Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/mbe

MBE, 19(12): 12730–12743.
DOI: 10.3934/mbe.2022594
Received: 30 March 2022
Revised: 07 August 2022
Accepted: 22 August 2022
Published: 31 August 2022

Research article

Group key agreement protocol for edge computing in industrial internet

Yifeng Yin[1], **Zhaobo Wang**[1,*], **Wanyi Zhou**[1], **Yong Gan**[2] **and Yanhua Zhang**[1]

[1] School of Computer and Communication Engineering, Zhengzhou University of Light Industry, Zhengzhou 450002, China

[2] Zhengzhou Institute of Technology, Zhengzhou 450044, China

* **Correspondence:** Email: wangzhb7508@163.com.

Abstract: Industrial internet security is a critical component of cyberspace safety. Furthermore, the encryption protocol is a critical component of cyberspace security. Due to the rapid development of industrial internet and edge computing, increasingly more devices are outsourcing their data to cloud servers to save costs. Edge devices should have a secure session key to reduce communication costs and share information. However, most key generation and storage are completed by a centralized third-party organization, which carries some security risks. In this context, this paper will propose a lightweight multi-dimensional virtual iteration of the group key agreement protocol. Group key agreement protocol allows for one-at-a-time encryption and timely key updates without the involvement of a trusted third party, and each device in the network can agreement a large number of keys. According to the analysis of this protocol, it has high security, rapid computation speed, and little storage space.

Keywords: industrial internet; edge computing; share information; group key agreement; multi-dimensional virtual iteration

1. Introduction

With the rise of the Industrial Internet concept, intelligent transformation is happening in the industrial field. In the actual industrial scenario, numbers of devices will generate large industrial data sets [1–3]. The traditional cloud computing data model uses centralized processing, and massive amounts of data rely on remote cloud computing centers for computation and storage, putting tremendous pressure on the industrial Internet architecture based on the cloud computing model [4, 5]. In addition, unscrupulous elements can easily steal data during long-distance transmission, so it cannot guarantee data of security. Compared with cloud computing, edge computing can store and share resources near mobile devices, providing users with low latency and high bandwidth access to information and computing resources [6, 7].

In the edge computing architecture of the Industrial Internet, the cloud server is a trusted root that connects edge servers. An edge server is a domain administrator responsible for storing common data from edge devices and transferring complex data to the cloud server [8, 9]. Edge devices collect data and then upload it to a local edge server. Edge devices can share information. These edge servers in the edge computing layer are effective solutions for enhancing cloud computing capabilities [10, 11].

To reduce the communication consumption of edge devices and ensure the security of resource sharing among edge devices, only users of the group with the same key can decrypt this information. Edge devices in a group can negotiate a shared key and use the shared key to encrypt the data of members in the group. Most of the existing schemes are to store and manage the encryption and decryption keys of the data directly by a third-party authority (e.g., certificate authorities) [12–17]. Then we have to trust the third-party organization that the key management is safe and secure. This management scheme can threaten the entire system's security once the central institution is attacked. As a result, the research on reliable and secure group key protocols is still ongoing [18, 19].

Naresh [20] proposed a cluster-based hybrid layered group key protocol, which supports mutual authentication among group members, but consumes a lot of communication costs and is difficult to guarantee forward and backward confidentiality. Braeken [21] proposed a public asymmetric group key protocol for unpaired authentication, which does not require the participation of the certificate authority, but consumes a lot of communication. Chen et al. [22] proposed a broadcast encryption scheme based on anonymous certificates. In combination with broadcast encryption, it uses a shared key to transmit messages. But the scheme uses bilinear pair operation, which makes it computationally more expensive. Literature [23] proposed a group key protocol supporting anonymity, which can ensure user privacy security in the cloud environment. Literature [24,25] proposed the use of an elliptic curve cryptography (ECC) to establish secure session keys between authorized users and devices. Edge devices cannot support ECC because of their limited computation and storage capabilities. Literature [26] proposed a key management scheme based on the permutation algorithm, which can effectively improve the computational efficiency compared to the public key encryption scheme. The scheme is based on polymorphic cryptography and has high-security performance. Lu et al. [27] proposed an anonymous authentication group key protocol based on an ECC with complete forward and backward confidentiality and higher security performance. Wang et al. [28] proposed polynomial public key encryption, which can effectively resist key disclosure attacks.

Research shows that numerous group key protocols use asymmetric encryption algorithms, such as elliptical curve cryptography, bilinear pair encryption, polynomial encryption, etc. These public key encryption cryptosystems have a high computational and storage cost, making them unsuitable for encrypting a significant amount of data. Therefore, this paper will propose a lightweight multi-dimensional virtual iteration of the group key agreement protocol, which can effectively solve the shortcomings of the public key encryption cryptosystem.

The research contributions of this paper are as follows:

(1) We adopt group key agreement technology to construct an information sharing platform among edge devices. The scheme adopts hashing, multi-dimensional virtual iteration, and symmetric cryptography, which is suitable for edge devices with limited resources.

(2) Decentralization, with no trusted third parties. This scheme does not need certificate authority to generate and store keys and has higher security.

(3) Lightweight computing. This scheme adopts a symmetric encryption algorithm, key generation

is faster and more efficient and has low computational complexity, so it is suitable for massive amounts of data encryption. This scheme can satisfy the lightweight computing of edge devices.

(4) Safety evaluation. The multi-dimensional virtual iterative function $VIF()$ proposed in this protocol is self-compiled, and an attacker will face multi-dimensional variation if it is cracked forcibly. Through the theoretical and performance analysis of the protocol, it is proved that the proposed scheme has the characteristics of an anti-eavesdropping attack and anti-man-in-the-middle attack, and has fast running speed, small storage space, low computational complexity, and high security performance.

2. Model design

2.1. *Key distribution model of current Industrial Internet*

Data collection is the foundation of edge computing. In Industrial Internet, edge devices collect data, then control network for monitoring, and finally stored in the cloud center for unified processing. In cloud storage services, users of edge devices lose direct control of their data and prevent servers from maliciously stealing information, users encrypt data and then upload it to the cloud center, where the data stored in the cloud center is ciphertext information.

The current industrial Internet data storage model uses a centralized key distribution mechanism. And the key is generated and stored by a trusted third-party authority, which is usually a certificate authority (CA). As shown in Figure 1, the model contains three entities: Edge Device, Cloud and User. An Edge Device with limited resources will choose a trusted certification authority to host the key and then get the key from the authority when sharing the data, as shown in steps 1–8.

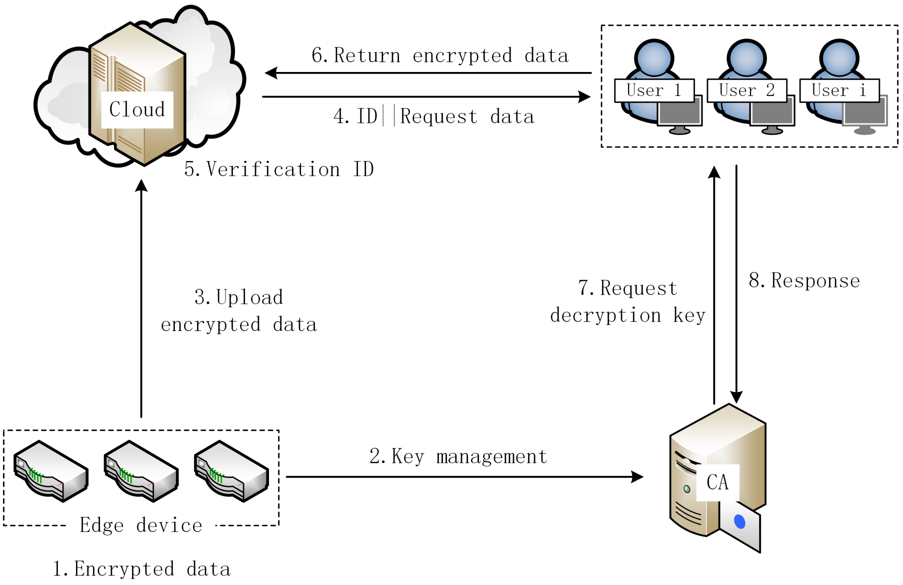

Figure 1. The Centralized key distribution model for the industrial internet.

The data model with a centralized key distribution mechanism is vulnerable to threats. If someone maliciously attacks the certificate authority, the encrypted and decrypted session keys will also become insecure, resulting in data privacy leakage.

2.2. Group key agreement based on multi-dimensional virtual iteration

In order to ensure the security of data shared by edge devices, a group key agreement protocol based on multi-dimensional virtual iteration is proposed, as shown in Figure 2.

Cloud: The cloud computing center provides relevant services to edge devices with storage space and computing resources. The edge devices get the required services through the network according to their needs.

Edge Server: Store data commonly used by edge devices and provide relevant services to edge devices on time.

Edge Devices: Edge devices implement the generation and processing of source data. Edge Devices can establish secure communication channels and negotiate session keys for securing data transmission between devices. Provide the required services to devices.

After edge devices obtains the transmitted information from the database, the server completes and exits the key agreement. The edge devices store and manage the information. When the key agreement ends or before the next key agreement starts, the server is asked to update the data to be sent again. The server works only in the current step and does not participate in subsequent key agreements, nor can it get key information and the plaintext and ciphertext transmitted.

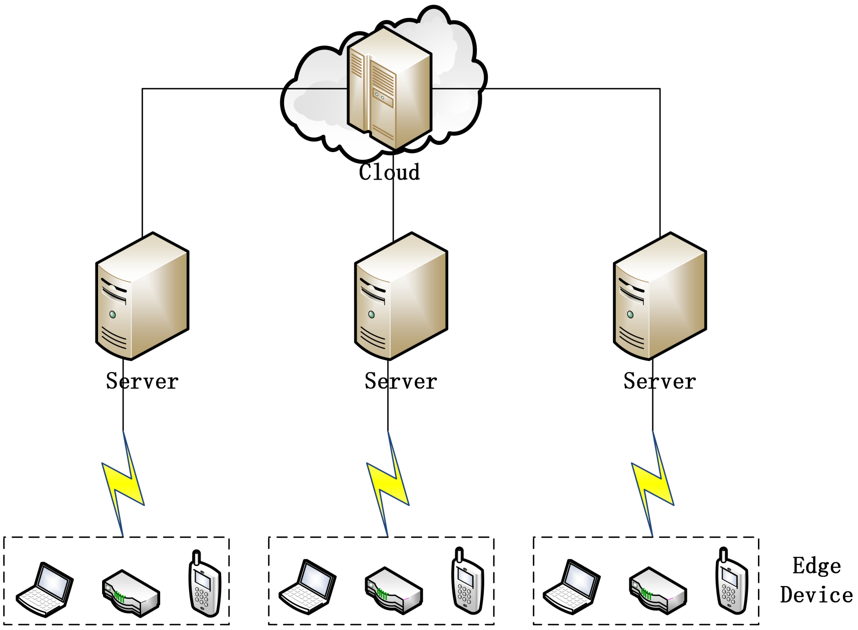

Figure 2. Group key agreement protocol model based on multi-dimensional virtual iteration.

During the group key agreement, each group of users can uniquely determine the key array on a single dimension. After m devices interact with their respective key arrays $K_i[n]$ through the token ring, the self-compiling of the function is triggered and a multi-dimensional virtual iteration space is constructed. The multi-dimensional virtual iteration space contains $m \times n$ security subsystems and is uniformly distributed. The group users permute and iterate over a number of security subsystems to get encryption keys. Repeatedly performing the permutation step, users get a lot of different secure encryption methods.

When the number of users is large enough, it is impossible to achieve a one-to-one correspondence between the number of security subsystems and the number of subspaces. So a Sha-256 cryptographic hash function needs to be defined to achieve a one-to-many mapping relationship between the number of subspaces and the security subsystems. The arbitrary length of the input, after a hash function encryption processing, can output the fixed-length information, make it uniformly distribute the fixed security subsystem to more than one way, then the multi-dimensional permutation space is a virtual fully populated state. From the user's point of view, it is a fully populated state. If this protocol wants to implement a group key agreement, it must meet the following conditions.

(1) Multi-party devices have the same multi-dimensional virtual iteration space S_{mn}, $m \times n$ security subsystems.

$$\overbrace{\begin{pmatrix} S_{11} & \dots & S_{1n} \\ \vdots & \ddots & \vdots \\ S_{m1} & \cdots & S_{mn} \end{pmatrix}}^{m*n} \tag{1}$$

(2) The device D_i holds the private key k_i and the private key array k_i', which cannot be used directly. The private key and the private key array need to be processed by a function to form the key control array $K_i[n]$, $VIF()$ which is used to construct a multi-dimensional iterative function .

(3) The devices have the same multi-dimensional virtual iteration function $VIF()$. It generates a unique and shared multi-dimensional virtual permutation space for the m devices.

(4) The multi-dimensional virtual iterative function $VIF()$ is self-compiling. If an attacker forces a hack on the function, he will face a multi-dimensional variant of the function.

3. Protocol design

3.1. Construct the four dimensional permutation network

Table 1 lists the basic symbols used in this article and their definitions.

This model comprises four users, where X-axis, Y-axis, Z-axis and color axis are the four-dimensional concrete coordinate axes. Four users provide a single-dimensional key control sequence to control a single-dimensional coordinate axis in the four-dimensional virtual permutation space. Achieving a distribution of four users that collaboratively control the safety subsystem and are in the model space. At last, x security subsystems are determined in the four-dimensional virtual iterative model space constructed by the function coordinates provided by the four users, as shown in Figure 3.

In this protocol, the initial key array $K_i[1], K_i[2], \cdots, K_i[n]$ held by device D_i is used as the i-th coordinate parameter. Each dimension represents a key array controlled by the device, where the color axis represents the fourth dimension. Assuming that the number of keys in each key array is 4, there are $4^4 = 256$ security subsystems in the key space constructed by four devices. When the devices determines a set of coordinates, the probability of replacing a security subsystem immediately is $\frac{1}{256}$.

Table 1. Notations and descriptions.

Symbol	Description
k_i	the private key of device i
$k_i^{`}$	the private key array of device i
ur_i	the network upload speed intercepted by device i
dr_i	the network download speed intercepted by device i
ut_i	the total number of network packet upload intercepted by device i
dt_i	the total number of network packet download intercepted by device i
S_{mn}	$m \times n$ security subsystems shared by devices
$Rand\,()$	pseudo-random function
$K_i\,[n]$	key control array
Token	token ring
$H\,()$	hash function of Sha256 encryption algorithm
$VIF\,(x, y)$	virtual iterating function with arguments x and y
$IVIF\,(i)$	virtual iterator function without the i-th device
ϕ_i	the i-th security subsystem generated by the devices
IFS^x	iterating function system x times

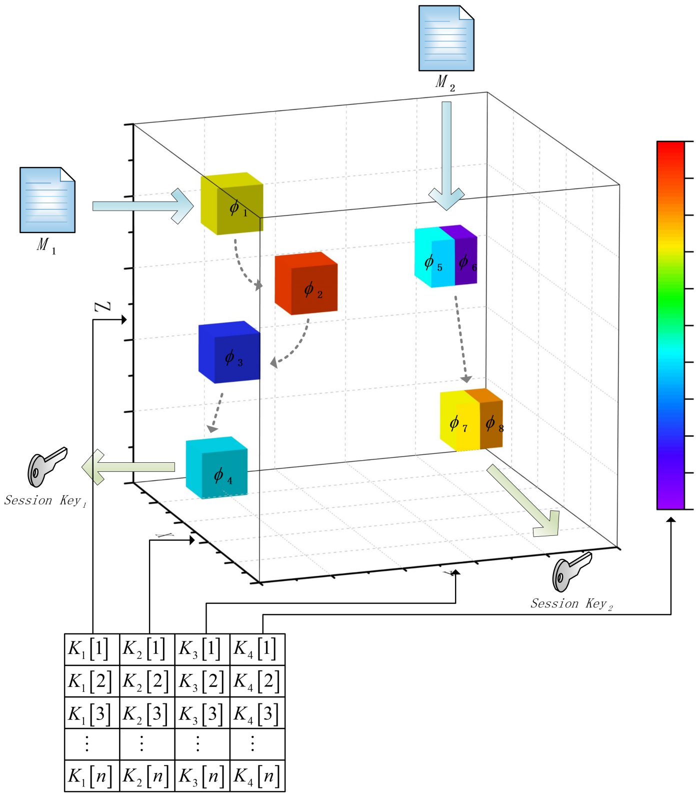
M_2
M_1
ϕ_1
ϕ_2
ϕ_3
ϕ_4
ϕ_5
ϕ_6
ϕ_7
ϕ_8
Z
Session Key$_1$
Session Key$_2$
$K_1[1]$ $K_2[1]$ $K_3[1]$ $K_4[1]$
$K_1[2]$ $K_2[2]$ $K_3[2]$ $K_4[2]$
$K_1[3]$ $K_2[3]$ $K_3[3]$ $K_4[3]$
⋮ ⋮ ⋮ ⋮
$K_1[n]$ $K_2[n]$ $K_3[n]$ $K_4[n]$

Figure 3. Four-dimensional virtual iterative model space.

Due to the permutation network contains 4^n candidate security subsystems, the function of the key array $K_i[1], K_i[2], \cdots, K_i[n]$ is to realize the polymorphism overloading principle in the virtual function. The key array arguments determine the security subsystem to be called and executed with equal probability, and finally, make the permutation network obtain a unique internal permutation array.

3.2. Protocol process

As shown in Figure 4, the steps of multi-dimensional virtual iterative group key agreement are as follows:

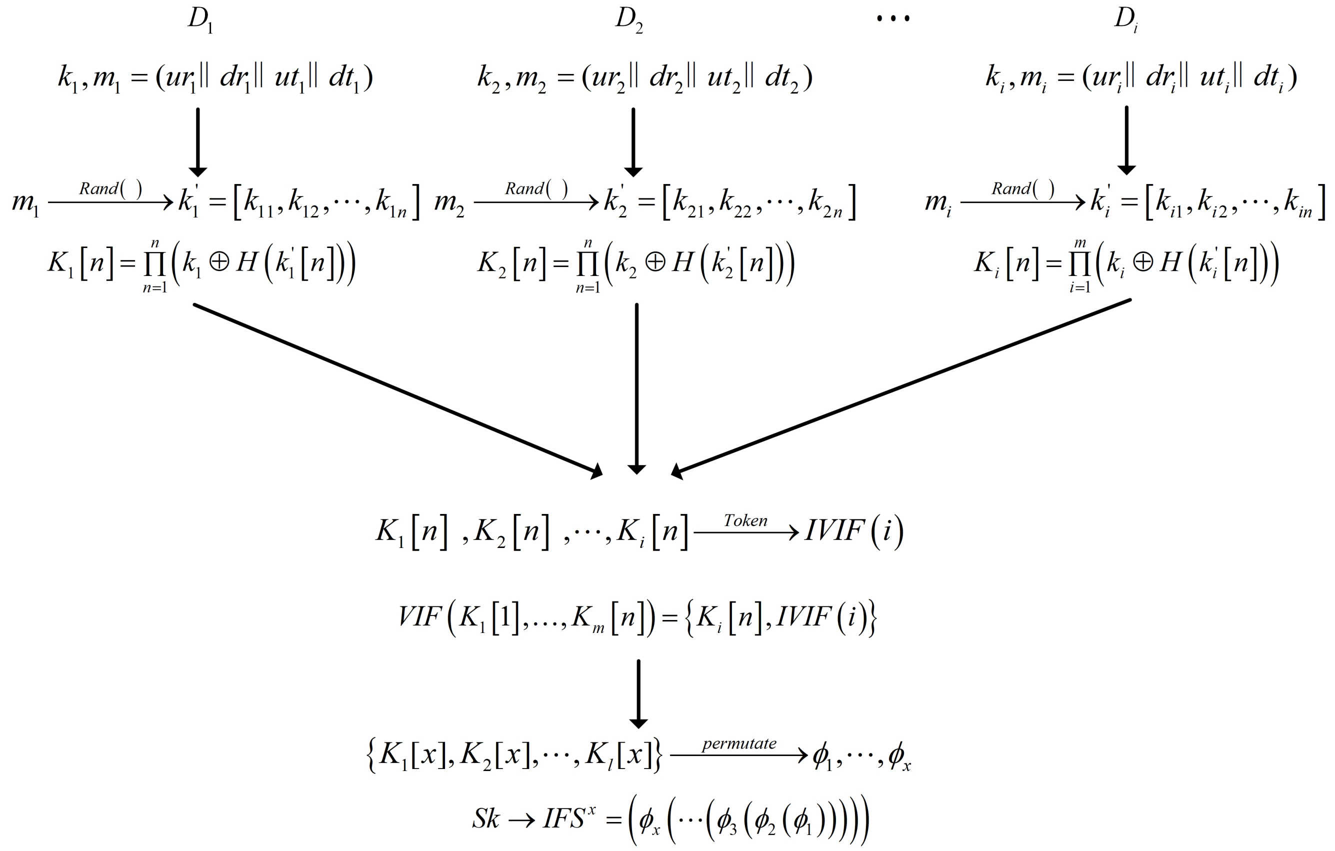
D_1
D_2
$\cdots$
D_i
$k_1, m_1 = (ur_1 \| dr_1 \| ut_1 \| dt_1)$
$k_2, m_2 = (ur_2 \| dr_2 \| ut_2 \| dt_2)$
$k_i, m_i = (ur_i \| dr_i \| ut_i \| dt_i)$
$m_1 \xrightarrow{Rand(\ )} k_1^{'} = [k_{11}, k_{12}, \cdots, k_{1n}]$
$m_2 \xrightarrow{Rand(\ )} k_2^{'} = [k_{21}, k_{22}, \cdots, k_{2n}]$
$m_i \xrightarrow{Rand(\ )} k_i^{'} = [k_{i1}, k_{i2}, \cdots, k_{in}]$
$K_1[n] = \prod_{n=1}^{n} (k_1 \oplus H(k_1^{'}[n]))$
$K_2[n] = \prod_{n=1}^{n} (k_2 \oplus H(k_2^{'}[n]))$
$K_i[n] = \prod_{i=1}^{m} (k_i \oplus H(k_i^{'}[n]))$
$K_1[n], K_2[n], \cdots, K_i[n] \xrightarrow{Token} IVIF(i)$
$VIF(K_1[1], \ldots, K_m[n]) = \{K_i[n], IVIF(i)\}$
$\{K_1[x], K_2[x], \cdots, K_l[x]\} \xrightarrow{permutate} \phi_1, \cdots, \phi_x$
$Sk \to IFS^x = (\phi_x(\cdots(\phi_3(\phi_2(\phi_1)))))$

Figure 4. Four-dimensional virtual iterative protocol.

(1) Initialization step: Device D_i holds the private key k_i, ur_i represents the network upload speed intercepted by device D_i, dr_i represents the network download speed intercepted by device D_i, ut_i represents the total number of network packet upload intercepted by device D_i, and dt_i represents the total number of network packet download intercepted by device D_i. At this point, device D_i converts the contained information into binary form and stores it in $m_i = (ur_i \| dr_i \| ut_i \| dt_i)$. Each user also includes a unique virtual iterator function $VIF()$. At the same time, Each device monitors the token ring and waits for the key protocol.

(2) Preparation step: Each device D_i uses the pseudo-random function $Rand()$ to pseudo-randomly

generate the existing key sequence $k_i^{\cdot} = [k_{i1}, k_{i2}, \cdots, k_{in},]$ from the existing m_i, the group user D_i takes the private key k_i and the private key array $k_i^{\cdot}$, device D_i uses the private key k_i and the hashed private key array $k_i^{\cdot}$ XOR to generate the real key array $K_i[n]$, as shown in Eq (2). The user waits for instructions from the token ring in sequence.

$$K_i[n] = \prod_{n=1}^{n} \left(k_i \oplus H\left(k_i^{\cdot}[n] \right) \right) \tag{2}$$

Device D_i has a key array $K_i[n]$, device D_m has a key array $K_m[n]$. And the system contains $m \times n$ security subsystems. When D_i get the token, device D_i secretly put $K_i[n]$ into $VIF()$ for subsequent virtual iteration space distribution.

(3) Construction step: The device starts a data sharing request, first obtains the token ring, uploads its own key array $K_i[n]$ to the multi-dimensional virtual iteration function $IVIF(i)$, and upon successful upload gets an incomplete virtual iteration function $VIF()$ except for the missing own key array only, as shown in Eq (3).

$$IVIF(i) = VIF(K_i[n], \cdots, K_{i-1}[n], 0, K_{i+1}[n], \cdots, K_m[n]) \tag{3}$$

After that, the device substitutes the key array $K_i[n]$ held by itself into $IVIF(i)$ to obtain the complete multi-dimensional virtual iteration function $VIF()$. The multi-dimensional virtual iteration function triggers the self-compilation function and maps onto a multi-dimensional permutation network, which contains $m \times n$ secure subsystems in the system. This space is a filled state, and m devices share this key space. K denotes the shared multi-dimensional permutation network, as show in Eq (4).

$$K = VIF(K_1[n], \cdots, K_{i-1}[n], K_i[n], K_{i+1}[n], \cdots, K_m[n]) \tag{4}$$

(4) Iterative step

Step 1: In the four-dimensional permutation network, the i-th coordinate selected by the four devices is a security subsystem in the permutation network, represented as ϕ_i. Each user randomly selects x initial keys. Then rapidly iterate x security subsystems, denoted as $\phi_1, \phi_2, \cdots, \phi_x$ in order.

Step 2: Generate an iterative function system IFS^x is generated by iterating x security subsystems obtained in Step 1 in a certain order through the self-compiler. Here, x is denoted as Eq (4), as shown in Eq (5).

$$SK \to IFS^4 = (\phi_4(\phi_3(\phi_2(\phi_1)))) \tag{5}$$

At last, the result of iteration can be used as the unique secure encryption key SK to provide the key array of a long period for various symmetric encryption algorithms to complete the key negotiation. Members encrypt their own information about the session key SK and send it to other users. Since the session key SK is unique, all members can decrypt the cipher text to get the information transmitted by other users.

Step 3: When each member negotiates the key again, cyclic Steps 1 and 2 can quickly replace the corresponding virtual iteration function only to generate a large number of secure keys.

4. Safety analysis and performance evaluation

4.1. Safety analysis

(1) Resist playback attack: According to the description of this protocol, the virtual iteration function needs to be updated every time multiple users share information. If an attacker steals the n-th shared key, the replay of the n-th key during the (n + 1)-th information sharing fails. At this point, the virtual iteration function has changed, and the attacker will not steal the information interaction of the n+1, so it can resist replay attacks.

(2) Resist eavesdropping attack: Each user's key array can only control one dimension of the virtual iteration function, so eavesdropping on a single or multiple users cannot get a complete virtual iteration function; Similarly, the coordinate of the iteration function is determined by each user, so the eavesdropper can't get the correct virtual iteration function. Therefore, it can resist eavesdropping attacks.

(3) Decentralization: This protocol does not require any third-party organization. Each device negotiates the key together and stores its information data independently. Even if a device is compromised by an attacker resulting in data leakage, the attacker cannot analyze the shared keys from the information obtained. Then it is not possible to get the information of others, so it will not affect the next data share of the device.

(4) Prevent man-in-the-middle attack: During the period of information sharing, an attacker might get $\left\{k_i, k_i^{'}, k_i\left[n\right]\right\}$ in the transmission channel, but the virtual iteration function and key space build all need security subsystem S_{mn}, S_{mn} does not exchange information in the transmission channel, and the attacker can obtain safety subsystems, also will not participate in the iteration phase to get the key SK. Even if the attacker gets $VIF\left(k_1\left[n\right], \cdots, k_{i-1}\left[n\right], 0, k_{i+1}\left[n\right], \cdots, k_m\left[n\right]\right)$, because $VIF\left(\right)$ is self-compiled and unreadable, if the attacker forces $VIF\left(\right)$, with each crack, the attacker will end up facing the multidimensional variance function $VIF\left(\right)$.

$$\begin{array}{c}
\left\{\begin{array}{l}
VIF^{(1,1,\cdots,1,1)}\left(K_1\left[n\right], \cdots, K_{i-1}\left[n\right], K_i\left[n\right], K_{i+1}\left[n\right], \cdots, K_m\left[n\right]\right) \\
VIF^{(1,1,\cdots,1,2)}\left(K_1\left[n\right], \cdots, K_{i-1}\left[n\right], K_i\left[n\right], K_{i+1}\left[n\right], \cdots, K_m\left[n\right]\right) \\
\cdots\cdots \\
VIF^{(1,1,\cdots,1,n)}\left(K_1\left[n\right], \cdots, K_{i-1}\left[n\right], K_i\left[n\right], K_{i+1}\left[n\right], \cdots, K_m\left[n\right]\right)
\end{array}\right\} \\
\left\{\begin{array}{l}
VIF^{(1,1,\cdots,1,1)}\left(K_1\left[n\right], \cdots, K_{i-1}\left[n\right], K_i\left[n\right], K_{i+1}\left[n\right], \cdots, K_m\left[n\right]\right) \\
VIF^{(1,1,\cdots,2,1)}\left(K_1\left[n\right], \cdots, K_{i-1}\left[n\right], K_i\left[n\right], K_{i+1}\left[n\right], \cdots, K_m\left[n\right]\right) \\
\cdots\cdots \\
VIF^{(1,1,\cdots,n,1)}\left(K_1\left[n\right], \cdots, K_{i-1}\left[n\right], K_i\left[n\right], K_{i+1}\left[n\right], \cdots, K_m\left[n\right]\right)
\end{array}\right\} \\
\cdots\cdots \\
\left\{\begin{array}{l}
VIF^{(1,1,\cdots,1,1)}\left(K_1\left[n\right], \cdots, K_{i-1}\left[n\right], K_i\left[n\right], K_{i+1}\left[n\right], \cdots, K_m\left[n\right]\right) \\
VIF^{(2,1,\cdots,1,1)}\left(K_1\left[n\right], \cdots, K_{i-1}\left[n\right], K_i\left[n\right], K_{i+1}\left[n\right], \cdots, K_m\left[n\right]\right) \\
\cdots\cdots \\
VIF^{(n,1,\cdots,1,1)}\left(K_1\left[n\right], \cdots, K_{i-1}\left[n\right], K_i\left[n\right], K_{i+1}\left[n\right], \cdots, K_m\left[n\right]\right)
\end{array}\right\}
\end{array} \tag{6}$$

(5) Mutual Authentication: In constructing the key space, all users participate in the spaces construction. An attacker cant get the key by participating in the token ring as an illegal user. Therefore, the resulting key space is identical and unique. If the key space generated by a user is different from that of another user, the user is invalid. The multi-dimensional virtual permutation mechanism contains the

unique permutation network generated by all users' security subsystems with equal probability, which is equivalent to the user's signature information. So it achieves mutual authentication between users.

4.2. Performance evaluation

Computational complexity and time overhead are the main performance indicators of protocol optimization. This section summarizes several encryption methods with four representative references [18,29–31].

Table 2. The symbols and data used in the performance analysis.

Operations	Symbol	Times(ms)
HMAC-SHA-256	T_h	0.067
RAND	T_{rand}	0.045
Modular inversion	T_{inv}	0.124
Bilinear pairing	T_{bp}	4.514
Multiplication over elliptic curve	T_{mul}	0.612
Addition over elliptic curve	T_{add}	0.125
Symmetric encryption operation	$T_{Ek()/Dk()}$	0.161

We used the advantage of the Crypto ++Library to measure the elapsed time of the cryptographic operations. The computer used for this test was an Ubuntu 11.10 operating system with an Intel Core Duo 1.86 GHz and 2 gigabytes of RAM. It showed the symbols and data used in the performance analysis in Table 2. Table 3 compares the time cost of the research protocol in this paper with the references and calculates the theoretical test time.

Table 3. Computational cost comparison.

Reference	T_{rand}	T_{bp}	T_{mul}	T_{add}	T_{inv}	T_h	$T_{Ek()/Dk()}$	sum
Cheng	2	2n+2	n+3	2n-4	1	4	-	$2T_{rand} + (2n+2)\,T_{bp} + (n+3)\,T_{mul} + (2n-4)\,T_{add} + T_{inv} + 4T_h$
Vinoth	-	-	2n	-	-	10n+9	4n+2	$2nT_{mul} + (10n+9)\,T_h + (4n+2)\,T_{EK()/DK()}$
Islam	3	-	2n+4	2n-1	-	4	-	$3T_{rand} + (2n+4)\,T_{mul} + (2n-1)\,T_{add} + 4T_h$
Wang	3	n+2	5	-	-	2n+6	-	$3T_{rnd} + 2T_{bp} + 5T_{mul} + (2n+6)\,T_h$
The proposed	1	-	-	-	-	n	n	$1T_{rand} + nT_h + nT_{Ek()/Dk()}$

Cheng et al. [29] proposed an authentication group key protocol based on Diffie-Hellman. In this protocol, the user's static private key and temporary key are bound to provide bidirectional authentication and resist temporary key leakage attack. However, extensive bilinear pairing, elliptic curve scalar multiplication and elliptic curve scalar addition are adopted. Therefore, the calculation time cost is relatively high.

Vinoth et al. [18] proposed to use secret sharing technology to construct a group key negotiation. Using only hashing function, XOR, and symmetric cryptography in the scheme can have lower communication overhead. Compared to Cheng, the computational overhead is lower.

Islam et al. [30] proposed an unpaired authentication group key protocol based on elliptic curve cryptography. It eliminates public key certificates and reduces computing costs. But in practice, a lot of circular curve scalar multiplication and elliptic curve scalar addition operations are still used.

Wang et al. [31] proposed a group key agreement based on device-to-device. The public key encryption method is used in the protocol, which has high security performance but requires a lot of calculation. Communication is too expensive compared to our protocol.

Figure 5 simulates the time required for the key agreement of our proposed protocol and other schemes. The number of key agreement users ranges from 1 to 50. With the increase of users, we can conclude that Cheng's scheme takes the longest time, while our scheme consumes less calculation and has better performance.

Compared to these protocols, our scheme has a shorter computation time. And has a stronger defense capability. In the construction phase of the multi-dimensional iterative space, with the increase of the number of devices, although it will relatively increase a small amount of storage overhead, the multi-dimensional replacement space and virtual iteration function constructed in the negotiation phase to achieve ”one key at a time”, close to the random state, can ensure the security problem. In addition, this protocol uses a symmetric encryption algorithm, which can quickly iterate through a large number of keys and encrypt data, greatly reducing the time overhead.

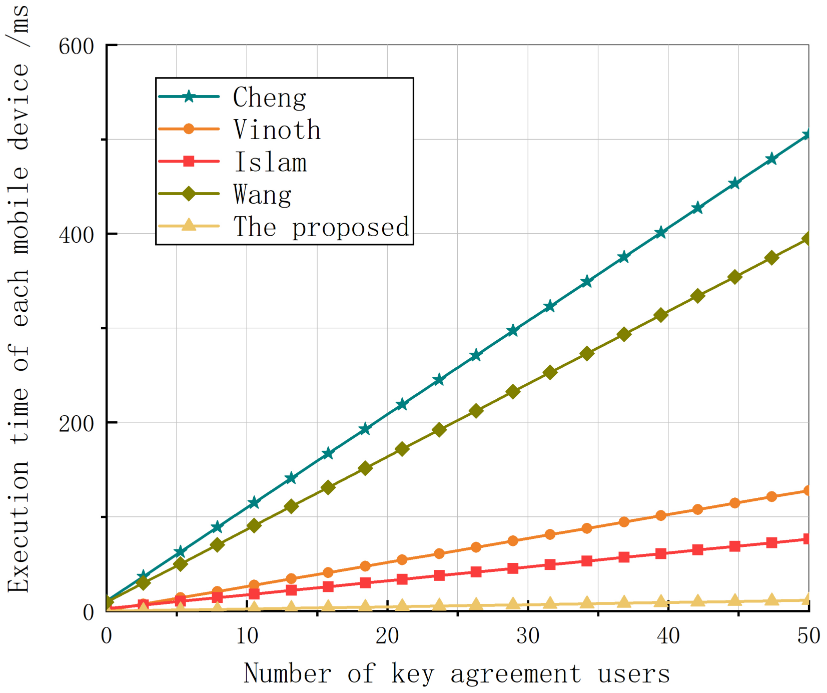

Figure 5. The time required for key agreement of our proposed protocol and other schemes.

5. Conclusions

In building industrial internet, it is necessary to pay attention to the security of shared keys of edge devices. Compared with the previous centralized key protocol, this protocol can better protect security and reduce data transmission time without resorting to any third party.

This paper proposes a multi-dimensional virtual iterative key protocol without certificates. Compared with the asymmetric protocol in references [18,29–31], it features short computation time, small storage space, and large encrypted plaintext data, which can quickly replace a large number of secure keys. Meanwhile, the multi-dimensional virtual iterative key space proposed by this protocol has good security against man-in-the-middle attacks and replay attacks. In the edge environment of industrial Internet, there are problems of large data volume and low security. This protocol has good practicality and application space for such platforms, also does not need to bear a large amount of computational consumption.

As the number of participating users increases, the storage cost will increase accordingly. Reducing storage consumption and increasing user identity authentication will also be the focus of future research.

Acknowledgments

This paper is supported by the Natural Science Foundation of Henan Province of China (Grant No.202300410508), the Natural Science Foundation of Henan Province (Grant No.222300420371), the Key Research Project of Higher Education of Henan Province (Grant No. 22A520047), the National Natural Science Foundation of China under Grant U1804263, and key foundation of Science and Technology Development of Henan Province No.142102210081.

Conflict of interest

The authors declare that there are no conflicts of interest.

References

1. D. Wang, An enterprise data pathway to industry 4.0, *IEEE Eng. Manag. Rev.*, **46** (2018), 46–48. https://doi.org/10.1109/EMR.2018.2866157
2. Z. Cai, X. Zheng, A private and efficient mechanism for data uploading in smart cyber-physical systems, *IEEE Trans. Network Sci. Eng.*, **7** (2020), 766–775. https://doi.org/10.1109/TNSE.2018.2830307
3. Y. Huo, C. Meng, R. Li, T. Jing, An overview of privacy preserving schemes for industrial internet of things, *China Commun.*, **17** (2020), 1–18. https://doi.org/10.23919/JCC.2020.10.001
4. J. Shen, T. Zhou, D. He, Y. Zhang, X. Sun, Y. Xiang, Block design-based key agreement for group data sharing in cloud computing, *IEEE Trans. Depend. Secure Comput.*, **16** (2019), 996–1010. https://doi.org/10.1109/TDSC.2017.2725953
5. Z. Zhang, L. Huang, R. Tang, T. Peng, L. Guo, X. Xiang, Industrial blockchain of things: A solution for trustless industrial data sharing and beyond, in *2020 IEEE 16th International Conference on Automation Science and Engineering (CASE)*, (2020), 1187–1192. https://doi.org/10.1109/CASE48305.2020.9216817
6. C. Pham, D. T. Nguyen, Y. Njah, N. H. Tran, K. K. Nguyen, M. Cheriet, Share-to-run iot services in edge cloud computing, *IEEE Int. Things J.*, **9** (2022), 497–509. https://doi.org/10.1109/JIOT.2021.3085777

7. T. Qiu, J. Chi, X. Zhou, Z. Ning, M. Atiquzzaman, D. O. Wu, Edge computing in industrial internet of things: Architecture, advances and challenges, *IEEE Commun. Surv. Tut.*, **22** (2020), 2462–2488. https://doi.org/10.1109/COMST.2020.3009103

8. F. Saeik, M. Avgeris, D. Spatharakis, N. Santi, D. Dechouniotis, J. Violos, et al., Task offloading in edge and cloud computing: A survey on mathematical, artificial intelligence and control theory solutions, *Comput. Networks*, **195** (2021), 108177. https://www.sciencedirect.com/science/article/pii/S1389128621002322

9. K. Fan, Q. Pan, J. Wang, T. Liu, H. Li, Y. Yang, Cross-domain based data sharing scheme in cooperative edge computing, in *2018 IEEE International Conference on Edge Computing (EDGE)*, (2018), 87–92. https://doi.org/10.1109/EDGE.2018.00019

10. X. Jia, D. He, N. Kumar, K. K. R. Choo, A provably secure and efficient identity-based anonymous authentication scheme for mobile edge computing, *IEEE Syst. J.*, **14** (2020), 560–571. https://doi.org/10.1109/JSYST.2019.2896064

11. X. Wang, Z. Zhou, P. Han, T. Meng, G. Sun, J. Zhai, Edge-stream: A stream processing approach for distributed applications on a hierarchical edge-computing system, in *2020 IEEE/ACM Symposium on Edge Computing (SEC)*, (2020), 14–27. https://doi.org/10.1109/SEC50012.2020.00009

12. E. Abirami, T. Padmavathy, Proficient key management scheme for multicast groups using group key agreement and broadcast encryption, in *2017 International Conference on Information Communication and Embedded Systems (ICICES)*, (2017), 1–5. https://doi.org/10.1109/ICICES.2017.8070789

13. R. Sharma, S. Kumar, M. C. Trivedi, Mobile cloud computing: A needed shift from cloud to mobile cloud, in *2013 5th International Conference and Computational Intelligence and Communication Networks*, (2013), 536–539. https://doi.org/10.1109/CICN.2013.116

14. M. Anisetti, C. A. Ardagna, E. Damiani, F. Gaudenzi, A semi-automatic and trustworthy scheme for continuous cloud service certification, *IEEE Trans. Serv. Comput.*, **13** (2020), 30–43. https://doi.org/10.1109/TSC.2017.2657505

15. Y. Wang, W. Zhang, X. Wang, M. K. Khan, P. Fan, Efficient privacy-preserving authentication scheme with fine-grained error location for cloud-based vanet, *IEEE Trans. Veh. Technol.*, **70** (2021), 10436–10449. https://doi.org/10.1109/TVT.2021.3107524

16. K. Yu, Z. Guo, Y. Shen, W. Wang, J. C.W. Lin, T. Sato, Secure artificial intelligence of things for implicit group recommendations, *IEEE Int. Things J.*, **9** (2022), 2698–2707. https://doi.org/10.1109/JIOT.2021.3079574

17. L. Tan, K. Yu, F. Ming, X. Cheng, G. Srivastava, Secure and resilient artificial intelligence of things: A honeynet approach for threat detection and situational awareness, *IEEE Consum. Electr. Mag.*, **11** (2022), 69–78. https://doi.org/10.1109/MCE.2021.3081874

18. R. Vinoth, L. J. Deborah, P. Vijayakumar, N. Kumar, Secure multifactor authenticated key agreement scheme for industrial iot, *IEEE Int. Things J.*, **8** (2021), 3801–3811. https://doi.org/10.1109/JIOT.2020.3024703

19. Q. Zhang, L. Zhu, R. Wang, J. Li, J. Yuan, T. Liang, et al., Group key agreement protocol among terminals of the intelligent information system for mobile edge computing, *Int. J. Intell. Syst.*, (2021). https: /doi.org/10.1002/int.22544

20. V. S. Naresh, S. Reddi, N. V. E. S. Murthy, A provably secure cluster-based hybrid hierarchical group key agreement for large wireless ad hoc networks, *Human-centric Comput. Inform. Sci.*, **9** (2019), 26. https://doi.org/10.1186/s13673-019-0186-5

21. A. Braeken, Pairing free certified common asymmetric group key agreement protocol for data sharing among users with different access rights, *Wireless Pers. Commun.*, **121** (2021). 307–318, https://doi.org/10.1007/s11277-021-08636-4

22. L. Chen, J. Li, Y. Zhang, Anonymous certificate-based broadcast encryption with personalized messages, *IEEE Trans. Broadcast.*, **66** (2020), 867–881. https://doi.org/10.1109/TBC.2020.2984974

23. K. Yu, L. Tan, C. Yang, K. K. R. Choo, A. K. Bashir, J. J. P. C. Rodrigues, et al., A blockchain-based shamir¡¯s threshold cryptography scheme for data protection in industrial internet of things settings, *IEEE Int. Things J.*, **9** (2022), 8154–8167. https://doi.org/10.1109/JIOT.2021.3125190

24. X. Li, J. Niu, M. Z. A. Bhuiyan, F. Wu, M. Karuppiah, S. Kumari, A robust ecc-based provable secure authentication protocol with privacy preserving for industrial internet of things, *IEEE Trans. Indust. Inform.*, **14** (2018), 3599–3609. https://doi.org/10.1109/TII.2017.2773666

25. J. Shen, S. Chang, J. Shen, Q. Liu, X. Sun, A lightweight multi-layer authentication protocol for wireless body area networks, *Future Gener. Comput. Syst.*, **78** (2018). 956–963, https://doi.org/10.1016/j.future.2016.11.033

26. Y. Yin, K. Liu, C. Hu, Y. Gan, The group key agreement protocol based on multi-dimensional virtual permutation, *IEEE Commun. Letters*, **24** (2020), 2728–2732. https://doi.org/10.1109/LCOMM.2020.3017660

27. Y. Lu, D. Zhao, An anonymous sip authenticated key agreement protocol based on elliptic curve cryptography, *Math. Biosci. Eng.*, **19** (2022), 66–85. https://doi.org/10.3934/mbe.2022003

28. X. Wang, B. Yang, An improved signature model of multivariate polynomial public key cryptosystem against key recovery attack, *Math. Biosci. Eng.*, **16** (2019), 7734–7750. https://doi.org/10.3934/mbe.2019388

29. Q. Cheng, C. Ma, F. Wei, Analysis and improvement of a new authenticated group key agreement in a mobile environment, *Ann. Telecommun.*, **66** (2011), 331–337., https://doi.org/10.1007/s12243-010-0213-z

30. R. Vinoth, L. J. Deborah, P. Vijayakumar, N. Kumar, Secure multifactor authenticated key agreement scheme for industrial iot, *IEEE Int. Things J.*, **8** (2021), 3801–3811. https://doi.org/10.1109/JIOT.2020.3024703

31. S. K. H. Islam, G. P. Biswas, A pairing-free identity-based authenticated group key agreement protocol for imbalanced mobile networks, *Ann. Telecommun.*, **67** (2012), 547–558. https://doi.org/10.1007/s12243-010-0213-z