralized key management schemes need at least 6 rounds to provide tolerance of single-point failure. The large number of interaction round makes them infeasible in the space networks.

F. ERROR CORRECTION CAPACITY

The group key is usually encrypted to guarantee security during its distribution. In this work, we encrypted the group key by using McEilece cryptosystem which based on QC-MDPC code. We simulated the error correction capacity of the proposed group key distribution scheme over the noisy channel, in which the group key is encrypted by the McEliece cryptosystem based on (2048, 1024, 40) QC-MDPC. MD5 is used to map the identity to h_{ID_i}. $H_{i,1}$ is generated by using formula (3). The other parameters such as s_i, $h_{i,2}$, $h_{i,3}$ used in the proposed scheme are generated randomly. The Hamming weight of the generated QC-MDPC code is 40. Furthermore, we compared the bit error rate (BER) of the proposed scheme

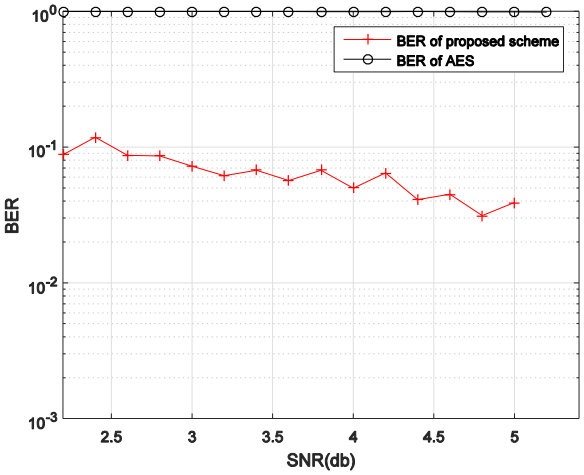


FIGURE 7. Comparison of BER of key distribution schemes over noisy channel.

with the key distribution scheme which encrypted the group key by using AES. The result is shown in FIGURE. 7.

From FIGURE.7, we can see that the BER of the proposed scheme reduced to 0 when the single noise rate (SNR) is higher than 5 while the BER of AES always approximates to 1. This means that the proposed scheme can correct all of the errors in the received keys when $SNR \geq 5$, while the classical key management always obtains error keys. Furthermore, the error correction capacity is increasing with the increase of code length. Thus, the proposed scheme can resistant channel noise and improve the success of key distribution greatly.

A qualitative comparison between the different key management schemes in terms of storage cost, computation cost, message cost, authentication, public key management, group key management, no single-point failure and error correction capacity is provided in Table 6:

From the above, we can conclude that the proposed scheme provides high security and robustness with lower computation cost and interaction round. As a centralized key management scheme, the storage overhead of our scheme is lower than the classical centralized key management schemes such as AGKM. Though its message overhead is higher than the traditional centralized key management schemes, the ability to tolerate single-point failure overcomes the critical flaw of centralized key management schemes. Furthermore, it provides authentication, public key management and group key

management simultaneously while the traditional schemes provide only group key management. The message overhead of the proposed scheme is higher than the decentralized key management schemes such as SGKMP and MAKA. However, the distributed key management schemes and the decentralized key management schemes need more round of interaction, which introduces a long time delay. Instead, the proposed scheme needs only one round. Furthermore, it can correct the errors caused by the noise so that the efficiency of key distribution is improved vastly. Thus, the proposed scheme has a significant advantage against the other key management schemes in space network.

VI. CONCLUSION

In this work, we proposed a novel centralized identity-based key management scheme by using McEliece public key cryptosystem for space network. The certificateless public-key of each node is generated by using the identity-based McEliece public key generation scheme. The identity of space nodes is used as a partial parameter of the nodes' public key to provide authentication. Furthermore, it can correct the errors caused by noise and resist the quantum attack. The hash function is employed to generate the group key which keeps the forward and backward secrecy. During the distribution of the group key, the group key is encrypted by the node's public key so that the security and reliability are ensured simultaneously. The KDC election mechanism is designed to avoid single-point failure. This overcomes the critical flaw that the traditional centralized key management scheme is unavailable when the KDC is failed. The comparisons of the proposed scheme with other schemes show that our scheme has higher security, higher robustness, lower storage overhead and lower interaction rounds. Therefore, we can conclude that the proposed key management scheme is suitable in space networks for its excellent security, reliability and efficiency.

REFERENCES

[1] C. Jiang, X. Wang, J. Wang, H.-H. Chen, and Y. Ren, "Security in space information networks," *IEEE Commun. Mag.*, vol. 53, no. 8, pp. 82–88, Aug. 2015.
[2] P. McDaniel and A. Prakash, P. Honeyman, Antigone, "A flexible framework for secure group communication," in *Proc. 8th Conf. Secur. Symp.*, Washington, DC, USA, 1999, pp. 99–114.
[3] T. P. RaniC and J. Kumar, "Survey on key pre distribution for security in wireless sensor networks," *Computer Science and Information Technology*. Berlin, Germany: Springer, 2012, pp. 248–252.
[4] C. K. Wong, M. Gouda, and S. S. Lam, "Secure group communications using key graphs," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 28, no. 4, pp. 68–79, Oct. 1998.
[5] J.-H. Son, J.-S. Lee, and S.-W. Seo, "Topological key hierarchy for energy-efficient group key management in wireless sensor networks," *Wireless Pers. Commun.*, vol. 52, no. 2, pp. 359–382, Jan. 2009.
[6] K. Ahmad, B. Bakhache, S. E. Assad, and S. Sindian, "A scalable key management scheme for secure IP multicast over DVB-S using chaos," in *Proc. 16th IEEE Medit. Electrotechnical Conf.*, Hammamet, Tunisia, Mar. 2012, pp. 736–740.
[7] J. Zhou, M. Song, J. Song, X.-W. Zhou, and L. Sun, "Autonomic group key management in deep space DTN," *Wireless Pers. Commun.*, vol. 77, no. 1, pp. 269–287, Nov. 2013.
[8] G. Caparra, S. Ceccato, S. Sturaro, and N. Laurenti, "A key management architecture for GNSS open service navigation message authentication," in *Proc. Eur. Navigat. Conf. (ENC)*, Lausanne, Switzerland, May 2017, pp. 287–297.
[9] V. Sureshkumar, R. Amin, and R. Anitha, "An enhanced bilinear pairing based authenticated key agreement protocol for multiserver environment," *Int. J. Commun. Syst.*, vol. 30, no. 17, p. e3358, Jun. 2017.
[10] L. Harn, C. F. Hsu, and B. Li, "Centralized Group Key Establishment Protocol," *Mobile Netw. Appl.*, vol. 23, no. 5, pp. 1132–1140, 2018.
[11] L. Qian, Z. Wenlu, and S. Ningning, "Research of centralized multicast key management for LEO satellite networks," in *Proc. Int. Conf. Cyberspace Technol. (CCT)*, Beijing, China, 2013, pp. 577–582.
[12] M. Gharib, Z. Moradlou, M. A. Doostari, and A. Movaghar, "Fully distributed ECC-based key management for mobile ad hoc networks," *Comput. Netw.*, vol. 113, pp. 269–283, Feb. 2017.
[13] Q. Zhang, J. Yuan, G. Guo, Y. Gan, and J. Zhang, "An authentication key establish protocol for WSNs based on combined key," *Wireless Pers. Commun.*, vol. 99, no. 1, pp. 95–110, Nov. 2017.
[14] S.-H. Seo, J. Won, S. Sultana, and E. Bertino, "Effective key management in dynamic wireless sensor networks," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 2, pp. 371–383, Feb. 2015.
[15] S. Ali, A. Rauf, N. Islam, H. Farman, B. Jan, M. Khan, and A. Ahmad, "SGKMP: A scalable group key management protocol," *Sustain. Cities Soc.*, vol. 39, pp. 37–42, May 2018.
[16] Y. Harbi, Z. Aliouat, A. Refoufi, S. Harous, and A. Bentaleb, "Enhanced authentication and key management scheme for securing data transmission in the Internet of Things," *Ad Hoc Netw.*, vol. 94, Nov. 2019, Art. no. 101948.
[17] R. Misoczki, J.-P. Tillich, N. Sendrier, and P. S. L. M. Barreto, "MDPC-McEliece: New McEliece variants from moderate density parity-check codes," in *Proc. IEEE Int. Symp. Inf. Theory*, Istanbul, Turkey, Jul. 2013, pp. 2069–2073.
[18] I. V. Maurich and T. Guneysu, "Lightweight code-based cryptography: QC-MDPC McEliece encryption on reconfigurable devices," in *Proc. Design Autom. Test Eur. Conf. Exhibit. (DATE)*, New York, NY, USA, 2014, pp. 1–6.
[19] A. Becker, A. Joux, and A. May, "Decoding random binary linear codes in $2^{n/20}$: How 1+1=0 improves information set decoding," in *Cryptology-Eurocrypt*. Berlin, Germany: Springer, 2012, pp. 520–536.
[20] Q. Guo, T. Johansson, and P. Stankovski, "A key recovery attack on MDPC with CCA security using decoding errors," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.* Berlin, Germany: Springer, 2016.
[21] M. Baldi, A. Barenghi, F. Chiaraluce, G Pelosi, and P. Santini. EDApkc: Low-density parity-check code-bssed public-key cryptosystem. U.S. Specification Revision 1.0. Accessed: Nov. 30, 2017. [Online]. Available: https://csrc.nist.gov/Projects/post-quantum-cryptography/Round-1-Submissions

JIE LIU received the B.Eng. degree from the Department of Information Security, PLA Information Engineering University, in 2003, and the M.Eng. degree from the Department of Computer Software and Theory, Harbin University of Science and Technology, in 2007, where he is currently pursuing the Ph.D. degree with the Department of Computer Science and Technology. Since 2012, he has been a Lecturer with the Department of Software Engineering, Harbin University of Science and Technology. His research interests include chaos cryptosystems, coding-based cryptosystems, and key management.

XIAOJUN TONG is currently a Professor and the Ph.D. Supervisor with the Harbin Institute of Technology. Her current research interests are in the areas of chaos cryptography and information security. She is a member of the Program Committee for the International Workshop on Chaos-Fractals Theories and Applications.

MIAO ZHANG was an Assistant Professor with the Harbin Institute of Technology at Weihai, Weihai, from 2004 to 2006, where she has been a Lecturer, since 2006. She holds doctoral position at the Harbin Institute of Technology. Her current research interests are in the areas of chaos cryptography, information security, and image processing.

ZHU WANG is currently a Professor with the Harbin Institute of Technology. His current research interests are wireless sensor networks and networking technology, and single processing.

JING MA is currently an Engineer with the Science and Technology on Information Assurance Laboratory. Her interest is information security.